Stratix IV Device Handbook, Volume 1

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

SIV5V1-2.0

Contents

Chapter Revision Dates xiii

Additional Information

About this Handbook . . . . .Info-xv
How to Contact Altera . . . . .Info-xv
Typographic Conventions . . . . .Info-xv

Section I. Device Core

Revision History I-1

Chapter 1. Stratix IV Device Family Overview

Introduction 1-1
Feature Summary 1-1
Stratix IV GX Devices 1-3
Stratix IV E Device Family 1-3
Architecture Features 1-4
High-Speed Transceiver Features 1-4
Highest Aggregate Data Bandwidth 1-4
Wide Range of Protocol Support 1-4
Stratix IV Device Diagnostic Features 1-5
Signal Integrity 1-6
FPGA Fabric and I/O Features 1-6
Device Core Features 1-6
System Integration 1-8
Integrated Software Platform 1-12
Ordering Information 1-13
Referenced Documents 1-13
Document Revision History 1-14

Chapter 2. Logic Array Blocks and Adaptive Logic Modules in Stratix IV Devices

Introduction 2-1
Logic Array Blocks 2-1
LAB Interconnects 2-3
LAB Control Signals 2-4
Adaptive Logic Modules 2-6
ALM Operating Modes 2-8
Normal Mode 2-8
Extended LUT Mode 2-10
Arithmetic Mode 2-11
Shared Arithmetic Mode 2-13
LUT-Register Mode 2-14
Register Chain 2-15
ALM Interconnects 2-17
Clear and Preset Logic Control 2-17
LAB Power Management Techniques 2-18
Conclusion 2-18
Referenced Documents 2-18

Document Revision History 2-19

Chapter 3. TriMatrix Embedded Memory Blocks in Stratix IV Devices

Introduction 3-1
Overview 3-1
TriMatrix Memory Block Types 3-3
Parity Bit Support 3-3
Byte Enable Support 3-3
Packed Mode Support 3-4
Address Clock Enable Support 3-4
Mixed Width Support 3-6
Asynchronous Clear 3-6
Error Correction Code (ECC) Support 3-7
Memory Modes 3-8
Single-Port RAM 3-8
Simple Dual-Port Mode 3-10
True Dual-Port Mode 3-12
Shift-Register Mode 3-14
ROM Mode 3-15
FIFO Mode 3-15
Clocking Modes 3-16
Independent Clock Mode 3-16
Input/Output Clock Mode 3-16
Read/Write Clock Mode 3-16
Single Clock Mode 3-17
Design Considerations 3-17
Selecting TriMatrix Memory Blocks 3-17
Conflict Resolution 3-17
Read During Write 3-17
Same-Port Read-During-Write Mode 3-18
Mixed-Port Read-During-Write Mode 3-19
Power-Up Conditions and Memory Initialization 3-20
Power Management 3-20
Conclusion 3-20
Referenced Documents 3-21
Document Revision History 3-21

Chapter 4. DSP Blocks in Stratix IV Devices

Introduction 4-1
Stratix IV DSP Block Overview 4-2
Stratix IV Simplified DSP Operation 4-3
Stratix IV Operational Modes Overview 4-7
Stratix IV DSP Block Resource Descriptions 4-8
Input Registers 4-9
Multiplier and First-Stage Adder 4-11
Pipeline Register Stage 4-12
Second-Stage Adder 4-12
Round and Saturation Stage 4-13
Second Adder and Output Registers 4-13

Stratix IV Operational Mode Descriptions 4-14
Independent Multiplier Modes 4-14
9-, 12-, and 18-Bit Multiplier 4-14
36-Bit Multiplier 4-18
Double Multiplier 4-19
Two-Multiplier Adder Sum Mode 4-21
18 x 18 Complex Multiply 4-22
Four-Multiplier Adder 4-23
High-Precision Multiplier Adder Mode 4-25
Multiply Accumulate Mode 4-26
Shift Modes 4-27
Rounding and Saturation Mode 4-29
DSP Block Control Signals 4-31
Software Support 4-32
Conclusion 4-33
Referenced Documents 4-33
Document Revision History 4-33

Chapter 5. Clock Networks and PLLs in Stratix IV Devices

Introduction 5-1
Clock Networks in Stratix IV Devices 5-1
Global Clock Networks 5-2
Regional Clock Networks 5-3
Periphery Clock Networks 5-5
Clocking Regions 5-5
Clock Network Sources 5-7
Dedicated Clock Inputs Pins 5-7
Logic Array Blocks (LABs) 5-7
PLL Clock Outputs 5-7
Clock Input Connections to PLLs 5-9
Clock Output Connections 5-9
Clock Control Block 5-11
Clock Enable Signals 5-13
Clock Source Control for PLLs 5-14
Cascading PLLs 5-15

PLLs in Stratix IV Devices . . . 5-16
Stratix IV PLL Hardware Overview . . . 5-18
PLL Clock I/O Pins . . . 5-19
PLL Control Signals . . . 5-22
pfdena . . . 5-22
areset . . . 5-22
locked . . . 5-22
Clock Feedback Modes . . . 5-22
Source Synchronous Mode . . . 5-23
Source-Synchronous Mode for LVDS Compensation . . . 5-24
No-Compensation Mode . . . 5-24
Normal Mode . . . 5-25
Zero-Delay Buffer Mode . . . 5-26
External Feedback Mode . . . 5-27
Clock Multiplication and Division . . . 5-28
Post-Scale Counter Cascading . . . 5-29
Programmable Duty Cycle . . . 5-29
Programmable Phase Shift . . . 5-30
Programmable Bandwidth . . . 5-31
Background . . . 5-31
Implementation . . . 5-32
Spread-Spectrum Tracking . . . 5-33
Clock Switchover . . . 5-33
Automatic Clock Switchover . . . 5-34
Manual Clock Switchover . . . 5-36
Guidelines . . . 5-37
PLL Reconfiguration . . . 5-38
PLL Reconfiguration Hardware Implementation . . . 5-38
Post-Scale Counters (C0 to C9) . . . 5-41
Scan Chain Description . . . 5-41
Charge Pump and Loop Filter . . . 5-43
Bypassing PLL . . . 5-44
Dynamic Phase-Shifting . . . 5-45
PLL Specifications . . . 5-47
Conclusion . . . 5-47
Referenced Documents . . . 5-47
Document Revision History . . . 5-48

Section II. I/O Interfaces

Revision History . . . 5-1

Chapter 6. I/O Features in Stratix IV Devices

Introduction . . . 6-1
I/O Standards Support . . . 6-2
I/O Standards and Voltage Levels . . . 6-3
I/O Banks . . . 6-5
Modular I/O Banks . . . 6-7

I/O Structure 6-17
3.3-V I/O Interface 6-19
External Memory Interfaces 6-19
High-Speed Differential I/O with DPA Support 6-19
Current Strength 6-20
Slew Rate Control 6-21
I/O Delay 6-21
Programmable IOE Delay 6-22
Programmable Output Buffer Delay 6-22
Open-Drain Output 6-22
Bus Hold 6-22
Pull-Up Resistor 6-23
Pre-Emphasis 6-23
Differential Output Voltage 6-23
MultiVolt I/O Interface 6-23
On-Chip Termination Support and I/O Termination Schemes 6-24
On-Chip Series (R_S) Termination without Calibration 6-25
On-Chip Series Termination with Calibration 6-25
On-Chip Parallel Termination with Calibration 6-27
Dynamic On-Chip Termination 6-28
LVDS Input On-Chip Termination (R_D) 6-28
OCT Calibration 6-29
OCT Calibration Block Location 6-29
Sharing OCT Calibration Block on Multiple I/O Banks 6-31
OCT Calibration Block Modes of Operation 6-32
Power-Up Mode 6-32
User Mode 6-32
OCT Calibration 6-33
Serial Data Transfer 6-34
Example of Using Multiple OCT Calibration Blocks 6-34
RS Calibration 6-35
Termination Schemes for I/O Standards 6-35
Single-Ended I/O Standards Termination 6-35
Differential I/O Standards Termination 6-37
LVDS 6-39
Differential LVPECL 6-40
RSDS 6-40
Mini-LVDS 6-41
Design Considerations 6-42
I/O Bank Restrictions 6-42
Non-Voltage-Referenced Standards 6-43
Voltage-Referenced Standards 6-43
Mixing Voltage-Referenced and Non-Voltage-Referenced Standards 6-43
Conclusion 6-43
Referenced Documents 6-44
Document Revision History 6-44

Chapter 7. External Memory Interfaces in Stratix IV Devices

Introduction 7-1

Memory Interfaces Pin Support 7-4
Using R_{UP}/R_{DN} Pins in a DQS/DQ Group Used for Memory Interfaces 7-25
Combining ×16/×18 DQS/DQ groups for ×36 QDRII+/QDRII SRAM Interface 7-25
Rules to Combine Groups 7-26
Stratix IV External Memory Interface Features 7-27
DQS Phase-Shift Circuitry 7-27
DLL 7-29
Phase Offset Control 7-39
DQS Logic Block 7-40
DQS Delay Chain 7-41
Update Enable Circuitry 7-42
DQS Postamble Circuitry 7-42
Leveling Circuitry 7-44
Dynamic On-Chip Termination Control 7-46
I/O Element Registers 7-46
Conclusion 7-50
Referenced Documents 7-50
Document Revision History 7-51

Chapter 8. High-Speed Differential I/O Interfaces and DPA in Stratix IV Devices

Introduction 8-1
Locations of I/O Banks 8-3
Stratix IV LVDS Channels 8-5
Stratix IV LVDS SERDES Block Diagram 8-7
Stratix IV ALTLVDS Port List 8-8
Stratix IV Differential Transmitter 8-11
Programmable Pre-Emphasis and Programmable VOD 8-13
Programmable VOD 8-14
Programmable Pre-Emphasis 8-14
Stratix IV Differential Receiver 8-15
Differential I/O Termination 8-17
Receiver Hardware Blocks 8-17
Dynamic Phase Alignment (DPA) Block 8-17
Synchronizer 8-18
Data Realignment Block (Bit Slip) 8-19
Deserializer 8-20
Receiver Data Path Modes 8-20
Non-DPA Mode 8-21
DPA Mode 8-21
Soft-CDR Mode 8-22
Stratix IV LVDS Interface with 'Use External PLL' Option Enabled 8-24
Stratix IV Left/Right PLLs (PLL_Lx/ PLL_Rx) 8-26
Stratix IV Clocking 8-26
Stratix IV Source-Synchronous Timing Budget 8-27
Differential Data Orientation 8-28
Differential I/O Bit Position 8-28
Transmitter Channel to Channel Skew for Non-DPA Mode 8-30
Receiver Skew Margin for Non-DPA Mode 8-30

Differential Pin Placement Guidelines 8-34
Guidelines for DPA-Enabled Differential Channels 8-34
DPA-Enabled Channels and Single-Ended I/Os 8-34
DPA-Enabled Channel Driving Distance 8-34
Using Corner and Center Left/Right PLLs 8-35
Using Both Center Left/Right PLLs 8-36
Guidelines for DPA-Disabled Differential Channels 8-37
DPA-Disabled Channels and Single-Ended I/Os 8-37
DPA-Disabled Channel Driving Distance 8-38
Using Corner and Center Left/Right PLLs 8-38
Using Both Center Left/Right PLLs 8-39
Referenced Documents 8-40
Document Revision History 8-41

Section III. System Integration

Revision History 8-1

Chapter 9. Hot Socketing and Power-On Reset in Stratix IV Devices

Introduction 9-1
Stratix IV E Hot-Socketing Specifications 9-1
Stratix IV E Devices Can Be Driven Before Power Up 9-2
I/O Pins Remain Tri-Stated During Power Up 9-2
Insertion or Removal of a Stratix IV E Device from a Powered-Up System 9-2
Hot-Socketing Feature Implementation in Stratix IV E Devices 9-3
Power-On Reset Circuitry 9-4
Power-On Reset Specifications 9-5
Conclusion 9-5
Referenced Documents 9-5
Revision History 9-6

Chapter 10. Configuration, Design Security, and Remote System Upgrades in Stratix IV Devices

Introduction 10-1
Configuration Devices 10-1
Configuration Schemes 10-2
Configuration Features 10-3
Power-On Reset Circuit 10-4
VCCPGM Pins 10-5
VCCPD Pins 10-5
Fast Passive Parallel Configuration 10-5
FPP Configuration Using a MAX II Device as an External Host 10-6
FPP Configuration Timing 10-10
FPP Configuration Using a Microprocessor 10-13
Fast Active Serial Configuration (Serial Configuration Devices) 10-14
Estimating Active Serial Configuration Time 10-19
Programming Serial Configuration Devices 10-20
Passive Serial Configuration 10-22
PS Configuration Using a MAX II Device as an External Host 10-22
PS Configuration Timing 10-27
PS Configuration Using a Microprocessor 10-28
PS Configuration Using a Download Cable 10-28
JTAG Configuration 10-31
Jam STAPL 10-37
Device Configuration Pins 10-37

Configuration Data Decompression 10-43
Remote System Upgrades 10-45
Functional Description 10-46
Enabling Remote Update 10-47
Configuration Image Types 10-48
Remote System Upgrade Mode 10-49
Overview 10-49
Remote Update Mode 10-49
Dedicated Remote System Upgrade Circuitry 10-51
Remote System Upgrade Registers 10-52
Remote System Upgrade Control Register 10-53
Remote System Upgrade Status Register 10-54
Remote System Upgrade State Machine 10-55
User Watchdog Timer 10-55
Quartus II Software Support 10-56
ALTREMOTE_UPDATE Megafunction 10-56
Design Security 10-57
Stratix IV Security Protection 10-58
Security Against Copying 10-58
Security Against Reverse Engineering 10-58
Security Against Tampering 10-58
AES Decryption Block 10-58
Flexible Security Key Storage 10-59
Stratix IV Design Security Solution 10-59
Security Modes Available 10-60
Volatile Key 10-60
Non-Volatile Key 10-60
Non-Volatile Key with Tamper Protection Bit Set 10-61
No Key Operation 10-61
Supported Configuration Schemes 10-61
Conclusion 10-63
Referenced Documents 10-63

Chapter 11. SEU Mitigation in Stratix IV Devices

Introduction 11-1
Error Detection Fundamentals 11-1
Configuration Error Detection 11-2
User Mode Error Detection 11-2
Automated Single Event Upset Detection 11-5
Critical Error Detection 11-5
Error Detection Pin Description 11-5
CRC_ERROR Pin 11-6
CRITICAL ERROR Pin 11-6
Error Detection Block 11-6
Error Detection Registers 11-7
Error Detection Timing 11-9
Software Support 11-10
Recovering From CRC Errors 11-11
Conclusion 11-11
Referenced Documents 11-12
Document Revision History 11-12

Chapter 12. JTAG Boundary-Scan Testing in Stratix IV Devices

Introduction 12-1
BST Architecture 12-1
BST Operation Control 12-1
I/O Voltage Support in a JTAG Chain 12-3
BST Circuitry 12-3
Boundary-Scan Description Language (BSDL) Support 12-3
Conclusion 12-4
Referenced Documents 12-4
Revision History 12-4

Chapter 13. Power Management in Stratix IV Devices

Introduction 13-1
Stratix IV Power Technology 13-1
    Programmable Power Technology 13-2
Stratix IV External Power Supply Requirements 13-3
Temperature Sensing Diode 13-4
    External Pin Connections 13-4
Conclusion 13-5
Referenced Document 13-5
Revision History 13-5

Chapter Revision Dates

The chapters in this book, *Stratix IV Device Handbook, Volume 1*, were revised on the following dates. Where chapters or groups of chapters are available separately, part numbers are listed.

Chapter 1	Stratix IV Device Family Overview Revised: *November 2008* Part Number: *SIV51001-2.0*
Chapter 2	Logic Array Blocks and Adaptive Logic Modules in Stratix IV Devices Revised: *November 2008* Part Number: *SIV51001-2.0*
Chapter 3	TriMatrix Embedded Memory Blocks in Stratix IV Devices Revised: *November 2008* Part Number: *SIII51003-2.0*
Chapter 4	DSP Blocks in Stratix IV Devices Revised: *November 2008* Part Number: *SIV51004-2.0*
Chapter 5	Clock Networks and PLLs in Stratix IV Devices Revised: *November 2008* Part Number: *SIV51005-2.0*
Chapter 6	I/O Features in Stratix IV Devices Revised: *November 2008* Part Number: *SIV51006-2.0*
Chapter 7	External Memory Interfaces in Stratix IV Devices Revised: *November 2008* Part Number: *SIV51007-2.0*
Chapter 8	High-Speed Differential I/O Interfaces and DPA in Stratix IV Devices Revised: *November 2008* Part Number: *SIV51008-2.0*
Chapter 9	Hot Socketing and Power-On Reset in Stratix IV Devices Revised: *November 2008* Part Number: *SIV51009-2.0*
Chapter 10	Configuration, Design Security, and Remote System Upgrades in Stratix IV Devices Revised: *November 2008* Part Number: *SIV51010-2.0*
Chapter 11	SEU Mitigation in Stratix IV Devices Revised: *November 2008* Part Number: *SIV51011-2.0*

Chapter 12 JTAG Boundary-Scan Testing in Stratix IV Devices
Revised: *November 2008*
Part Number: *SIV51012-2.0*

Chapter 13 Power Management in Stratix IV Devices
Revised: *November 2008*

Chapter 13 Power Management in Stratix IV Devices
Revised: *November 2008*
Part Number: *SIV51013-2.0*

About this Handbook

This handbook provides comprehensive information about the Altera® Stratix® IV family of devices.

How to Contact Altera

For the most up-to-date information about Altera products, see the following table.

Contact *(Note 1)*	Contact Method	Address
Technical support	Website	www.altera.com/support
Technical training	Website	www.altera.com/training
	Email	custrain@altera.com
Altera literature services	Email	literature@altera.com
Non-technical support (General)	Email	nacomp@altera.com
(Software Licensing)	Email	authorization@altera.com

Note:

(1) You can also contact your local Altera sales office or sales representative.

Typographic Conventions

The following table shows the typographic conventions that this document uses.

Visual Cue	Meaning
Bold Type with Initial Capital Letters	Indicates command names, dialog box titles, dialog box options, and other GUI labels. For example, **Save As** dialog box. For GUI elements, capitalization matches the GUI.
bold type	Indicates directory names, project names, disk drive names, file names, file name extensions, dialog box options, software utility names, and other GUI labels. For example, **\qdesigns** directory, **d:** drive, and **chiptrip.gdf** file.
Italic Type with Initial Capital Letters	Indicates document titles. For example, *AN 519: Stratix IV Design Guidelines.*
Italic type	Indicates variables. For example, *n* + 1. Variable names are enclosed in angle brackets (< >). For example, <*file name*> and <*project name*>.**pof** file.
Initial Capital Letters	Indicates keyboard keys and menu names. For example, Delete key and the Options menu.
"Subheading Title"	Quotation marks indicate references to sections within a document and titles of Quartus II Help topics. For example, "Typographic Conventions."

Visual Cue	Meaning
`Courier type`	Indicates signal, port, register, bit, block, and primitive names. For example, `data1`, `tdi`, and `input`. Active-low signals are denoted by suffix `n`. For example, `resetn`. Indicates command line commands and anything that must be typed exactly as it appears. For example, `c:\qdesigns\tutorial\chiptrip.gdf`. Also indicates sections of an actual file, such as a Report File, references to parts of files (for example, the AHDL keyword `SUBDESIGN`), and logic function names (for example, `TRI`).
1., 2., 3., and a., b., c., and so on.	Numbered steps indicate a list of items when the sequence of the items is important, such as the steps listed in a procedure.
■ ■	Bullets indicate a list of items when the sequence of the items is not important.
☞	The hand points to information that requires special attention.
⚠ CAUTION	A caution calls attention to a condition or possible situation that can damage or destroy the product or your work.
⚠ WARNING	A warning calls attention to a condition or possible situation that can cause you injury.
↵	The angled arrow instructs you to press **Enter**.
👣	The feet direct you to more information about a particular topic.

Section I. Device Core

This section provides a complete overview of all features relating to the Stratix® IV device family, which is the most architecturally advanced, high-performance, low-power FPGA in the market place. This section includes the following chapters:

- Chapter 1, Stratix IV Device Family Overview
- Chapter 2, Logic Array Blocks and Adaptive Logic Modules in Stratix IV Devices
- Chapter 3, TriMatrix Embedded Memory Blocks in Stratix IV Devices
- Chapter 4, DSP Blocks in Stratix IV Devices
- Chapter 5, Clock Networks and PLLs in Stratix IV Devices

Revision History

Refer to each chapter for its own specific revision history. For information on when each chapter was updated, refer to the Chapter Revision Dates section, which appears in the full handbook.

1. Stratix IV Device Family Overview

SIV51001-2.0

Introduction

Altera® Stratix® IV FPGAs deliver a breakthrough level of system bandwidth and power efficiency for high-end applications, allowing designers to innovate without compromise. Stratix IV FPGAs are based on Taiwan Semiconductor Manufacturing Company's (TSMC's) 40-nm process technology and surpass all other high-end FPGAs available today, with the highest logic density, most transceivers, and lowest power.

The Stratix IV device family contains two variants optimized to meet different application needs:

- Stratix IV GX transceiver FPGAs—up to 531,200 logic elements (LEs) and 48 full-duplex clock data recovery (CDR)-based transceivers at up to 8.5 Gbps
- Stratix IV E (Enhanced) FPGAs—up to 681,100 LEs, 31,491 Kbits RAM, 1,360 18 × 18-bit multipliers

The complete Altera high-end solution includes the lowest risk, lowest total cost path to volume using HardCopy® IV ASICs for both family variants, a comprehensive portfolio of application solutions customized for end-markets, and the industry leading Quartus® II software for increasing productivity and performance.

This chapter contains the following sections:

- "Feature Summary" on page 1–1
- "Architecture Features" on page 1–4
- "Integrated Software Platform" on page 1–12

Feature Summary

The following list summarizes the features in the Stratix IV device families:

- Up to 32 full-duplex CDR-based transceivers supporting data rates between 600 Mbps and 8.5 Gbps. Up to 16 additional PMA-Only transceivers supporting data rates between 600 Mbps and 3.2 Gbps.

  For more information about the additional PMA-Only channels, refer to the "*Configuring Clock Multiplier Unit (CMU) Channels as Additional Transceiver Channels*" section in the *Upcoming Stratix IV Device Features* document.

- Dedicated circuitry to support physical layer functionality for popular serial protocols, such as PCI Express (PIPE) Gen1 and Gen2, Gigabit Ethernet, Serial RapidIO, SONET/SDH, XAUI/HiGig, (OIF) CEI-6G, SD/HD/3G-SDI, Fibre Channel, SFI-5, and Interlaken

- Complete PCI Express (PIPE) protocol solution with embedded PCI Express hard IP blocks that implement PHY-MAC layer, Data Link layer, and Transaction layer functionality

  Refer to the *PCI Express Compiler User Guide* for more information

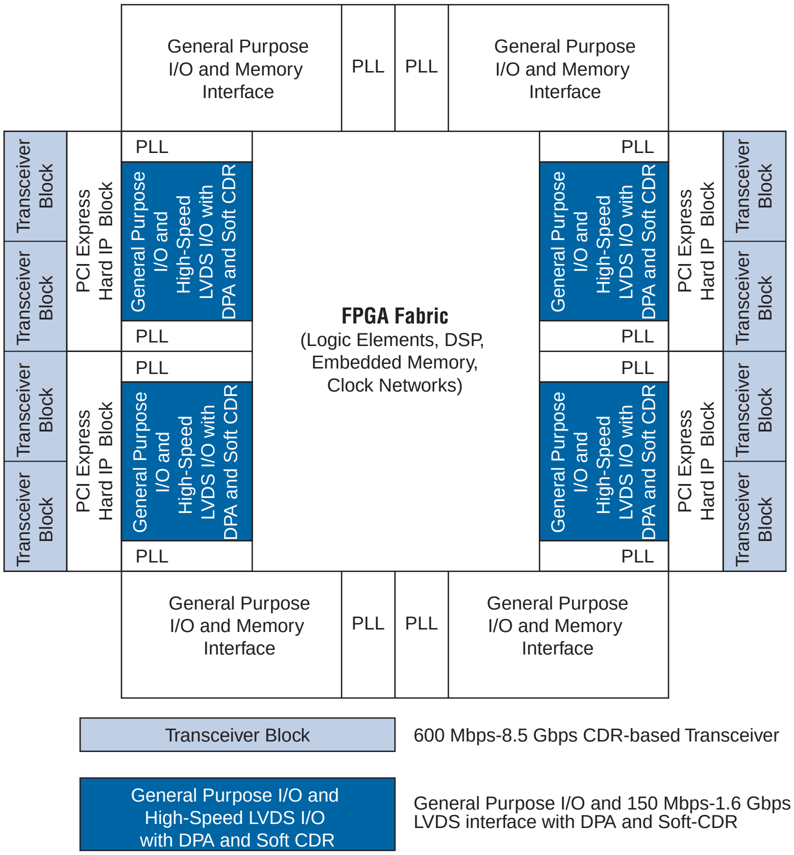
- Programmable transmitter pre-emphasis and receiver equalization circuitry to compensate for frequency-dependent losses in the physical medium
- Typical physical medium attachment (PMA) power consumption of 100 mW at 3.125 Gbps and 135 mW at 6.375 Gbps per channel
- 72,600 to 681,100 equivalent LEs per device
- 7,370 to 31,491 Kbits of enhanced TriMatrix memory consisting of three RAM block sizes to implement true dual-port memory and FIFO buffers
- High-speed DSP blocks configurable as 9 × 9-bit, 12 × 12-bit, 18 × 18-bit, and 36 × 36-bit full-precision multipliers at up to 540 MHz
- Up to 16 global clocks (GCLK), 88 regional clocks (RCLK), and 132 periphery clocks (PCLK) per device
- Programmable Power Technology that minimizes power while maximizing device performance
- Up to 1,104 user I/O pins arranged in 24 modular I/O banks that support a wide range of single-ended and differential I/O standards
- Support for high-speed external memory interfaces including DDR, DDR2, DDR3 SDRAM, RLDRAM II, QDR II, and QDR II+ SRAM on up to 24 modular I/O banks
- High-speed LVDS I/O support with serializer/deserializer (SERDES), dynamic phase alignment (DPA) and soft-CDR circuitry at data rates up to 1.6 Gbps
- Support for source-synchronous bus standards, including SGMII, Gigabit Ethernet, SPI-4 Phase 2 (POS-PHY Level 4), SFI-4.1, XSBI, UTOPIA IV, NPSI, and CSIX-L1
- Pinouts for Stratix IV E devices designed to allow migration of designs from Stratix III to Stratix IV E with minimal printed circuit board impact

Stratix IV GX Devices

Stratix IV GX devices provide up to 32 transceiver channels per device with physical coding sublayer (PCS) and physical medium attachment (PMA) support at data rates between 600 Mbps and 8.5 Gbps. Up to 16 additional channels with PMA-only support at data rates between 600 Mbps and 3.2 Gbps are also available, for a total of up to 48 transceiver channels per device.

Figure 1–1 shows a high-level Stratix IV GX chip view.

Figure 1–1. Stratix IV GX Chip View *(Note 1)*


General Purpose I/O and Memory Interface
PLL
PLL
General Purpose I/O and Memory Interface
Transceiver Block
Transceiver Block
Transceiver Block
Transceiver Block
PCI Express Hard IP Block
PCI Express Hard IP Block
PLL
General Purpose I/O and High-Speed LVDS I/O with DPA and Soft CDR
PLL
PLL
General Purpose I/O and High-Speed LVDS I/O with DPA and Soft CDR
PLL
FPGA Fabric
(Logic Elements, DSP, Embedded Memory, Clock Networks)
PLL
General Purpose I/O and High-Speed LVDS I/O with DPA and Soft CDR
PLL
PLL
General Purpose I/O and High-Speed LVDS I/O with DPA and Soft CDR
PLL
PCI Express Hard IP Block
PCI Express Hard IP Block
Transceiver Block
Transceiver Block
Transceiver Block
Transceiver Block
General Purpose I/O and Memory Interface
PLL
PLL
General Purpose I/O and Memory Interface
Transceiver Block
600 Mbps-8.5 Gbps CDR-based Transceiver
General Purpose I/O and High-Speed LVDS I/O with DPA and Soft CDR
General Purpose I/O and 150 Mbps-1.6 Gbps LVDS interface with DPA and Soft-CDR

Note to Figure 1–1:

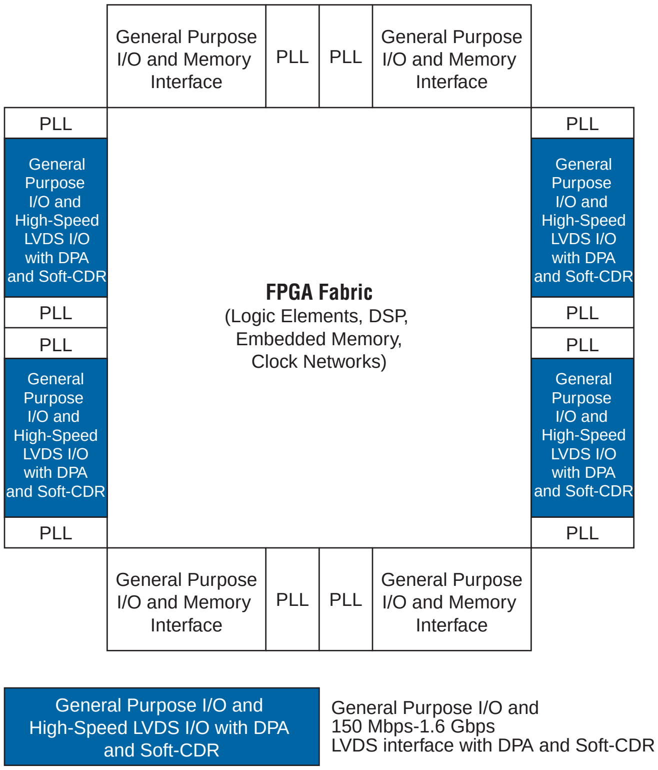
(1) Resource counts vary with device selection, package selection, or both.

Stratix IV E Device Family

Stratix IV E devices provide an excellent solution for applications that do not require high-speed CDR-based transceivers, but are logic, user I/O, or memory intensive.

Figure 1–2 shows a high-level Stratix IV E chip view.

Figure 1–2. Stratix IV E Chip View *(Note 1)*


General Purpose I/O and Memory Interface
PLL
PLL
General Purpose I/O and Memory Interface
PLL
General Purpose I/O and High-Speed LVDS I/O with DPA and Soft-CDR
PLL
PLL
General Purpose I/O and High-Speed LVDS I/O with DPA and Soft-CDR
PLL
FPGA Fabric
(Logic Elements, DSP, Embedded Memory, Clock Networks)
PLL
General Purpose I/O and High-Speed LVDS I/O with DPA and Soft-CDR
PLL
PLL
General Purpose I/O and High-Speed LVDS I/O with DPA and Soft-CDR
PLL
General Purpose I/O and Memory Interface
PLL
PLL
General Purpose I/O and Memory Interface
General Purpose I/O and High-Speed LVDS I/O with DPA and Soft-CDR
General Purpose I/O and 150 Mbps-1.6 Gbps LVDS interface with DPA and Soft-CDR

Note to Figure 1–2:

(1) Resource counts vary with device selection, package selection, or both.

Architecture Features

As discussed in this section, Stratix IV device family features are divided into high-speed transceiver features, and FPGA fabric and I/O features.

☞ The high-speed transceiver features apply only to the Stratix IV GX family variant.

High-Speed Transceiver Features

Stratix IV GX high-speed transceiver features include:

Highest Aggregate Data Bandwidth

- Up to 32 full-duplex transceiver channels with PCS and PMA support at data rates between 600 Mbps and 8.5 Gbps
- Up to 16 additional full-duplex transceiver channels with PMA-only support at data rates between 600 Mbps and 3.2 Gbps

Wide Range of Protocol Support

- Physical layer support for the following serial protocols: PCI Express (PIPE) Gen1 and Gen2, Gigabit Ethernet, Serial RapidIO, SONET/SDH, XAUI/HiGig, (OIF) CEI-6G, SD/HD/3G-SDI, Fibre Channel, SFI-5, and Interlaken

- Extremely flexible and easy-to-configure transceiver data path to implement proprietary protocols
- PCI Express (PIPE) Support
    - Complete PCI Express (PIPE) Gen1 and Gen2 protocol stack solution compliant to PCI Express Base Specification 2.0 that includes PHY-MAC, Data Link, and Transaction layer circuitry embedded in PCI Express hard IP blocks

    Refer to the *PCI Express Compiler User Guide* for more information.

    - Root complex and end point applications
    - ×1, ×4, and ×8 lane configurations
    - PIPE2.0-compliant interface
    - Embedded circuitry to switch between Gen1 and Gen2 data rates
    - Built-in circuitry for electrical idle generation and detection, receiver detect, power state transitions, lane reversal, and polarity inversion
    - 8B/10B encoder and decoder, receiver synchronization state machine, and ± 300 parts per million (ppm) clock compensation circuitry
    - Transaction layer support for up to two virtual channels (VCs)
- XAUI/HiGig Support
    - Compliant to IEEEP802.3ae specification
    - Embedded state machine circuitry to convert XGMII idle code groups (||I||) to and from idle ordered sets (||A||, ||K||, ||R||) at the transmitter and receiver, respectively
    - 8B/10B encoder and decoder, receiver synchronization state machine, lane deskew, and ± 100 ppm clock compensation circuitry
- Gigabit Ethernet Support
    - Compliant to IEEE802.3-2005 specification
    - Automatic idle ordered set (/I1/, /I2/) generation at the transmitter, depending on the current running disparity
    - 8B/10B encoder and decoder, receiver synchronization state machine, and ± 100 ppm clock compensation circuitry
- Support for other protocol features such as MSB to LSB transmission in SONET/SDH configuration and spread-spectrum clocking in PCI Express (PIPE) configurations

Stratix IV Device Diagnostic Features

- Serial loopback from transmitter serializer to receiver CDR for transceiver PCS and PMA diagnostics
- Reverse serial loopback pre- and post-CDR to transmitter buffer for physical link diagnostics
- Loopback master and slave capability in PCI Express hard IP blocks

For more information refer to the *PCI Express Compiler User Guide*.

Signal Integrity

Stratix IV devices simplify the challenge of signal integrity through a number of chip, package, and board-level enhancements to enable efficient high-speed data transfer into and out of the device. These enhancements include:

- Programmable 3-tap transmitter pre-emphasis with up to 8192 pre-emphasis levels to compensate for pre-cursor and post-cursor inter-symbol interference (ISI)
- Up to 900% boost capability on the first pre-emphasis post-tap
- User-controlled and adaptive 4-stage receiver equalization with up to 16 dB of high-frequency gain
- On-die power supply regulators for transmitter and receiver phase-locked loop (PLL) charge pump and voltage controlled oscillator (VCO) for superior noise immunity
- On-package and on-chip power supply decoupling to satisfy transient current requirements at higher frequencies, thereby reducing the need for on-board decoupling capacitors
- Calibration circuitry for transmitter and receiver on-chip termination (OCT) resistors

For more information about SI support in the Quartus II software, refer to the *Quartus II Handbook*.

FPGA Fabric and I/O Features

The Stratix IV FPGA fabric and I/O features include:

Device Core Features

- Up to 531,200 LEs in Stratix IV GX devices and up to 681,100 LEs in Stratix IV E devices, efficiently packed in unique and innovative adaptive logic modules (ALMs)
- 10 ALMs per logic array block (LAB) deliver faster performance, improved logic utilization, and optimized routing
- Advanced power saving techniques, including a variety of process, circuit, and architecture optimizations and innovations
- Programmable Power Technology available to select power-driven compilation options for reduced static power consumption

Embedded Memory

- TriMatrix embedded memory architecture provides three different memory block sizes to efficiently address the needs of diversified FPGA designs: 640-bit MLAB, 9-Kbit M9K, and 144-Kbit M144K
- Up to 31,491 Kbit of embedded memory operating at up to 600 MHz
- Each memory block is independently configurable to be a single- or dual-port RAM, FIFO, ROM, or shift register

Digital Signal Processing (DSP) Blocks

- Flexible DSP blocks configurable as 9 × 9-bit, 12 × 12-bit, 18 × 18-bit, and 36 × 36-bit full-precision multipliers at up to 540 MHz with rounding and saturation capabilities
- Faster operation due to fully pipelined architecture and built-in addition, subtraction, and accumulation units to combine multiplication results
- Optimally designed to support advanced features such as adaptive filtering, barrel shifters, and finite and infinite impulse response (FIR and IIR) filters

Clock Networks

- Up to 16 GCLKs and 88 RCLKs optimally routed to meet the internal logic f_{MAX} frequency of 600 MHz
- Up to 112 and 132 additional PCLKs in Stratix IV GX and Stratix IV E devices, respectively
- Up to 66 (16 GCLK + 22 RCLK + 28 PCLK) and 71 (16 GCLK + 22 RCLK + 33 PCLK) unique clock resources per device quadrant in Stratix IV GX and Stratix IV E devices, respectively

PLLs

- Three to 12 PLLs per device supporting spread-spectrum input clocking, programmable bandwidth, clock switch over, dynamic reconfiguration, and delay compensation
- Additional PLLs available from unused transceiver blocks (Stratix IV GX devices only)
- On-chip PLL power supply regulators to minimize noise coupling

I/O Features

- Sixteen to 24 modular I/O banks per device with 24 to 48 I/Os per bank designed and packaged for optimal SSN performance and migration capability
- Support for a wide range of industry I/O standards, including single-ended (LVTTL/CMOS/PCI/PCIX), differential (LVDS/mini-LVDS/RSDS), voltage-referenced single-ended and differential (SSTL/HSTL Class I/II) I/O standards
- On-chip series (R_S) and on-chip parallel (R_T) termination with auto-calibration for single-ended I/Os and on-chip differential (R_D) termination for differential I/Os
- Programmable output drive strength, slew rate control, bus hold, and weak pull-up capability for single-ended I/Os
- User I/O /GND/V_{CC} ratio of 8:1:1 to reduce loop inductance in the package
- Programmable transmitter differential output voltage (V_{OD}) and pre-emphasis for high-speed LVDS I/O

High-Speed Differential I/O with DPA and Soft-CDR

- Dedicated circuitry on the left and right sides of the device to support differential links at data rates from 150 Mbps to 1.6 Gbps
- Up to 98 and 132 differential SERDES in Stratix IV GX and Stratix IV E devices, respectively
- DPA circuitry at the receiver automatically compensates for channel-to-channel and channel-to-clock skew in source synchronous interfaces
- Soft-CDR circuitry at the receiver allows implementation of asynchronous serial interfaces with embedded clock at up to 1.6 Gbps data rate (SGMII and Gigabit Ethernet)

External Memory Interfaces

- Support for existing and emerging memory interface standards such as DDR SDRAM, DDR2 SDRAM, DDR3 SDRAM, QDRII SRAM, QDRII+ SRAM, and RLDRAM II
- DDR3 rates up to 1,067 Mbps/533 MHz
- Programmable DQ group widths of 4 to 36 bits (includes parity bits)
- Dynamic on-chip termination, trace mismatch compensation, read-write leveling, and half-rate register capabilities provide a robust external memory interface solution

System Integration

- Stratix IV E devices support hot socketing
- Four different configuration modes: Passive Serial, Fast Passive Parallel, Fast Active Serial, and JTAG
- Ability to perform remote system upgrades
- 256-bit advanced encryption standard (AES) encryption of configuration bits protects your design against copying, reverse engineering, and tampering.
- Built-in soft error detection for configuration RAM cells

Table 1–1 shows the Stratix IV GX device features.

Table 1–1. Stratix IV GX Device Features (Part 1 of 2)

Feature	EP4SGX70	EP4SGX110		EP4SGX230			EP4SGX290				EP4SGX360				EP4SGX530		
Package Option	F780	F780	F1152	F780	F1152	F1517	F780	F1152	F1517	F1932	F780	F1152	F1517	F1932	F1152	F1517	F1932
ALMs	29,040	42,240		91,200			116,480				141,440				212,480		
LEs	72,600	105,600		228,000			291,200				353,600				531,200		
0.6 Gbps-8.5 Gbps Transceivers (2)	8	8	16	8	16	24	16	16	24	24	16	16	24	24	16	24	32
0.6 Gbps-3.2 Gbps Transceivers (PMA-Only) (3)	0	0	0	0	0, 8	12	0	0, 8	12	12	0	0, 8	12	12	8	12	16
PCI Express Hard IP Blocks	1	1	2	1	2		2			4	2			4	2		4
High-Speed LVDS SERDES (150 Mbps - 1.6 Gbps)	28	28	28	28	44	88	0	44	88	88	0	44	88	88	44	88	98
SPI-4.2 Links	1	1		1	2	4	0	2	4	4	0	2	4		2	4	
M9K Blocks (256×36 bits)	462	660		1,235			936				1,248				1,280		
M144K Blocks (2048×72 bits)	16	16		22			36				48				64		
Total Memory (MLAB+M9K+M144K) Kbits	7,370	9,564		17,133			17,248				22,564				27,376		
Embedded Multipliers 18×18	384	512		1,288			832				1,040				1,024		
PLLs	3	3	4	3	6	8	4	6	8	12	4	6	8	12	6	8	12

Table 1–1. Stratix IV GX Device Features (Part 2 of 2)

Feature	EP4SGX70	EP4SGX110		EP4SGX230			EP4SGX290				EP4SGX360				EP4SGX530		
User I/Os *(1)*	368	368	368	368	560	736	288	560	736	864	288	560	736	864	560	736	904

Notes to Table 1–1:

(1) The total number of user I/Os includes the high-speed LVDS I/Os.

(2) The total number of transceivers is divided equally between the left and right side of each device, except for the devices in the F780 package. These devices have eight transceiver channels located only on the right side of the device.

(3) For more information about PMA-only channels, refer to the "*Configuring Clock Multiplier Unit (CMU) Channels as Additional Transceiver Channels*" section in the *Upcoming Stratix IV Device Features* document.

Table 1–2 shows the Stratix IV GX device package options.

Table 1–2. Stratix IV GX Device Package Options and I/O Pin and Transceiver Counts *(2)*, *(6)*

Device	User I/Os, Transceiver Count *(1)*				
	F780 (29mm×29mm)	**F1152 (35mm×35mm) *(3)***	**F1152 (35mm×35mm) *(3)*, *(7)***	**F1517 (40mm×40mm) *(7)***	**F1932 (45mm×45mm)**
EP4SGX70	▲ 368, 8	—	—	—	—
EP4SGX110	368, 8	▲ 368, 16	—	—	—
EP4SGX230	▼ 368, 8	560, 16	▲ 560, 24	▲ 736, 36	—
EP4SGX290	▲ 288, 16 *(4)*	560, 16	560, 24	736, 36	—
EP4SGX360	▼ 288, 16 *(4)*	▼ 560, 16	560, 24	736, 36	—
EP4SGX530	—	—	▼ 560, 24 *(5)*	▼ 736, 36 *(5)*	904, 48

Notes to Table 1–2:

(1) For each device package, the first number indicates the number of I/Os; the second number indicates the transceiver count.

(2) Device packages in the same column and marked under the same arrow sign have vertical migration capability.

(3) EP4SGX230, EP4SGX290, and EP4SGX360 devices have two variants in the F1152 package option—one with no PMA-only transceiver channels and the other with eight PMA-only transceiver channels. Vertical migration capability is available for devices of the same variant only.

(4) The 780-pin EP4SGX290 and EP4SGX360 devices are available only in 33mm×33mm Hybrid flip-chip package.

(5) The 1152-pin and 1517-pin EP4SGX530 devices are available only in 42.5mm×42.5mm Hybrid flip-chip packages.

(6) I/O counts do not include dedicated clock inputs that can be used as data inputs.

(7) When migrating between hybrid and flip chip packages, there is an additional keep-out area. For more information, refer to the Altera Device Package Information Data Sheet.

Table 1–3 shows the Stratix IV E device features.

Table 1–3. Stratix IV E Device Features

Feature	EP4SE110	EP4SE230	EP4SE290			EP4SE360			EP4SE530			EP4SE680		
Package Option	F780	F780	F780	F1152	F1517	F780	F1152	F1517	F1152	F1517	F1760	F1152	F1517	F1760
ALMs	42,240	91,200	116,480			141,440			212,480			272,440		
LEs	105,600	228,000	291,200			353,600			531,200			681,100		
High-Speed LVDS SERDES (150Mbps-1.6Gbps)	56	56	56	88	88	56	88	88	88	112	112	88	112	132
SPI-4.2 Links	3	3	3	4	4	3	4		4	6		4	6	7
M9K Blocks (256×36 bits)	660	1,235	936			1,248			1,280			1,529		
M144K Blocks (2048×72 bits)	16	22	36			48			64			64		
Total Memory (MLAB+M9K+M144K) Kbits	9,564	17,133	17,248			22,564			27,376			31,491		
Embedded Multipliers (18×18)	512	1,288	832			1,040			1,024			1,360		
PLLs	4	4	4	8	12	4	8	12	8	12	12	8	12	12
User I/Os *(1)*	480	480	480	736	864	480	736	864	736	960	960	736	960	1,104

Note to Table 1–3:

(1) The total number of user I/Os includes the high-speed LVDS I/Os.

Table 1–4 summarizes the Stratix IV E device package options.

Table 1–4. Stratix IV E Device Package Plan *(Note 1)*, *(5)*

Device	User I/Os			
	F780 (29mm×29mm) *(4)*	**F1152 (35mm×35mm) *(4)***	**F1517 (40mm×40mm)**	**F1760 (42.5mm×42.5mm)**
EP4SE110	▲ 480	—	—	—
EP4SE230	480	—	—	—
EP4SE290	480 *(2)*	▲ 736	▲ 864	—
EP4SE360	▼ 480 *(2)*	736	864	—
EP4SE530	—	736 *(3)*	960 *(3)*	▲ 960
EP4SE680	—	▼ 736 *(6)*	▼ 960	▼ 1,104

Notes to Table 1–4:

(1) Device packages in the same column and marked under the same arrow sign have vertical migration capability.

(2) The 780-pin EP4SE290 and EP4SE360 devices are available only in 33mm×33mm Hybrid flip-chip package.

(3) The 1152-pin and 1517-pin EP4SE530 devices are available only in 42.5mm×42.5mm Hybrid flip-chip package.

(4) When migrating between hybrid and flip chip packages, there is an additional keep-out area. For more information, refer to the Altera Device Package Information Data Sheet.

(5) I/O counts do not include dedicated clock inputs that can be used as data inputs.

(6) The 1152-pin EP4SE680 device is available only in a 40mm x 40mm Hybrid flip-chip package.

Integrated Software Platform

The Quartus II software provides an integrated environment for HDL and schematic design entry, compilation and logic synthesis, full simulation and advanced timing analysis, SignalTap II logic analyzer, and device configuration of Stratix IV designs. The Quartus II software provides the MegaWizard® Plug-In Manager user interface to generate different functional blocks, such as memory, PLL, and digital signal processing logic. For transceivers, the Quartus II software provides the ALTGX MegaWizard Plug-In Manager interface that guides you through configuration of the transceiver based on the application requirements.

Altera's state-of-the-art Stratix IV GX transceiver, with its fully reconfigurable hardware, optimal signal integrity, and integrated Quartus II software platform allows you to implement low-power and reliable high-speed serial interface applications.

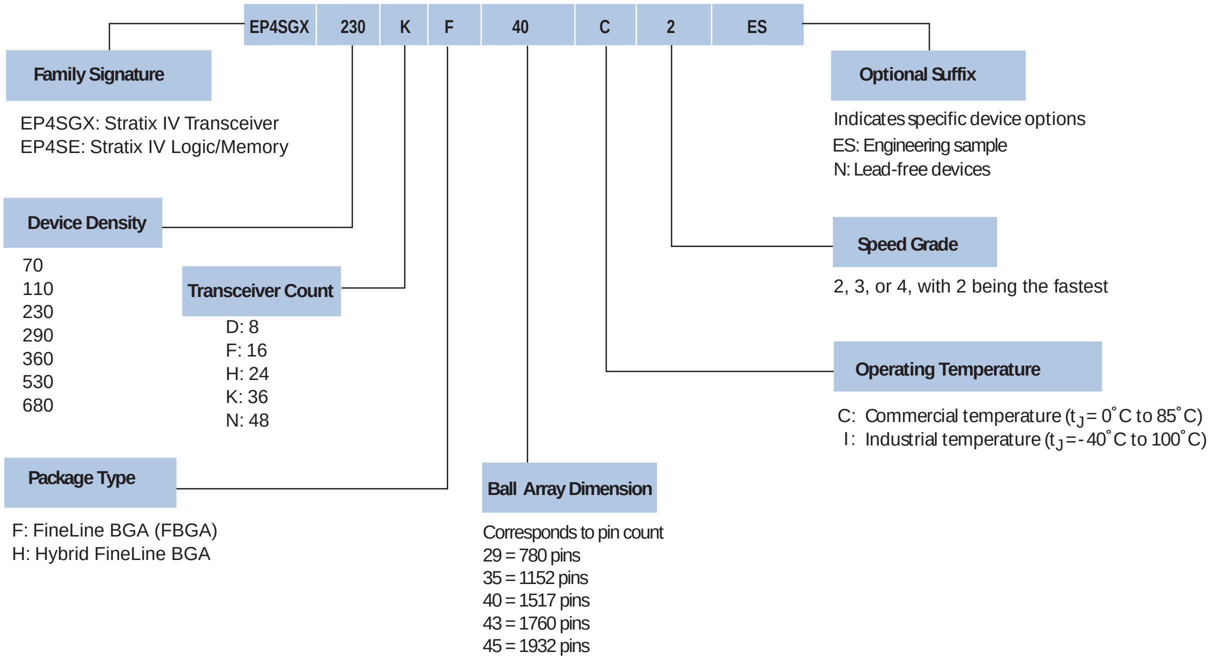
Refer to the *Quartus II Handbook* for more information about the Quartus II software features.

The Quartus II software supports the Windows XP/2000/NT/98, Sun Solaris, Linux Red Hat v7.1, and HP-UX operating systems. It also supports seamless integration with industry-leading EDA tools through the NativeLink interface.

Ordering Information

This section describes the Stratix IV ordering information. Figure 1–3 describes the ordering codes for Stratix IV devices.

Figure 1–3. Stratix IV Device Packaging Ordering Information



Referenced Documents

This chapter references the following documents

- *PCI Express Compiler User Guide*
- *Quartus II Handbook*
- *Upcoming Stratix IV Device Features*

Document Revision History

Table 1–5 shows the revision history for this document.

Table 1–5. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	■ Updated "Feature Summary" on page 1–1 ■ Updated "Stratix IV Device Diagnostic Features" on page 1–5 ■ Updated "FPGA Fabric and I/O Features" on page 1–6 ■ Updated Table 1–1 ■ Updated Table 1–2 ■ Updated "Integrated Software Platform" on page 1–12	—
July 2008 v1.1	Revised "Introduction"	—
May 2008 v1.0	Initial Release.	—

2. Logic Array Blocks and Adaptive Logic Modules in Stratix IV Devices

SIV51002-2.0

Introduction

This chapter describes the features of the logic array block (LAB) in the Stratix® IV core fabric. The logic array block is composed of basic building blocks known as adaptive logic modules (ALMs) that can be configured to implement logic functions, arithmetic functions, and register functions.

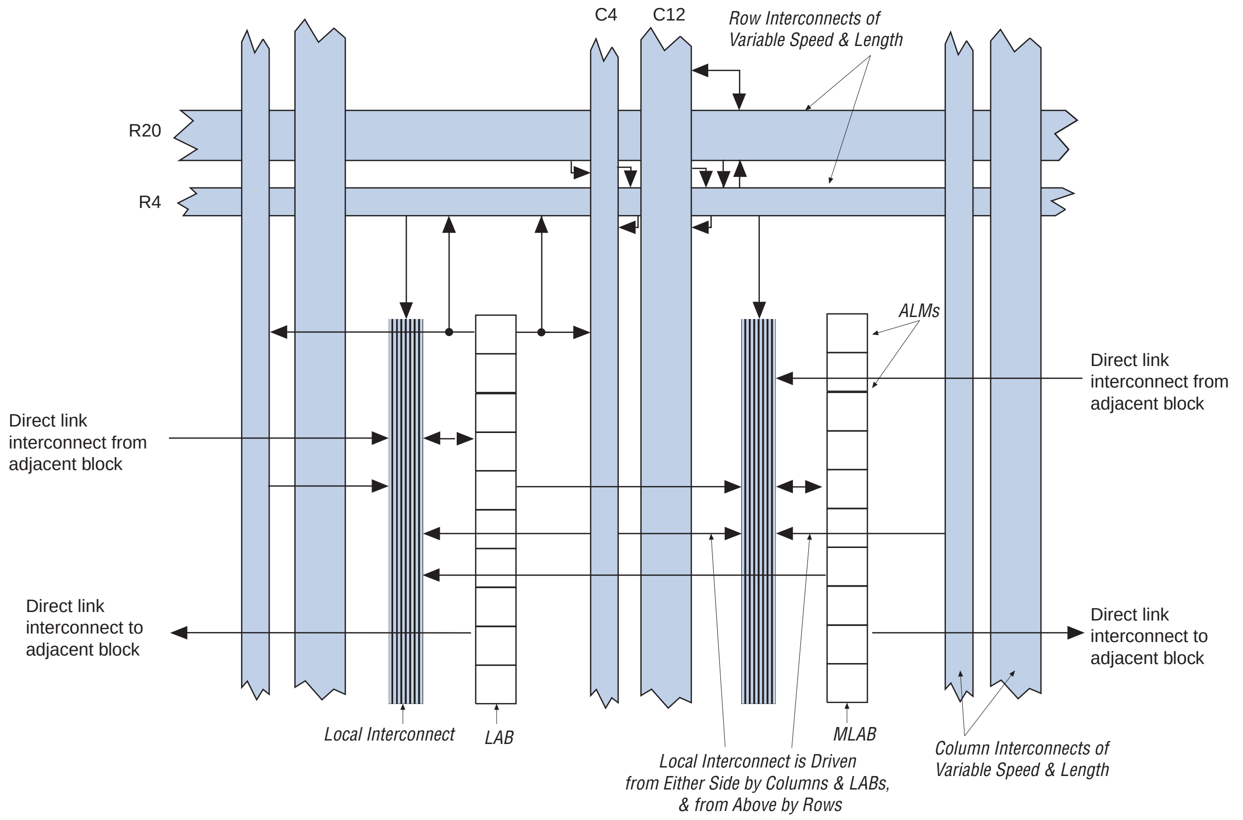
This chapter contains the following sections:

- "Logic Array Blocks" on page 2–1
- "Adaptive Logic Modules" on page 2–6

Logic Array Blocks

Each LAB consists of ten ALMs, various carry chains, shared arithmetic chains, LAB control signals, local interconnect, and register chain connection lines. The local interconnect transfers signals between ALMs in the same LAB. The direct link interconnect allows an LAB to drive into the local interconnect of its left and right neighbors. Register chain connections transfer the output of the ALM register to the adjacent ALM register in an LAB. The Quartus® II Compiler places associated logic in an LAB or adjacent LABs, allowing the use of local, shared arithmetic chain, and register chain connections for performance and area efficiency. Figure 2–1 shows the Stratix IV LAB structure and the LAB interconnects.

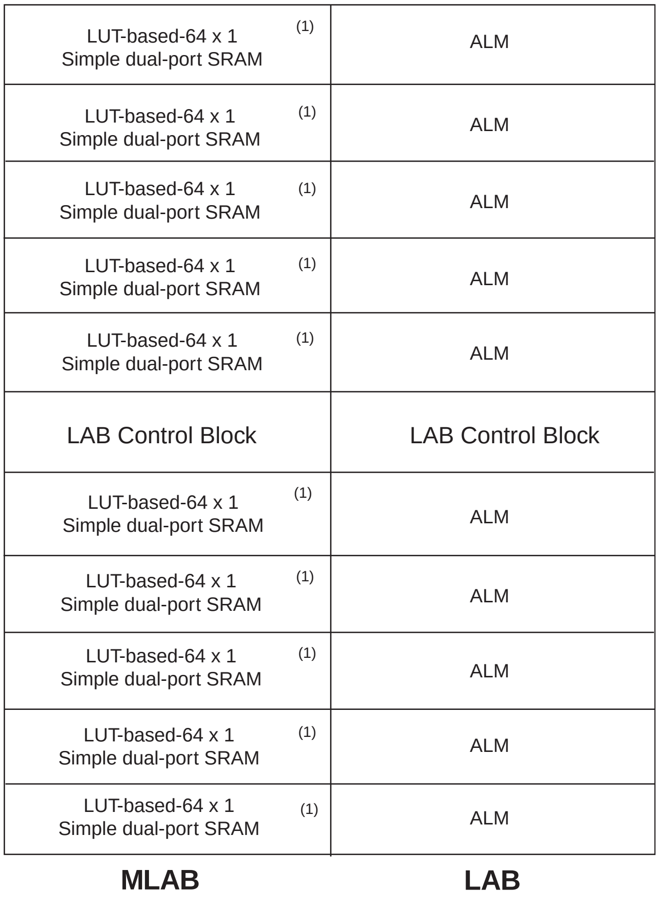
Figure 2–1. Stratix IV LAB Structure


C4
C12
Row Interconnects of Variable Speed & Length
R20
R4
ALMs
Direct link interconnect from adjacent block
Direct link interconnect from adjacent block
Direct link interconnect to adjacent block
Direct link interconnect to adjacent block
Local Interconnect
LAB
MLAB
Local Interconnect is Driven from Either Side by Columns & LABs, & from Above by Rows
Column Interconnects of Variable Speed & Length

The LAB of Stratix IV has a new derivative called Memory LAB (MLAB), which adds look-up table (LUT)-based SRAM capability to the LAB as shown in Figure 2–2. The MLAB supports a maximum of 640 bits of simple dual-port static random access memory (SRAM). You can configure each ALM in an MLAB as either a 64 × 1 or a 32 × 2 block, resulting in a configuration of either a 64 × 10 or a 32 × 20 simple dual-port SRAM block. MLAB and LAB blocks always coexist as pairs in all Stratix IV families. MLAB is a superset of the LAB and includes all LAB features. Figure 2–2 shows an overview of LAB and MLAB topology.

The MLAB is described in detail in the *TriMatrix Embedded Memory Blocks in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook*.

Figure 2–2. Stratix IV LAB and MLAB Structure


LUT-based-64 x 1 Simple dual-port SRAM (1)
ALM
LUT-based-64 x 1 Simple dual-port SRAM (1)
ALM
LUT-based-64 x 1 Simple dual-port SRAM (1)
ALM
LUT-based-64 x 1 Simple dual-port SRAM (1)
ALM
LUT-based-64 x 1 Simple dual-port SRAM (1)
ALM
LAB Control Block
LAB Control Block
LUT-based-64 x 1 Simple dual-port SRAM (1)
ALM
LUT-based-64 x 1 Simple dual-port SRAM (1)
ALM
LUT-based-64 x 1 Simple dual-port SRAM (1)
ALM
LUT-based-64 x 1 Simple dual-port SRAM (1)
ALM
LUT-based-64 x 1 Simple dual-port SRAM (1)
ALM
MLAB
LAB

Note to Figure 2–2:

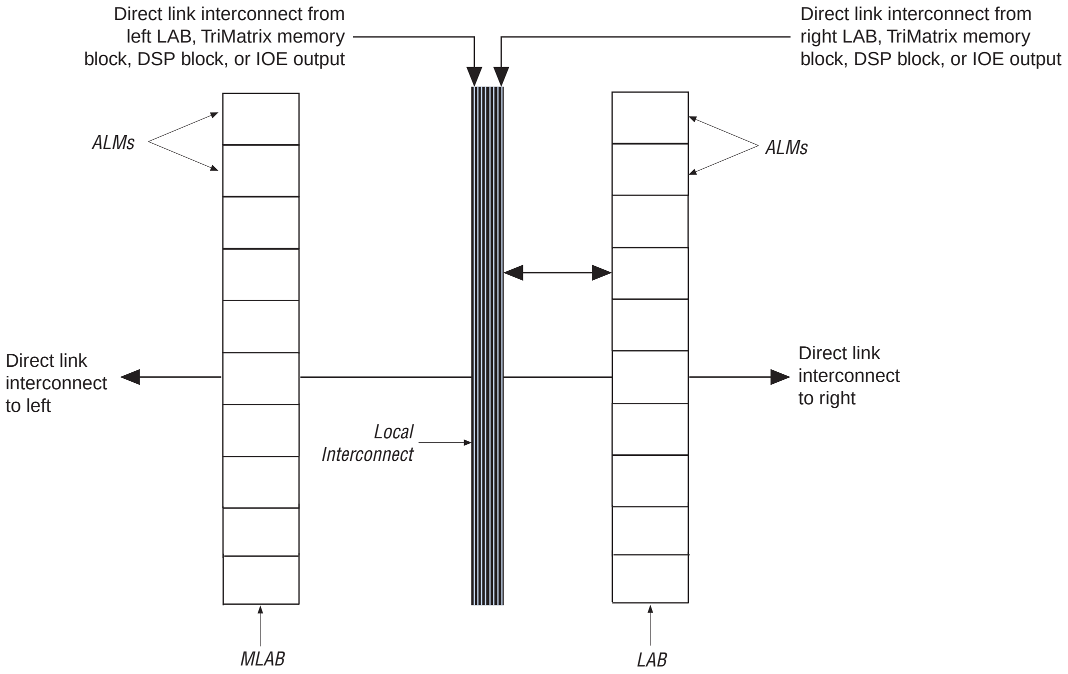
(1) You can use MLAB ALM as a regular LAB ALM or configure it as a dual-port SRAM, as shown.

LAB Interconnects

The LAB local interconnect can drive ALMs in the same LAB. It is driven by column and row interconnects and ALM outputs in the same LAB. Neighboring LABs/MLABs, M9K RAM blocks, M144K blocks, or DSP blocks from the left or right can also drive an LAB’s local interconnect through the direct link connection. The direct link connection feature minimizes the use of row and column interconnects, providing higher performance and flexibility. Each LAB can drive 30 ALMs through fast local and direct link interconnects.

Figure 2–3 shows the direct link connection.

Figure 2–3. Direct Link Connection


Direct link interconnect from left LAB, TriMatrix memory block, DSP block, or IOE output
Direct link interconnect from right LAB, TriMatrix memory block, DSP block, or IOE output
ALMs
ALMs
Direct link interconnect to left
Direct link interconnect to right
Local Interconnect
MLAB
LAB

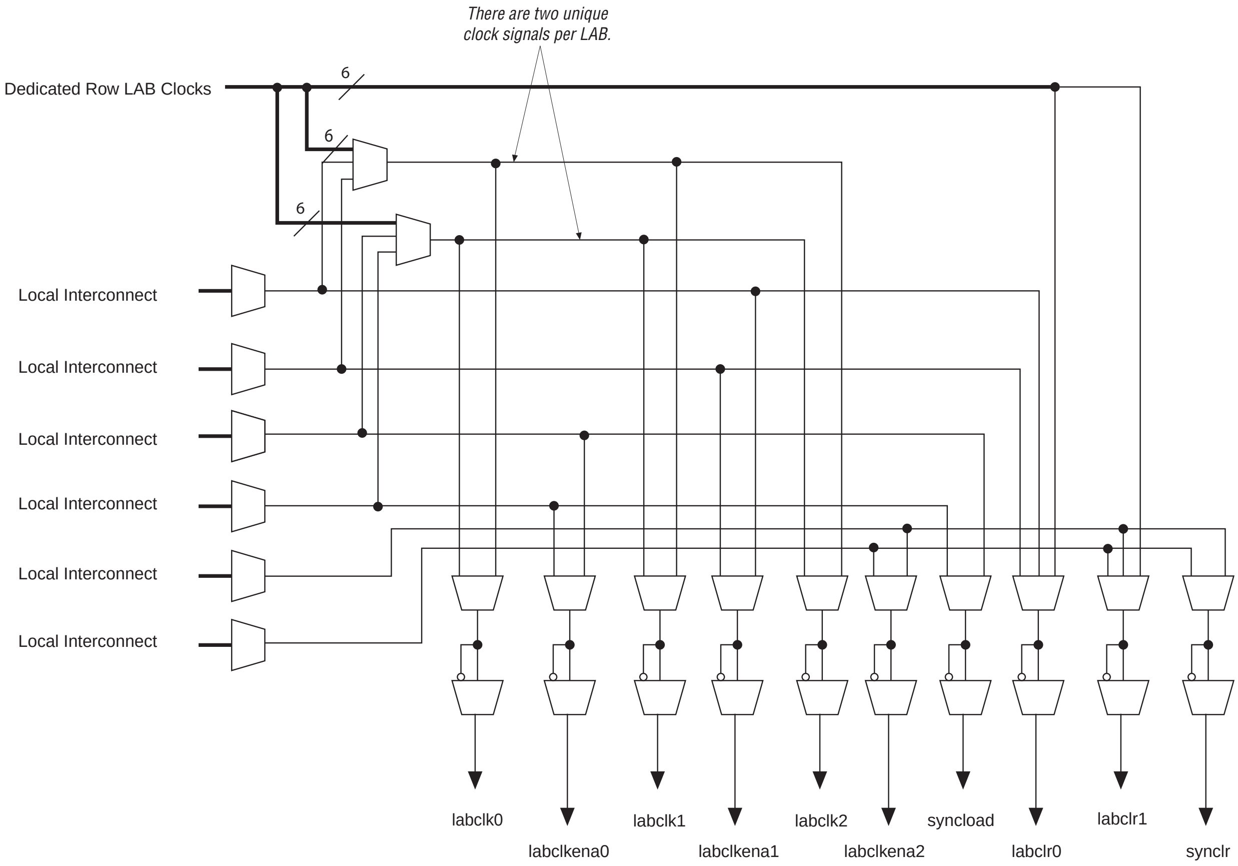
LAB Control Signals

Each LAB contains dedicated logic for driving control signals to its ALMs. The control signals include three clocks, three clock enables, two asynchronous clears, a synchronous clear, and synchronous load control signals. This gives a maximum of 10 control signals at a time. Although you generally use synchronous-load and clear signals when implementing counters, you can also use them with other functions.

Each LAB has two unique clock sources and three clock enable signals, as shown in Figure 2–4. The LAB control block can generate up to three clocks using the two clock sources and three clock enable signals. Each LAB's clock and clock enable signals are linked. For example, any ALM in a particular LAB using the `labclk1` signal also uses the `labclkena1` signal. If the LAB uses both the rising and falling edges of a clock, it also uses two LAB-wide clock signals. De-asserting the clock enable signal turns off the corresponding LAB-wide clock.

The LAB row clocks [5..0] and LAB local interconnects generate the LAB-wide control signals. The MultiTrack™ interconnect's inherent low skew allows clock and control signal distribution in addition to data. Figure 2–4 shows the LAB control signal generation circuit.

Figure 2–4. LAB-Wide Control Signals



Adaptive Logic Modules

The basic building block of logic in the Stratix IV architecture, the ALM provides advanced features with efficient logic utilization. Each ALM contains a variety of LUT-based resources that can be divided between two combinational adaptive LUTs (ALUTs) and two registers. With up to eight inputs to the two combinational ALUTs, one ALM can implement various combinations of two functions. This adaptability allows an ALM to be completely backward-compatible with four-input LUT architectures. One ALM can also implement any function with up to six inputs and certain seven-input functions.

In addition to the adaptive LUT-based resources, each ALM contains two programmable registers, two dedicated full adders, a carry chain, a shared arithmetic chain, and a register chain. Through these dedicated resources, an ALM can efficiently implement various arithmetic functions and shift registers. Each ALM drives all types of interconnects: local, row, column, carry chain, shared arithmetic chain, register chain, and direct link. Figure 2–5 shows a high-level block diagram of the Stratix IV ALM.

Figure 2–5. High-Level Block Diagram of the Stratix IV ALM

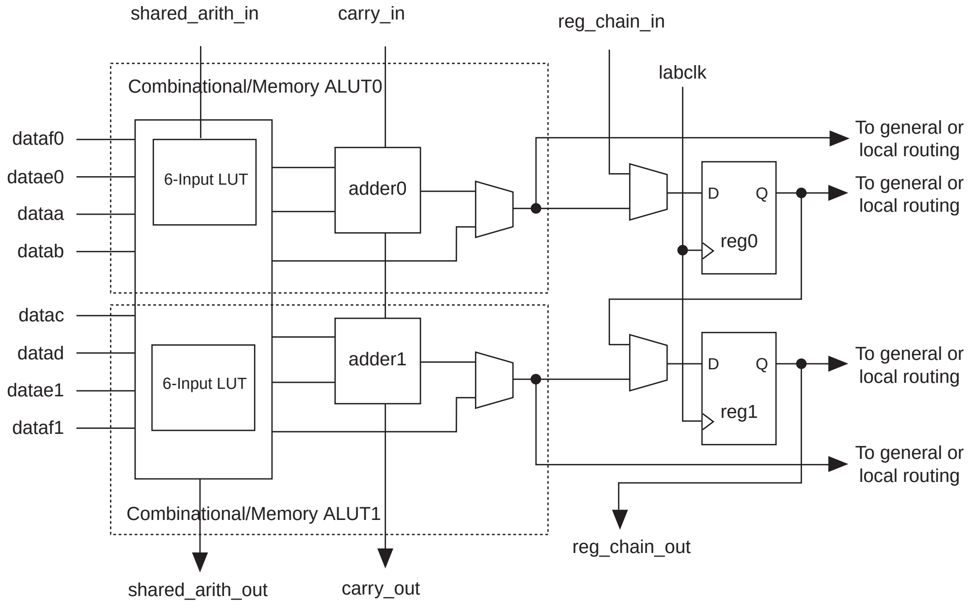
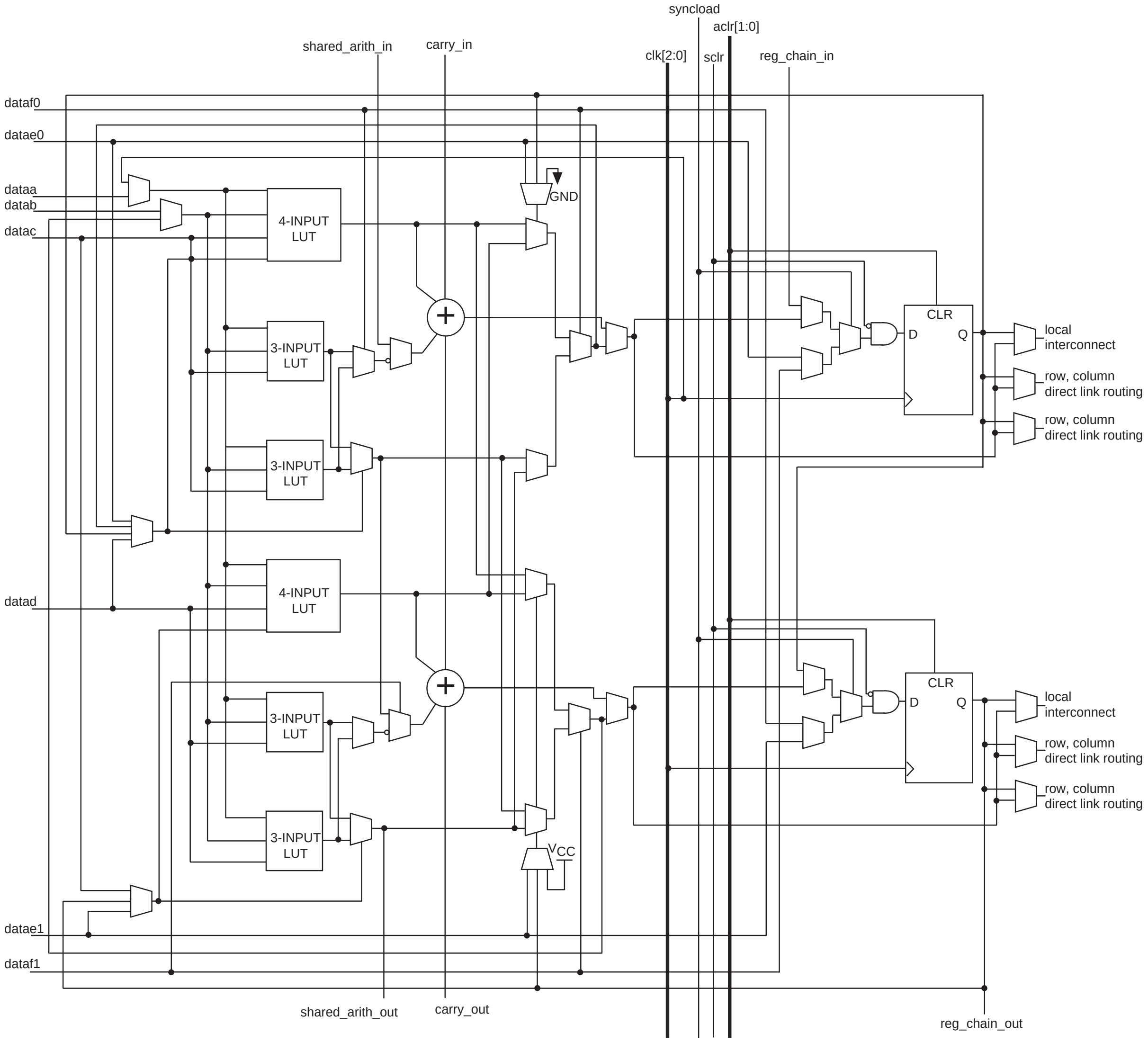


Figure 2–6 shows a detailed view of all the connections in an ALM.

Figure 2–6. Stratix IV ALM Connection Details



One ALM contains two programmable registers. Each register has data, clock, clock enable, synchronous and asynchronous clear, and synchronous load and clear inputs. Global signals, general-purpose I/O pins, or any internal logic can drive the register's clock and clear-control signals. Either general-purpose I/O pins or internal logic can drive the clock enable. For combinational functions, the register is bypassed and the output of the LUT drives directly to the outputs of an ALM.

Each ALM has two sets of outputs that drive the local, row, and column routing resources. The LUT, adder, or register outputs can drive these output drivers (refer to Figure 2–6). For each set of output drivers, two ALM outputs can drive column, row, or direct link routing connections, and one of these ALM outputs can also drive local interconnect resources. This allows the LUT or adder to drive one output while the register drives another output.

This feature, called register packing, improves device utilization because the device can use the register and the combinational logic for unrelated functions. Another special packing mode allows the register output to feed back into the LUT of the same ALM so that the register is packed with its own fan-out LUT. This provides another mechanism for improved fitting. The ALM can also drive out registered and unregistered versions of the LUT or adder output.

ALM Operating Modes

The Stratix IV ALM can operate in one of the following modes:

- Normal
- Extended LUT
- Arithmetic
- Shared Arithmetic
- LUT-Register

Each mode uses ALM resources differently. In each mode, eleven available inputs to an ALM—the eight data inputs from the LAB local interconnect, carry-in from the previous ALM or LAB, the shared arithmetic chain connection from the previous ALM or LAB, and the register chain connection—are directed to different destinations to implement the desired logic function. LAB-wide signals provide clock, asynchronous clear, synchronous clear, synchronous load, and clock enable control for the register. These LAB-wide signals are available in all ALM modes.

☞ Refer to "LAB Control Signals" on page 2–4 for more information on the LAB-wide control signals.

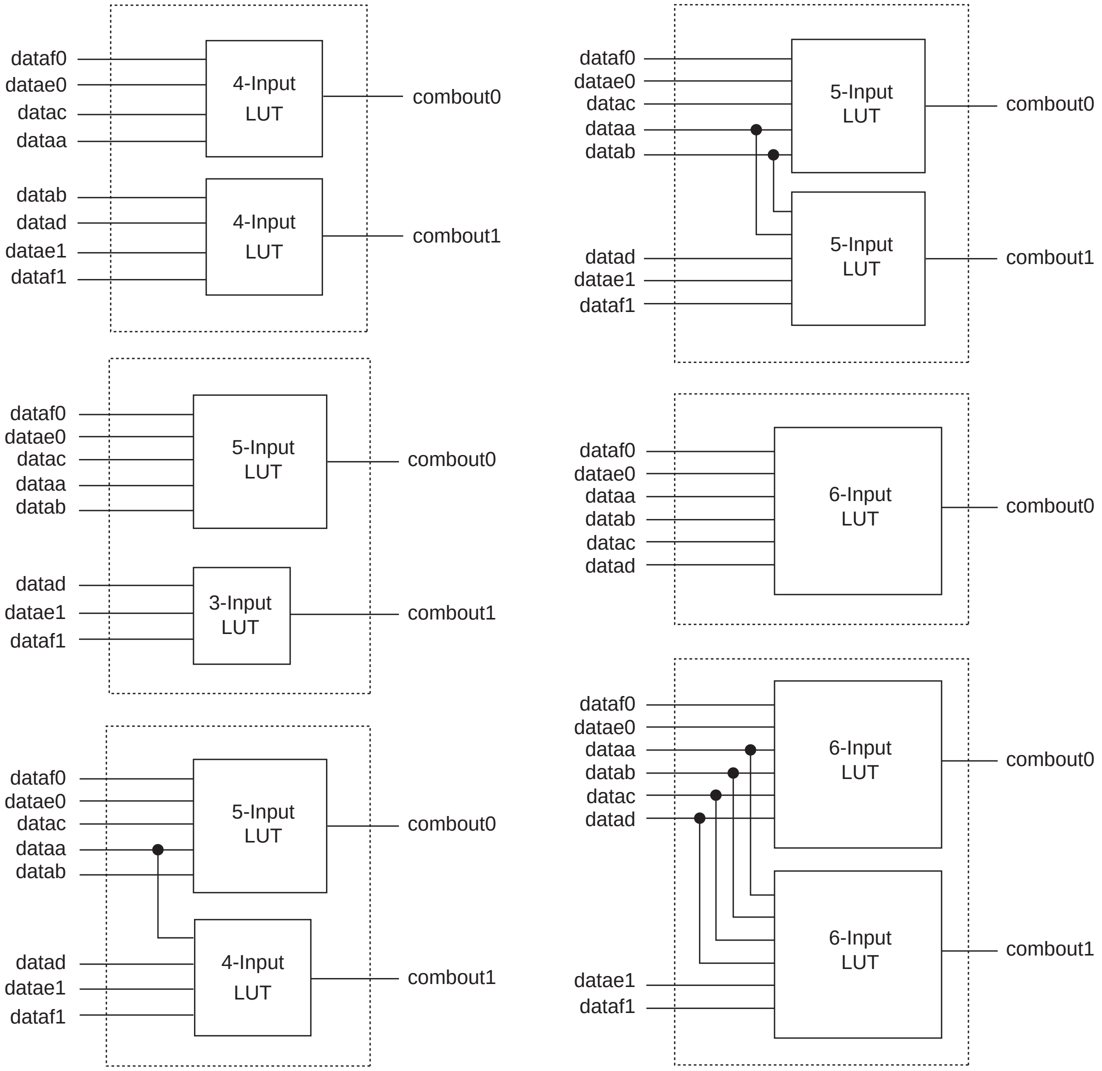
The Quartus II software and supported third-party synthesis tools, in conjunction with parameterized functions such as the library of parameterized modules (LPM) functions, automatically choose the appropriate mode for common functions such as counters, adders, subtractors, and arithmetic functions.

Normal Mode

The normal mode is suitable for general logic applications and combinational functions. In this mode, up to eight data inputs from the LAB local interconnect are inputs to the combinational logic. The normal mode allows two functions to be implemented in one Stratix IV ALM, or a single function of up to six inputs. The ALM can support certain combinations of completely independent functions and various combinations of functions that have common inputs.

Figure 2–7 shows the supported LUT combinations in normal mode.

Figure 2–7. ALM in Normal Mode *(Note 1)*



Note to Figure 2–7:

(1) Combinations of functions with fewer inputs than those shown are also supported. For example, combinations of functions with the following number of inputs are supported: 4 and 3, 3 and 3, 3 and 2, and 5 and 2.

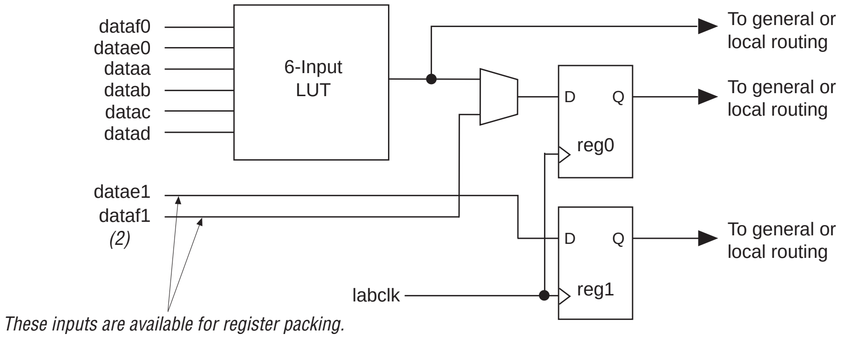
The normal mode provides complete backward compatibility with four-input LUT architectures.

For the packing of 2 five-input functions into one ALM, the functions must have at least two common inputs. The common inputs are `dataa` and `datab`. The combination of a four-input function with a five-input function requires one common input (either `dataa` or `datab`).

In the case of implementing 2 six-input functions in one ALM, four inputs must be shared and the combinational function must be the same. In a sparsely used device, functions that could be placed in one ALM may be implemented in separate ALMs by the Quartus II software to achieve the best possible performance. As a device begins to fill up, the Quartus II software automatically utilizes the full potential of the Stratix IV ALM. The Quartus II Compiler automatically searches for functions using common inputs or completely independent functions to be placed in one ALM and make efficient use of the device resources. In addition, you can manually control resource usage by setting location assignments.

Any six-input function can be implemented utilizing inputs `dataa`, `datab`, `datac`, `datad`, and either `datae0` and `dataf0` or `datae1` and `dataf1`. If `datae0` and `dataf0` are utilized, the output is driven to `register0`, and/or `register0` is bypassed and the data drives out to the interconnect using the top set of output drivers (refer to Figure 2–8). If `datae1` and `dataf1` are utilized, the output either drives to `register1` or bypasses `register1` and drives to the interconnect using the bottom set of output drivers. The Quartus II Compiler automatically selects the inputs to the LUT. ALMs in normal mode support register packing.

Figure 2–8. Input Function in Normal Mode *(Note 1)*


dataf0
datae0
dataa
datab
datac
datad
6-Input LUT
To general or local routing
D
Q
reg0
To general or local routing
datae1
dataf1
(2)
D
Q
To general or local routing
labclk
reg1
These inputs are available for register packing.

Notes to Figure 2–8:

(1) If `datae1` and `dataf1` are used as inputs to a six-input function, `datae0` and `dataf0` are available for register packing.

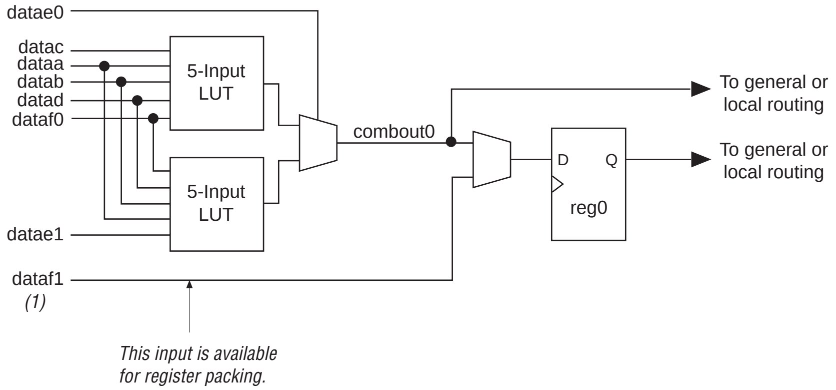
(2) The `dataf1` input is available for register packing only if the six-input function is unregistered.

Extended LUT Mode

Use the extended LUT mode to implement a specific set of seven-input functions. The set must be a 2-to-1 multiplexer fed by two arbitrary five-input functions sharing four inputs. Figure 2–9 shows the template of supported seven-input functions utilizing extended LUT mode. In this mode, if the seven-input function is unregistered, the unused eighth input is available for register packing.

Functions that fit into the template shown in Figure 2–9 occur naturally in designs. These functions often appear in designs as “if-else” statements in Verilog HDL or VHDL code.

Figure 2–9. Template for Supported Seven-Input Functions in Extended LUT Mode


datae0
datac
dataa
datab
datad
dataf0
5-Input LUT
5-Input LUT
datae1
dataf1
(1)
combout0
To general or local routing
To general or local routing
D
Q
reg0
This input is available for register packing.

Note to Figure 2–9:

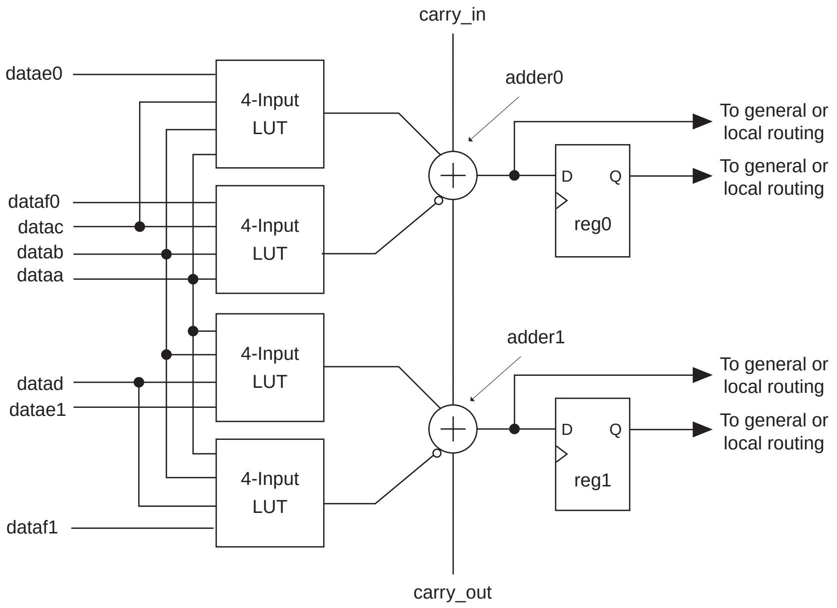
(1) If the seven-input function is unregistered, the unused eighth input is available for register packing. The second register, `reg1`, is not available.

Arithmetic Mode

The arithmetic mode is ideal for implementing adders, counters, accumulators, wide parity functions, and comparators. The ALM in arithmetic mode uses two sets of 2 four-input LUTs along with two dedicated full adders. The dedicated adders allow the LUTs to be available to perform pre-adder logic; therefore, each adder can add the output of 2 four-input functions.

The four LUTs share the `dataa` and `datab` inputs. As shown in Figure 2–10, the carry-in signal feeds to `adder0`, and the carry-out from `adder0` feeds to the carry-in of `adder1`. The carry-out from `adder1` drives to `adder0` of the next ALM in the LAB. ALMs in arithmetic mode can drive out registered and/or unregistered versions of the adder outputs.

Figure 2–10. ALM in Arithmetic Mode



While operating in arithmetic mode, the ALM can support simultaneous use of the adder's carry output along with combinational logic outputs. In this operation, the adder output is ignored. This usage of the adder with the combinational logic output provides resource savings of up to 50% for functions that can use this ability.

The arithmetic mode also offers clock enable, counter enable, synchronous up/down control, add/subtract control, synchronous clear, and synchronous load. The LAB local interconnect data inputs generate the clock enable, counter enable, synchronous up/down, and add/subtract control signals. These control signals are good candidates for the inputs that are shared between the four LUTs in the ALM. The synchronous clear and synchronous load options are LAB-wide signals that affect all registers in the LAB. These signals can also be individually disabled or enabled per register. The Quartus II software automatically places any registers that are not used by the counter into other LABs.

Carry Chain

The carry chain provides a fast carry function between the dedicated adders in arithmetic or shared-arithmetic mode. The two-bit carry select feature in Stratix IV devices halves the propagation delay of carry chains within the ALM. Carry chains can begin in either the first ALM or the fifth ALM in an LAB. The final carry-out signal is routed to an ALM, where it is fed to local, row, or column interconnects.

The Quartus II Compiler automatically creates carry-chain logic during design processing, or you can create it manually during design entry. Parameterized functions such as LPM functions automatically take advantage of carry chains for the appropriate functions.

The Quartus II Compiler creates carry chains longer than 20 (10 ALMs in arithmetic or shared arithmetic mode) by linking LABs together automatically. For enhanced fitting, a long carry chain runs vertically, allowing fast horizontal connections to TriMatrix™ memory and DSP blocks. A carry chain can continue as far as a full column.

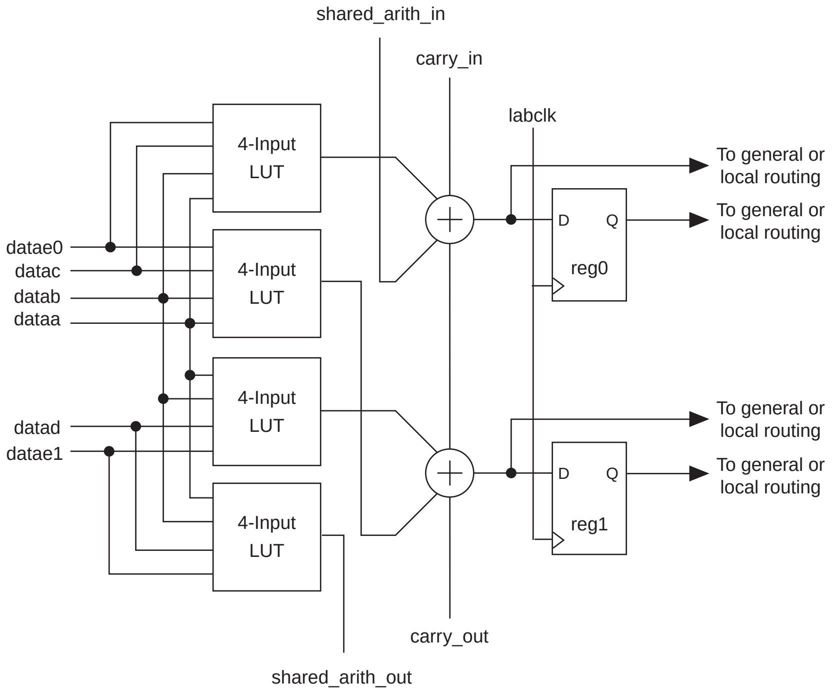
To avoid routing congestion in one small area of the device when a high fan-in arithmetic function is implemented, the LAB can support carry chains that only utilize either the top half or the bottom half of the LAB before connecting to the next LAB. This leaves the other half of the ALMs in the LAB available for implementing narrower fan-in functions in normal mode. Carry chains that use the top five ALMs in the first LAB carry into the top half of the ALMs in the next LAB within the column. Carry chains that use the bottom five ALMs in the first LAB carry into the bottom half of the ALMs in the next LAB within the column. In every alternate LAB column, the top half can be bypassed; in the other MLAB columns, the bottom half can be bypassed.

☞ Refer to "ALM Interconnects" on page 2–17 for more information about carry-chain interconnects.

Shared Arithmetic Mode

In shared arithmetic mode, the ALM can implement a three-input add within an ALM. In this mode, the ALM is configured with 4, four-input LUTs. Each LUT either computes the sum of three inputs or the carry of three inputs. The output of the carry computation is fed to the next adder (either to `adder1` in the same ALM or to `adder0` of the next ALM in the LAB) using a dedicated connection called the shared arithmetic chain. This shared arithmetic chain can significantly improve the performance of an adder tree by reducing the number of summation stages required to implement an adder tree. Figure 2–11 shows the ALM using this feature.

Figure 2–11. ALM in Shared Arithmetic Mode



You can find adder trees in many different applications. For example, the summation of the partial products in a logic-based multiplier can be implemented in a tree structure. Another example is a correlator function that can use a large adder tree to sum filtered data samples in a given time frame to recover or de-spread data that was transmitted utilizing spread-spectrum technology.

Shared Arithmetic Chain

The shared arithmetic chain available in enhanced arithmetic mode allows the ALM to implement a three-input add. This significantly reduces the resources necessary to implement large adder trees or correlator functions.

The shared arithmetic chains can begin in either the first or sixth ALM in an LAB. The Quartus II Compiler creates shared arithmetic chains longer than 20 (10 ALMs in arithmetic or shared arithmetic mode) by linking LABs together automatically. For enhanced fitting, a long shared arithmetic chain runs vertically, allowing fast horizontal connections to TriMatrix memory and DSP blocks. A shared arithmetic chain can continue as far as a full column.

Similar to the carry chains, the top and bottom halves of shared arithmetic chains in alternate LAB columns can be bypassed. This capability allows the shared arithmetic chain to cascade through half of the ALMs in an LAB while leaving the other half available for narrower fan-in functionality. Every other LAB column is top-half by-passable, while the other LAB columns are bottom-half by-passable.

☞ Refer to "ALM Interconnects" on page 2–17 for more information on shared arithmetic chain interconnect.

LUT-Register Mode

LUT-Register mode allows third-register capability within an ALM. Two internal feedback loops allow combinational `ALUT1` to implement the master latch and combinational `ALUT0` to implement the slave latch needed for the third register. The LUT register shares its clock, clock enable, and asynchronous clear sources with the top dedicated register. Figure 2–12 shows the register constructed using two combinational blocks within the ALM.

Figure 2–12. LUT Register from Two Combinational Blocks

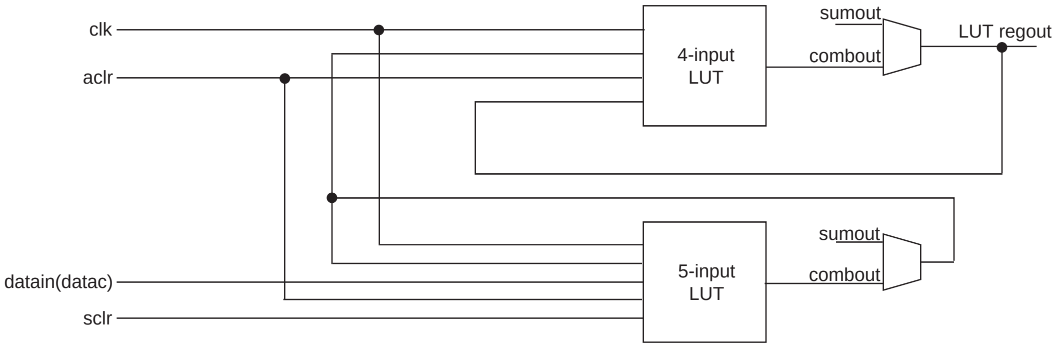
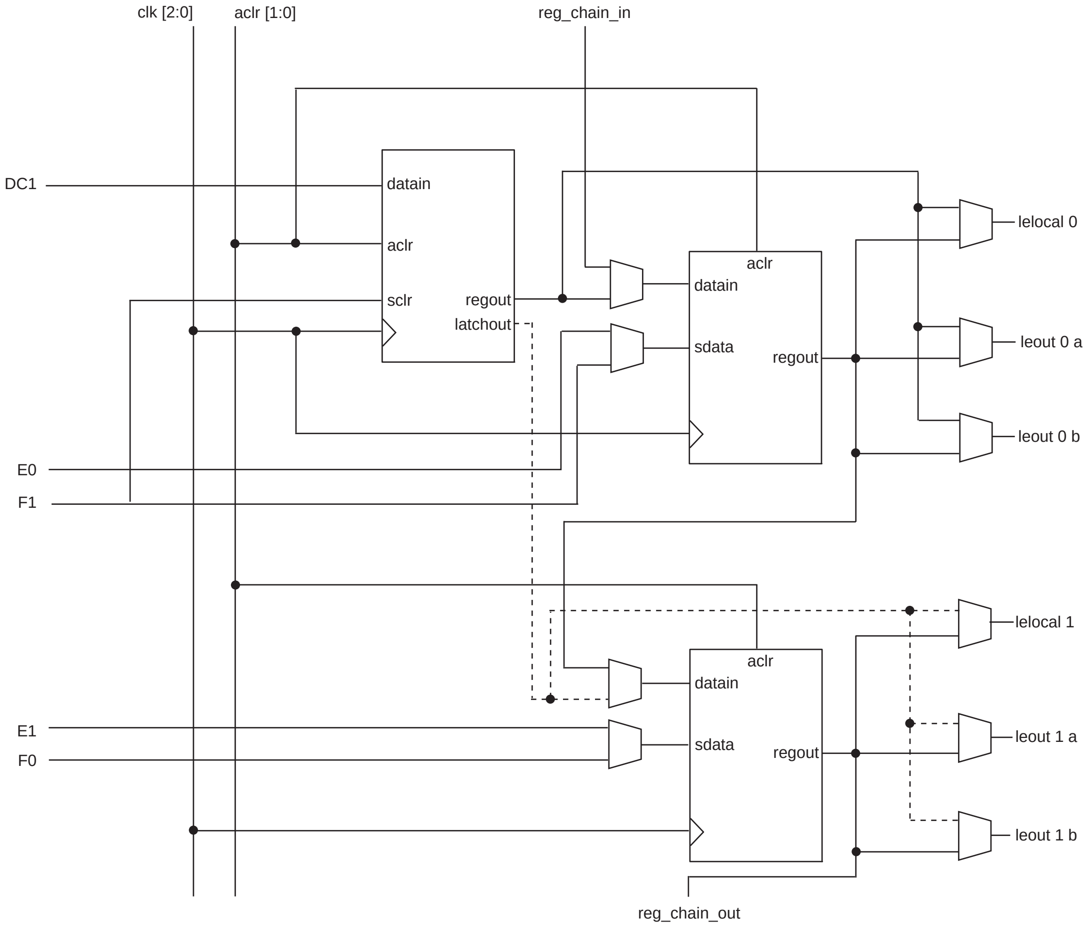


Figure 2–13 shows the ALM in LUT-register mode.

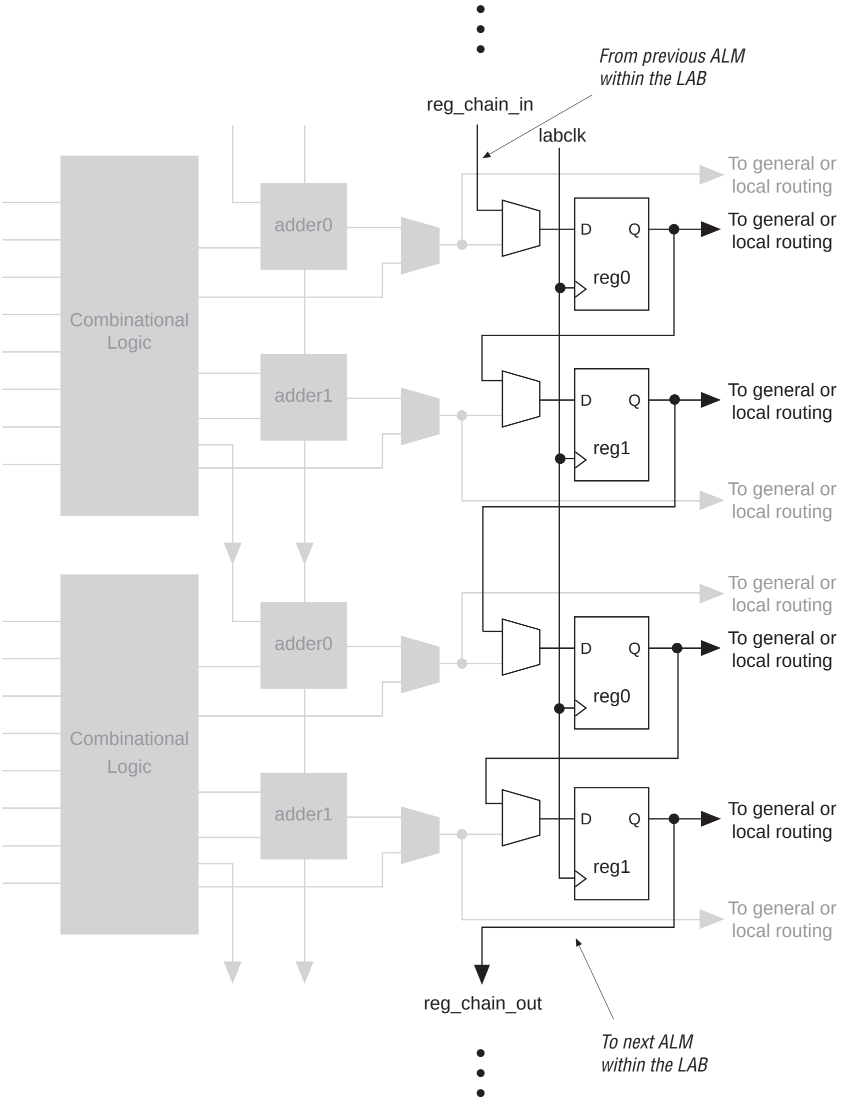
Figure 2–13. ALM in LUT-Register Mode with 3-Register Capability



Register Chain

In addition to the general routing outputs, the ALMs in an LAB have register-chain outputs. The register-chain routing allows registers in the same LAB to be cascaded together. The register-chain interconnect allows an LAB to use LUTs for a single combinational function and the registers to be used for an unrelated shift-register implementation. These resources speed up connections between ALMs while saving local interconnect resources (refer to Figure 2–14). The Quartus II Compiler automatically takes advantage of these resources to improve utilization and performance.

Figure 2–14. Register Chain within an LAB *(Note 1)*



Note to Figure 2–14:

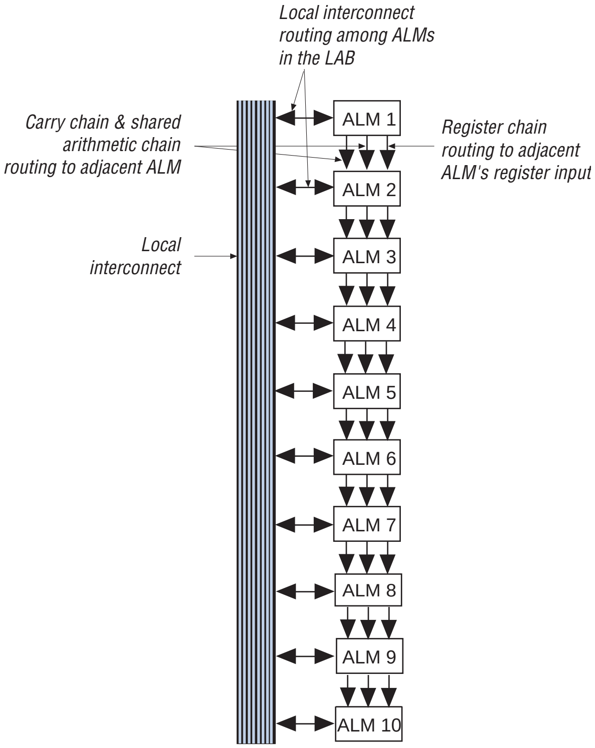
(1) You can use the combinational or adder logic to implement an unrelated, un-registered function.

Refer to "ALM Interconnects" on page 2–17 for more information about register chain interconnect.

ALM Interconnects

There are three dedicated paths between ALMs: register cascade, carry chain, and shared arithmetic chain. Stratix IV devices include an enhanced interconnect structure in LABs for routing shared arithmetic chains and carry chains for efficient arithmetic functions. The register chain connection allows the register output of one ALM to connect directly to the register input of the next ALM in the LAB for fast shift registers. These ALM-to-ALM connections bypass the local interconnect. The Quartus II compiler automatically takes advantage of these resources to improve utilization and performance. Figure 2–15 shows the shared arithmetic chain, carry chain, and register chain interconnects.

Figure 2–15. Shared Arithmetic Chain, Carry Chain, and Register Chain Interconnects


Local interconnect routing among ALMs in the LAB
Carry chain & shared arithmetic chain routing to adjacent ALM
Register chain routing to adjacent ALM's register input
Local interconnect
ALM 1
ALM 2
ALM 3
ALM 4
ALM 5
ALM 6
ALM 7
ALM 8
ALM 9
ALM 10

Clear and Preset Logic Control

LAB-wide signals control the logic for the register's clear signal. The ALM directly supports an asynchronous clear function. You can achieve the register preset through the Quartus II software's NOT-gate push-back logic option. Each LAB supports up to two clears.

Stratix IV devices provide a device-wide reset pin (DEV_CLRn) that resets all registers in the device. An option set before compilation in the Quartus II software controls this pin. This device-wide reset overrides all other control signals.

LAB Power Management Techniques

The following techniques are used to manage static and dynamic power consumption within the LAB:

- To save AC power, the Quartus II software forces all adder inputs low when ALM adders are not in use.
- Stratix IV LABs operate in high-performance mode or low-power mode. The Quartus II software automatically chooses the appropriate mode for an LAB, based on the design, to optimize speed versus leakage trade-offs.
- Clocks represent a significant portion of dynamic power consumption due to their high switching activity and long paths. The LAB clock that distributes a clock signal to registers within an LAB is a significant contributor to overall clock power consumption. Each LAB's clock and clock enable signal are linked. For example, a combinational ALUT or register in a particular LAB using the `labclk1` signal also uses the `labclkena1` signal. To disable an LAB-wide clock power consumption without disabling the entire clock tree, use the LAB-wide clock enable to gate the LAB-wide clock. The Quartus II software automatically promotes register-level clock enable signals to the LAB-level. All registers within an LAB that share a common clock and clock enable are controlled by a shared, gated clock. To take advantage of these clock enables, use a clock-enable construct in your HDL code for the registered logic.

For details about implementing static and dynamic power consumption within the LAB, refer to the *Power Optimization* chapter in volume 2 of the *Quartus II Handbook*.

Conclusion

Logic array blocks and adaptive logic modules are the basic building blocks of the Stratix IV device. You can use these to configure logic functions, arithmetic functions, and register functions. The ALM provides advanced features with efficient logic utilization and is completely backward-compatible.

Referenced Documents

This chapter references the following documents:

- *Power Optimization* chapter in volume 2 of the *Quartus II Handbook*
- *TriMatrix Embedded Memory Blocks in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook*

Document Revision History

Table 2–1 shows the revision history for this document.

Table 2–1. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	■ Updated Figure 2–6. ■ Made minor editorial changes.	—
May 2008 v1.0	Initial Release.	—

3. TriMatrix Embedded Memory Blocks in Stratix IV Devices

SIV51003-2.0

Introduction

TriMatrix™ embedded memory blocks provide three different sizes of embedded SRAM to efficiently address the needs of Stratix® IV FPGA designs. TriMatrix memory includes 640-bit memory logic array blocks (MLABs), 9-Kbit M9K blocks, and 144-Kbit M144K blocks. MLABs have been optimized to implement filter delay lines, small first-in first-out (FIFO) buffers, and shift registers. You can use the M9K blocks for general purpose memory applications, and the M144K blocks are ideal for processor code storage, packet buffering, and video frame buffering.

You can independently configure each embedded memory block to be a single- or dual-port RAM, FIFO buffer, ROM, or shift register using the Quartus® II MegaWizard® Plug-In Manager. You can stitch together multiple blocks of the same type to produce larger memories with minimal timing penalty. TriMatrix memory provides up to 31,491 Kbits of embedded SRAM at up to 600 MHz operation. This chapter describes TriMatrix memory blocks, modes, features, and design considerations.

This chapter contains the following sections:

- "Overview" on page 3–1
- "Memory Modes" on page 3–8
- "Clocking Modes" on page 3–16
- "Design Considerations" on page 3–17
- "Conclusion" on page 3–20

Overview

Table 3–1 summarizes the features supported by the three sizes of TriMatrix memory.

Table 3–1. Summary of TriMatrix Memory Features (Part 1 of 2)

Feature	MLABs	M9K Blocks	M144K Blocks
Maximum performance	600 MHz	600 MHz	600 MHz
Total RAM bits (including parity bits)	640	9216	147,456

Table 3–1. Summary of TriMatrix Memory Features (Part 2 of 2)

Feature	MLABs	M9K Blocks	M144K Blocks
Configurations (depth × width)	64×8 64×9 64×10 32×16 32×18 32×20	8K×1 4K×2 2K×4 1K×8 1K×9 512×16 512×18 256×32 256×36	16K×8 16K×9 8K×16 8K×18 4K×32 4K×36 2K×64 2K×72
Parity bits	✓	✓	✓
Byte enable	✓	✓	✓
Packed mode	—	✓	✓
Address clock enable	✓	✓	✓
Single-port memory	✓	✓	✓
Simple dual-port memory	✓	✓	✓
True dual-port memory	—	✓	✓
Embedded shift register	✓	✓	✓
ROM	✓	✓	✓
FIFO buffer	✓	✓	✓
Simple dual-port mixed width support	—	✓	✓
True dual-port mixed width support	—	✓	✓
Memory initialization file (**.mif**)	✓	✓	✓
Mixed-clock mode	✓	✓	✓
Power-up condition	Outputs cleared if registered, otherwise reads memory contents.	Outputs cleared	Outputs cleared
Register clears	Output registers	Output registers	Output registers
Write/Read operation triggering	Write: Falling clock edges Read: Rising clock edges	Write and Read: Rising clock edges	Write and Read: Rising clock edges
Same-port read-during-write	Outputs set to old data or don't care	Outputs set to old data, new data, or don't care	Outputs set to old data, new data, or don't care
Mixed-port read-during-write	Outputs set to old data or don't care	Outputs set to old data or don't care	Outputs set to old data or don't care
ECC Support	Soft IP support using Quartus II	Soft IP support using Quartus II	Built-in support in ×64 wide SDP mode or soft IP support using Quartus II

Table 3–2 shows the capacity and distribution of the TriMatrix memory blocks in each Stratix IV family member.

Table 3–2. TriMatrix Memory Capacity and Distribution in Stratix IV Devices

Device	MLABs	M9K Blocks	M144K Blocks	Total Dedicated RAM Bits (dedicated memory blocks only)	Total RAM Bits (including MLABs)
EP4SE110	2112	660	16	8244 Kb	9564 Kb
EP4SE230	4560	1235	22	14,283 Kb	17,133 Kb
EP4SE290	5824	936	36	13,608 Kb	17,248 Kb
EP4SE360	7072	1248	48	18,144 Kb	22,564 Kb
EP4SE530	10,624	1280	64	20,736 Kb	27,376 Kb
EP4SE680	13,622	1529	64	22,977 Kb	31,491 Kb
EP4SGX70	1452	462	16	6462 Kb	7370 Kb
EP4SGX110	2112	660	16	8244 Kb	9564 Kb
EP4SGX230	4560	1235	22	14,283 Kb	17,133 Kb
EP4SGX290	5824	936	36	13,608 Kb	17,248 Kb
EP4SGX360	7072	1248	48	18,144 Kb	22,564 Kb
EP4SGX530	10,624	1280	64	20,736 Kb	27,376 Kb

TriMatrix Memory Block Types

While the M9K and M144K memory blocks are dedicated resources, the MLABs are dual-purpose blocks. They can be configured as regular logic array blocks (LABs) or as MLABs. Ten adaptive logic modules (ALMs) make up one MLAB. Each ALM in an MLAB can be configured as either a 64 × 1 or a 32 × 2 block, resulting in a 64 × 10 or 32 × 20 simple dual-port SRAM block in a single MLAB.

Parity Bit Support

All TriMatrix memory blocks have built-in parity-bit support. The ninth bit associated with each byte can store a parity bit or serve as an additional data bit. No parity function is actually performed on the ninth bit.

Byte Enable Support

All TriMatrix memory blocks support byte enables that mask the input data so that only specific bytes of data are written. The unwritten bytes retain the previously written values. The write enable (`wren`) signals, along with the byte enable (`byteena`) signals, control the RAM blocks' write operations.

The default value for the byte enable signals is high (enabled), in which case writing is controlled only by the write enable signals. The byte enable registers have no clear port. When using parity bits on the M9K and M144K blocks, the byte enable controls all nine bits (eight bits of data plus one parity bit). When using parity bits on the MLAB, the byte-enable controls all 10 bits in the widest mode.

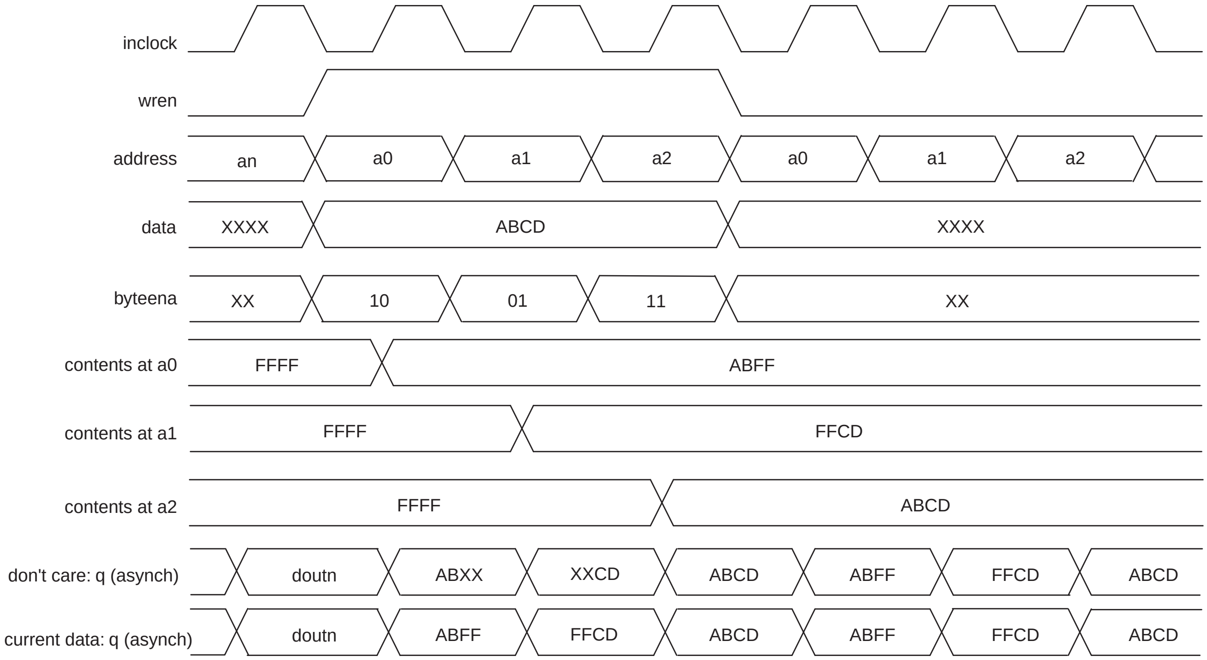
Byte enables operate in a one-hot fashion, with the least significant bit (LSB) of the `byteena` signal corresponding to the least significant byte of the data bus. For example, if using a RAM block in ×18 mode, with `byteena = 01`, `data[8..0]` is enabled and `data[17..9]` is disabled. Similarly, if `byteena = 11`, both `data[8..0]` and `data[17..9]` are enabled. Byte enables are active high.

You cannot use the byte enable feature when using the error correction coding (ECC) feature on M144K blocks.

Figure 3–1 shows how the write enable (`wren`) and byte enable (`byteena`) signals control the operations of the RAM.

When a byte-enable bit is de-asserted during a write cycle, the corresponding data byte output can appear as either a "don't care" value or the current data at that location. The output value for the masked byte is controllable using the Quartus II software. When a byte-enable bit is asserted during a write cycle, the corresponding data byte output also depends on the setting chosen in the Quartus II software.

Figure 3–1. Stratix IV Byte Enable Functional Waveform



Packed Mode Support

Stratix IV M9K and M144K blocks support packed mode. The packed mode feature packs two independent single-port RAMs into one memory block. The Quartus II software automatically implements packed mode where appropriate by placing the physical RAM block into true dual-port mode and using the most significant bit (MSB) of the address to distinguish between the two logical RAMs. The size of each independent single-port RAM must not exceed half of the target block size.

Address Clock Enable Support

All Stratix IV memory blocks support address clock enable, which holds the previous address value for as long as the signal is enabled (`addressstall` = `1`). When the memory blocks are configured in dual-port mode, each port has its own independent address clock enable. The default value for the address clock enable signals is low (disabled).

Figure 3–2 shows an address clock enable block diagram. The address clock enable is referred to by the port name `addressstall`.

Figure 3–2. Stratix IV Address Clock Enable Block Diagram

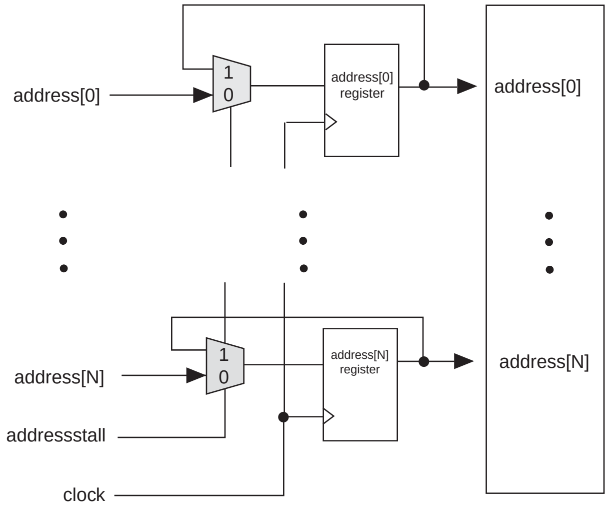


Figure 3–3 shows the address clock enable waveform during the read cycle.

Figure 3–3. Stratix IV Address Clock Enable During Read Cycle Waveform

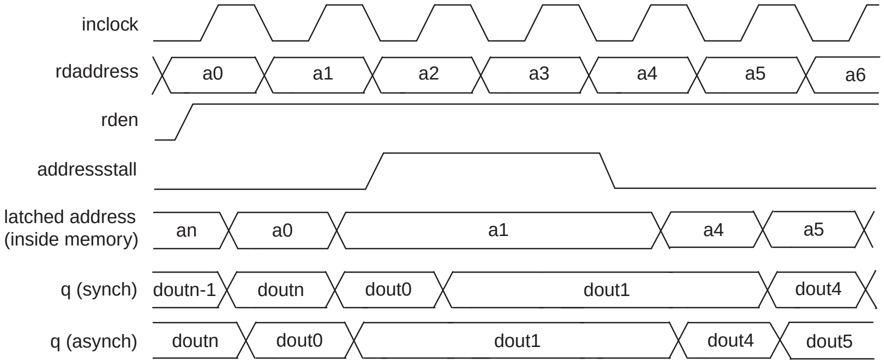
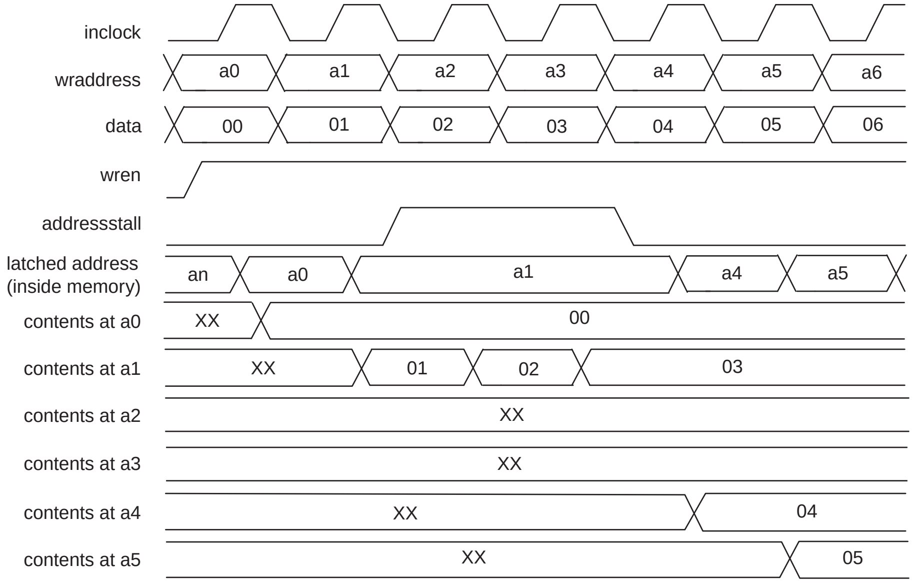


Figure 3–4 shows the address clock enable waveform during write cycle.

Figure 3–4. Stratix IV Address Clock Enable During Write Cycle Waveform



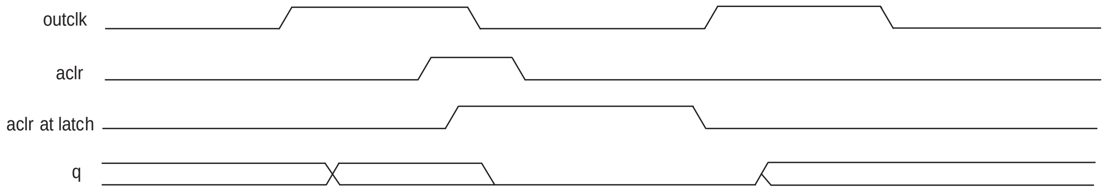
Mixed Width Support

M9K and M144K memory blocks support mixed data widths inherently. MLABs can support mixed data widths through emulation using the Quartus II software. When using simple dual-port, true dual-port, or FIFO modes, mixed width support allows you to read and write different data widths to a memory block. See "Memory Modes" on page 3–8 for details on the different widths supported per memory mode.

Asynchronous Clear

Stratix IV TriMatrix memory blocks support asynchronous clears on the output latches and output registers. Therefore, if your RAM is not using the output registers, you can still clear the RAM outputs using the output latch asynchronous clear. A functional waveform showing this functionality is shown in Figure 3–5.

Figure 3–5. Output Latch Asynchronous Clear Waveform



You can selectively enable asynchronous clears per logical memory using the Quartus II RAM MegaWizard Plug-In Manager.

For more information, refer to the *RAM Megafunction User Guide*.

Error Correction Code (ECC) Support

Stratix IV M144K blocks have built-in support for error correction code (ECC) when in ×64-wide simple dual-port mode. ECC allows you to detect and correct data errors in the memory array. The M144K blocks have a single-error-correction double-error-detection (SECDED) implementation. SECDED can detect and fix a single bit error in a 64-bit word, or detect two bit errors in a 64-bit word. It cannot detect three or more errors.

The M144K ECC status is communicated using a three-bit status flag `eccstatus[2..0]`. The status flag can be either registered or unregistered. When registered, it uses the same clock and asynchronous clear signals as the output registers. When not registered, it cannot be asynchronously cleared.

Table 3–3 shows the truth table for the ECC status flags.

Table 3–3. Truth Table for ECC Status Flags

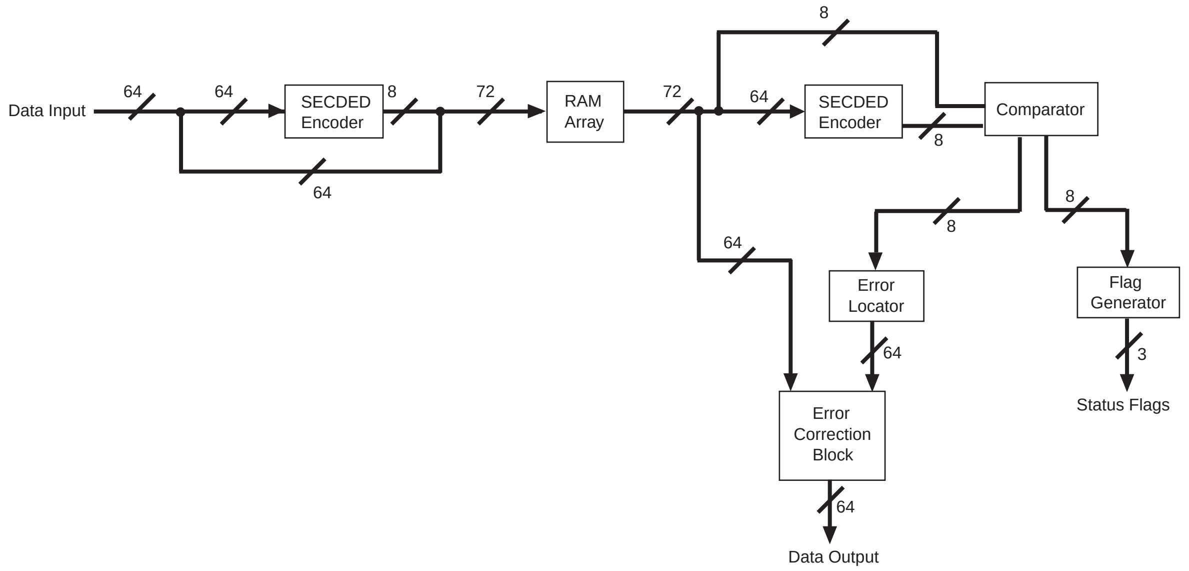
Status	eccstatus[2]	eccstatus[1]	eccstatus[0]
No error	0	0	0
Single error and fixed	0	1	1
Double error and no fix	1	0	1
Illegal	0	0	1
Illegal	0	1	0
Illegal	1	0	0
Illegal	1	1	X

☞ You cannot use the byte enable feature when ECC is engaged.

☞ Read during write “old data” mode is not supported when ECC is engaged.

Figure 3–6 shows a block diagram of the ECC block of the M144K.

Figure 3–6. ECC Block Diagram of the M144K



Memory Modes

Stratix IV TriMatrix memory blocks allow you to implement fully synchronous SRAM memory in multiple modes of operation. M9K and M144K blocks do not support asynchronous memory (unregistered inputs). MLABs support asynchronous (flow-through) read operations.

Depending on which TriMatrix memory block you target, the following modes may be used:

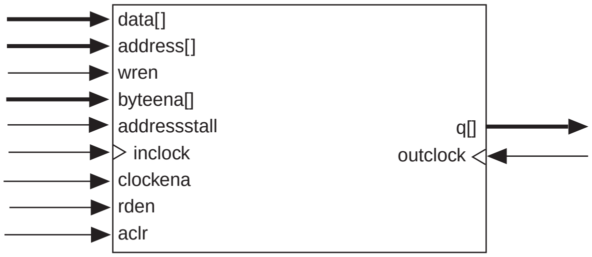
- Single-port
- Simple dual-port
- True dual-port
- Shift-register
- ROM
- FIFO

☞ When using the memory blocks in ROM, single-port, simple dual-port, or true dual-port mode, you can corrupt the memory contents if you violate the setup or hold-time on any of the memory block input registers. This applies to both read and write operations.

Single-Port RAM

All TriMatrix memory blocks support single-port mode. Single-port mode allows you to do either one-read or one-write operation at a time. Simultaneous reads and writes are not supported in single-port mode. Figure 3–7 shows the single-port RAM configuration.

Figure 3–7. Single-Port Memory *(Note 1)*



Note to Figure 3–7:

(1) You can implement two single-port memory blocks in a single M9K or M144K block. See "Packed Mode Support" on page 3–4 for more details.

During a write operation, behavior of the RAM outputs is configurable. If you use the read-enable signal and perform a write operation with the read enable deactivated, the RAM outputs retain the values they held during the most recent active read enable. If you activate read enable during a write operation, or if you are not using the read-enable signal at all, the RAM outputs either show the new data being written, the old data at that address, or a don't care value. To choose the desired behavior, set the read-during-write behavior to either new data, old data, or don't care in the RAM MegaWizard Plug-In Manager in the Quartus II software. See "Read During Write" on page 3–17 for more details about this behavior.

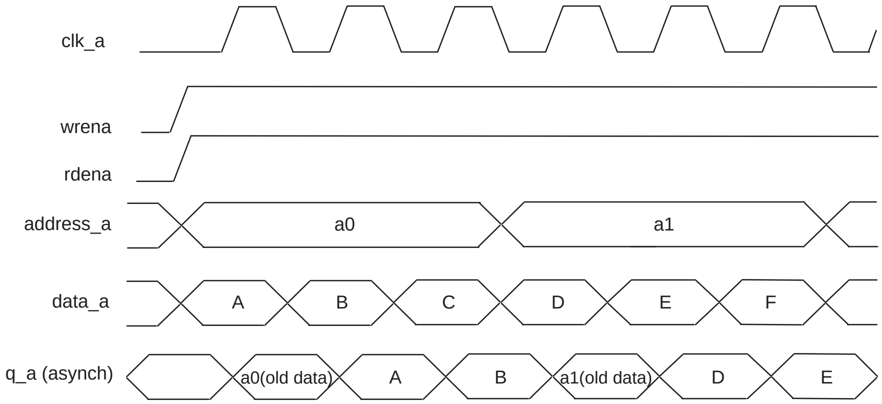
Table 3–4 shows the possible port width configurations for TriMatrix memory blocks in single-port mode.

Table 3–4. Stratix IV Port Width Configurations for MLABs, M9K Blocks, and M144K Blocks (Single-Port Mode)

	MLABs	**M9K Blocks**	**M144K Blocks**
Port Width Configurations	64×8	8K×1	16K×8
	64×9	4K×2	16K×9
	64×10	2K×4	8K×16
	32×16	1K×8	8K×18
	32×18	1K×9	4K×32
	32×20	512×16	4K×36
		512×18	2K×64
		256×32	2K×72
		256×36	

Figure 3–8 shows timing waveforms for read and write operations in single-port mode with unregistered outputs. Registering the RAM's outputs would simply delay the q output by one clock cycle.

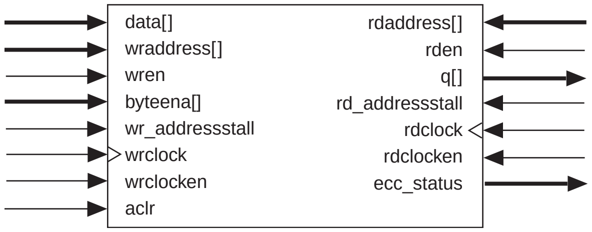
Figure 3–8. Timing Waveform for Read-Write Operations (Single-Port Mode)



Simple Dual-Port Mode

All TriMatrix memory blocks support simple dual-port mode. Simple dual-port mode allows you to perform one-read and one-write operation to different locations at the same time. Figure 3–9 shows a simple dual-port configuration.

Figure 3–9. Stratix IV Simple Dual-Port Memory *(Note 1)*



Note to Figure 3–9:

(1) Simple dual-port RAM supports input/output clock mode in addition to the read/write clock mode shown.

Simple dual-port mode supports different read and write data widths (mixed width support). Table 3–5 shows the mixed width configurations for the M9K blocks in simple dual-port mode. MLABs do not have native support for mixed width operation. The Quartus II software can implement mixed width memories in MLABs by using more than one MLAB.

Table 3–5. Stratix IV M9K Block Mixed-Width Configurations (Simple Dual-Port Mode) (Part 1 of 2)

Read Port	Write Port								
	8K×1	4K×2	2K×4	1K×8	512×16	256×32	1K×9	512×18	256×36
8K×1	✓	✓	✓	✓	✓	✓	—	—	—
4K×2	✓	✓	✓	✓	✓	✓	—	—	—
2K×4	✓	✓	✓	✓	✓	✓	—	—	—
1K×8	✓	✓	✓	✓	✓	✓	—	—	—
512×16	✓	✓	✓	✓	✓	✓	—	—	—
256×32	✓	✓	✓	✓	✓	✓	—	—	—

Table 3–5. Stratix IV M9K Block Mixed-Width Configurations (Simple Dual-Port Mode) (Part 2 of 2)

Read Port	Write Port								
	8K×1	**4K×2**	**2K×4**	**1K×8**	**512×16**	**256×32**	**1K×9**	**512×18**	**256×36**
1K×9	—	—	—	—	—	—	✓	✓	✓
512×18	—	—	—	—	—	—	✓	✓	✓
256×36	—	—	—	—	—	—	✓	✓	✓

Table 3–6 shows the mixed width configurations for the M144K blocks in simple dual-port mode.

Table 3–6. Stratix IV M144K Block Mixed-Width Configurations (Simple Dual-Port Mode)

Read Port	Write Port							
	16K×8	**8K×16**	**4K×32**	**2K×64**	**16K×9**	**8K×18**	**4K×36**	**2K×72**
16K×8	✓	✓	✓	✓	—	—	—	—
8K×16	✓	✓	✓	✓	—	—	—	—
4K×32	✓	✓	✓	✓	—	—	—	—
2K×64	✓	✓	✓	✓	—	—	—	—
16K×9	—	—	—	—	✓	✓	✓	✓
8K×18	—	—	—	—	✓	✓	✓	✓
4K×36	—	—	—	—	✓	✓	✓	✓
2K×72	—	—	—	—	✓	✓	✓	✓

In simple dual-port mode, M9K and M144K blocks support separate write-enable and read-enable signals. You can save power by keeping the read-enable signal low (inactive) when not reading. Read-during-write operations to the same address can either output a don't care value or old data. To choose the desired behavior, set the read-during-write behavior to either don't care or old data in the RAM MegaWizard Plug-In Manager in the Quartus II software. See "Read During Write" on page 3–17 for more details about this behavior.

MLABs only support a write-enable signal. Read-during-write behavior for the MLABs can be either don't care, new data, or old data. The available choices depend on the configuration of the MLAB.

Figure 3–10 shows timing waveforms for read and write operations in simple dual-port mode with unregistered outputs. Registering the RAM outputs would simply delay the q output by one clock cycle.

Figure 3–10. Stratix IV Simple Dual-Port Timing Waveforms

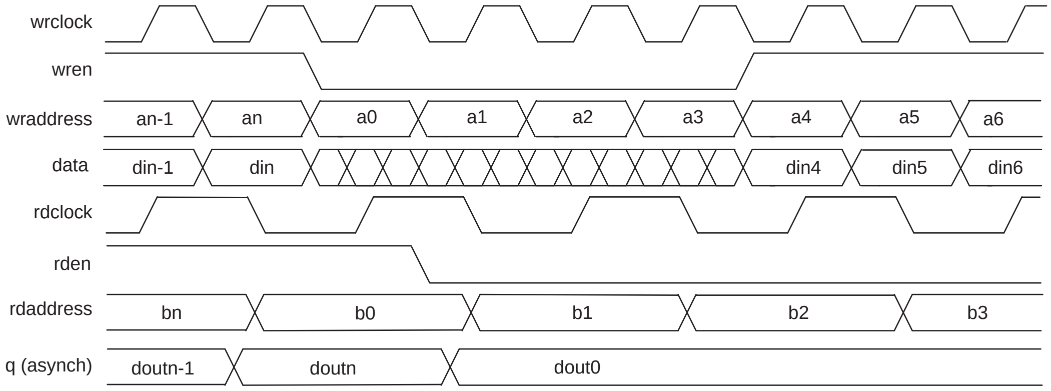
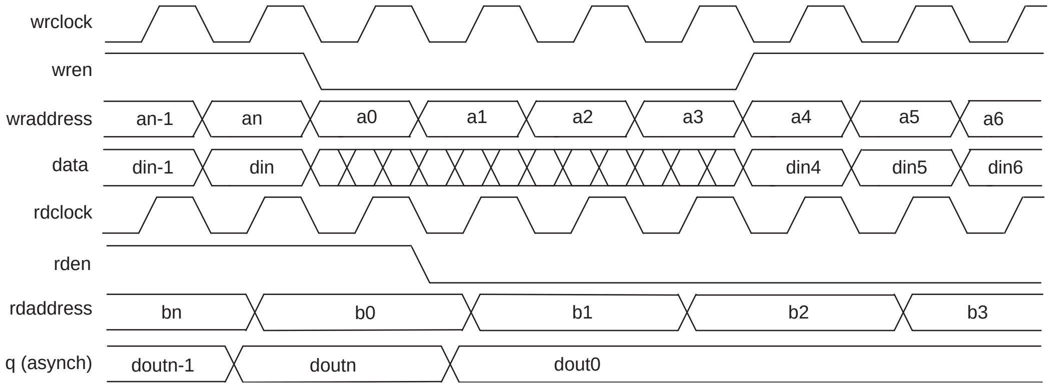


Figure 3–11 shows timing waveforms for read and write operations in mixed-port mode with unregistered outputs.

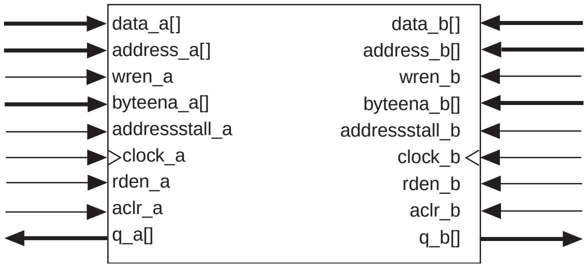
Figure 3–11. Stratix IV Mixed-Port Read-During-Write Timing Waveforms



True Dual-Port Mode

Stratix IV M9K and M144K blocks support true dual-port mode. Sometimes called bi-directional dual-port, this mode allows you to perform any combination of two port operations: two reads, two writes, or one read and one write at two different clock frequencies. Figure 3–12 shows the true dual-port RAM configuration.

Figure 3–12. Stratix IV True Dual-Port Memory *(Note 1)*



Note to Figure 3–12:

(1) True dual-port memory supports input/output clock mode in addition to the independent clock mode shown.

The widest bit configuration of the M9K and M144K blocks in true dual-port mode is as follows:

- 512 × 16-bit (×18-bit with parity) (M9K)
- 4K × 32-bit (×36-bit with parity) (M144K)

Wider configurations are unavailable because the number of output drivers is equivalent to the maximum bit width of the respective memory block. Because true dual-port RAM has outputs on two ports, its maximum width equals half of the total number of output drivers. Table 3–7 lists the possible M9K block mixed-port width configurations in true dual-port mode.

Table 3–7. Stratix IV M9K Block Mixed-Width Configuration (True Dual-Port Mode)

Read Port	Write Port						
	8K×1	4K×2	2K×4	1K×8	512×16	1K×9	512×18
8K×1	✓	✓	✓	✓	✓	—	—
4K×2	✓	✓	✓	✓	✓	—	—
2K×4	✓	✓	✓	✓	✓	—	—
1K×8	✓	✓	✓	✓	✓	—	—
512×16	✓	✓	✓	✓	✓	—	—
1K×9	—	—	—	—	—	✓	✓
512×18	—	—	—	—	—	✓	✓

Table 3–8 lists the possible M144K block mixed-port width configurations in true dual-port mode.

Table 3–8. Stratix IV M144K Block Mixed-Width Configurations (True Dual-Port Mode)

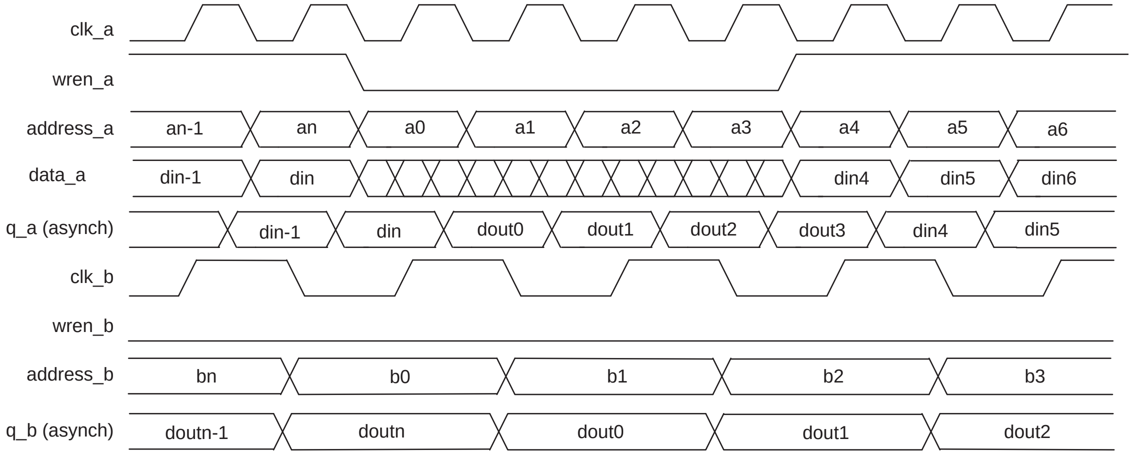
Read Port	Write Port					
	16K×8	8K×16	4K×32	16K×9	8K×18	4K×36
16K×8	✓	✓	✓	—	—	—
8K×16	✓	✓	✓	—	—	—
4K×32	✓	✓	✓	—	—	—
16K×9	—	—	—	✓	✓	✓
8K×18	—	—	—	✓	✓	✓
4K×36	—	—	—	✓	✓	✓

In true dual-port mode, M9K and M144K blocks support separate write-enable and read-enable signals. You can save power by keeping the read-enable signal low (inactive) when not reading. Read-during-write operations to the same address can either output new data at that location or old data. To choose the desired behavior, set the read-during-write behavior to either new data or old data in the RAM MegaWizard Plug-In Manager in the Quartus II software. See "Read During Write" on page 3–17 for more details about this behavior.

In true dual-port mode you can access any memory location at any time from either port. When accessing the same memory location from both ports, you must avoid possible write conflicts. A write conflict happens when you attempt to write to the same address location from both ports at the same time. This results in unknown data being stored to that address location. No conflict resolution circuitry is built into the Stratix IV TriMatrix memory blocks. You must handle address conflicts external to the RAM block.

Figure 3–13 shows true dual-port timing waveforms for the write operation at port A and read operation at port B, with the **Read-During-Write** behavior set to new data. Registering the RAM's outputs would simply delay the q outputs by one clock cycle.

Figure 3–13. Stratix IV True Dual-Port Timing Waveform



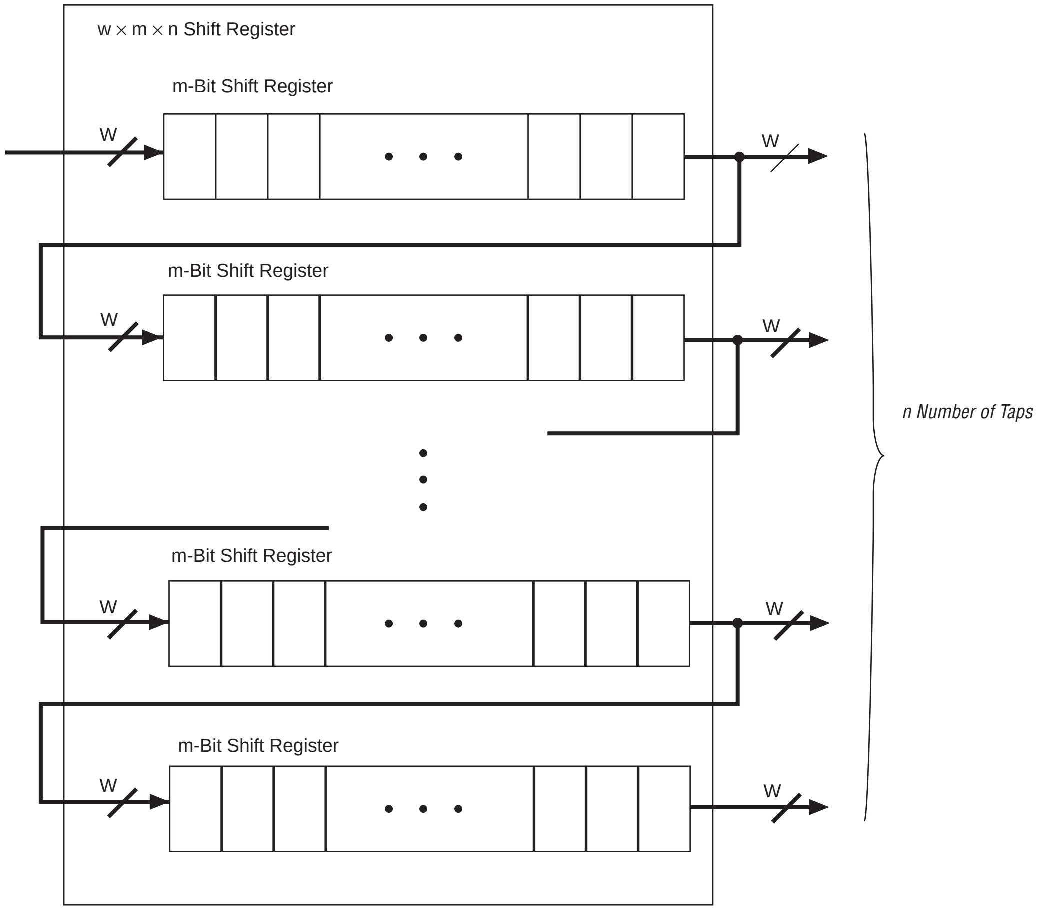
Shift-Register Mode

All Stratix IV memory blocks support shift register mode. Embedded memory block configurations can implement shift registers for digital signal processing (DSP) applications, such as finite impulse response (FIR) filters, pseudo-random number generators, multi-channel filtering, and auto- and cross-correlation functions. These and other DSP applications require local data storage, traditionally implemented with standard flip flops that quickly exhaust many logic cells for large shift registers. A more efficient alternative is to use embedded memory as a shift-register block, which saves logic cell and routing resources.

The size of a shift register (w × m × n) is determined by the input data width (w), the length of the taps (m), and the number of taps (n). You can cascade memory blocks to implement larger shift registers.

Figure 3–14 shows the TriMatrix memory block in shift-register mode.

Figure 3–14. Stratix IV Shift-Register Memory Configuration



ROM Mode

All Stratix IV TriMatrix memory blocks support ROM mode. A memory initialization file (**.mif**) initializes the ROM contents of these blocks. The address lines of the ROM are registered on M9K and M144K blocks, but can be unregistered on MLABs. The outputs can be registered or unregistered. Output registers can be asynchronously cleared. The ROM read operation is identical to the read operation in the single-port RAM configuration.

FIFO Mode

All TriMatrix memory blocks support FIFO mode. MLABs are ideal for designs with many small, shallow FIFO buffers. To implement FIFO buffers in your design, use the Quartus II software FIFO MegaWizard Plug-In Manager. Both single- and dual-clock (asynchronous) FIFO buffers are supported.

Refer to the *Single- and Dual-Clock FIFO Megafunctions User Guide* for more information about implementing FIFO buffers.

Clocking Modes

Stratix IV TriMatrix memory blocks support the following clocking modes:

- Independent
- Input/output
- Read/write
- Single clock

☞ Violating the setup or hold time on the memory block address registers could corrupt the memory contents. This applies to both read and write operations.

Table 3–9 shows the clocking mode versus memory mode support matrix.

Table 3–9. Stratix IV TriMatrix Memory Clock Modes

Clocking Mode	True Dual-Port Mode	Simple Dual-Port Mode	Single-Port Mode	ROM Mode	FIFO Mode
Independent	✓	—	—	✓	—
Input/output	✓	✓	✓	✓	—
Read/write	—	✓	—	—	✓
Single clock	✓	✓	✓	✓	✓

Independent Clock Mode

Stratix IV TriMatrix memory blocks can implement independent clock mode for true dual-port memories. In this mode, a separate clock is available for each port (clock A and clock B). Clock A controls all registers on the port A side, while clock B controls all registers on the port B side. Each port also supports independent clock enables for both port A and port B registers, respectively. Asynchronous clears are supported only for output latches and output registers on both ports.

Input/Output Clock Mode

Stratix IV TriMatrix memory blocks can implement input/output clock mode for true dual-port and simple dual-port memories. In this mode, an input clock controls all registers related to the data input to the memory block including data, address, byte enables, read enables, and write enables. An output clock controls the data output registers. Asynchronous clears are available on output latches and output registers only.

Read/Write Clock Mode

Stratix IV TriMatrix memory blocks can implement read/write clock mode for simple dual-port memories. In this mode, a write clock controls the data-input, write-address, and write-enable registers. Similarly, a read clock controls the data-output, read-address, and read-enable registers. The memory blocks support independent clock enables for both the read and write clocks. Asynchronous clears are available on data output latches and registers only.

Single Clock Mode

Stratix IV TriMatrix memory blocks can implement single-clock mode for true dual-port, simple dual-port, and single-port memories. In this mode, a single clock, together with a clock enable, is used to control all registers of the memory block. Asynchronous clears are available on output latches and output registers only.

Design Considerations

This section describes guidelines for designing with TriMatrix memory blocks.

Selecting TriMatrix Memory Blocks

The Quartus II software automatically partitions user-defined memory into embedded memory blocks by taking into account both speed and size constraints placed on your design. For example, the Quartus II software may spread memory out across multiple memory blocks when resources are available to increase the performance of the design. You can manually assign memory to a specific block size using the RAM MegaWizard Plug-In Manager.

MLABs can implement single-port SRAM through emulation using the Quartus II software. Emulation results in minimal additional logic resources being used. Because of the dual-purpose architecture of the MLAB, it only has data input registers and output registers in the block. MLABs gain input address registers and additional optional data output registers from adjacent ALMs by using register packing.

For more information about register packing, see the *Logic Array Blocks and Adaptive Logic Modules in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook*.

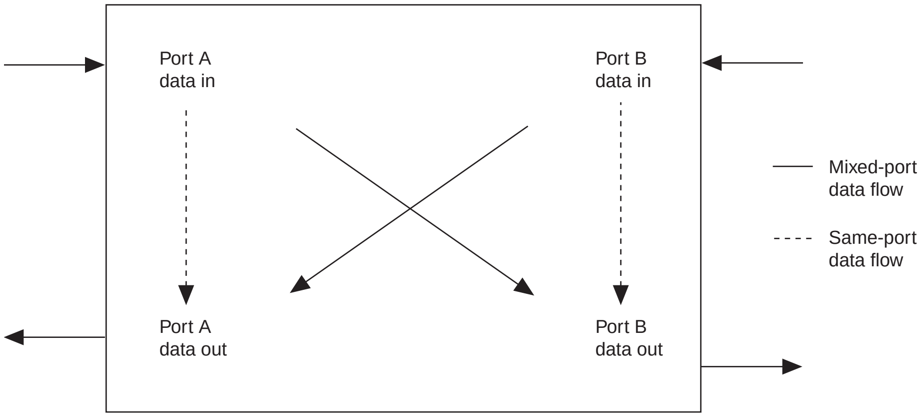
Conflict Resolution

When using the memory blocks in true dual-port mode, it is possible to attempt two write operations to the same memory location (address). Since no conflict resolution circuitry is built into the memory blocks, this results in unknown data being written to that location. Therefore, you must implement conflict resolution logic external to the memory block to avoid address conflicts.

Read During Write

You can customize the read-during-write behavior of the Stratix IV TriMatrix memory blocks to suit your design needs. Two types of read-during-write operations are available: same port and mixed port. Figure 3–15 shows the difference between the two types.

Figure 3–15. Stratix IV Read-During-Write Data Flow



Same-Port Read-During-Write Mode

This mode applies to either a single-port RAM or the same port of a true dual-port RAM. In same-port read-during-write mode, three output choices are available: new data mode (or flow-through), old data mode, or don’t care mode. In new data mode, the new data is available on the rising edge of the same clock cycle on which it was written. In old data mode, the RAM outputs reflect the old data at that address before the write operation proceeds. In don’t care mode, the RAM outputs don't care values for a read-during-write operation.

Figure 3–16 shows sample functional waveforms of same-port read-during-write behavior with new data.

Figure 3–16. Same Port Read-During Write: New Data Mode

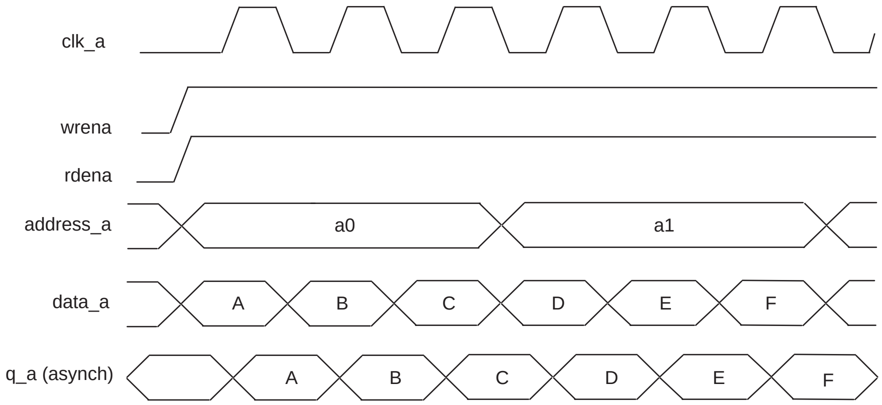
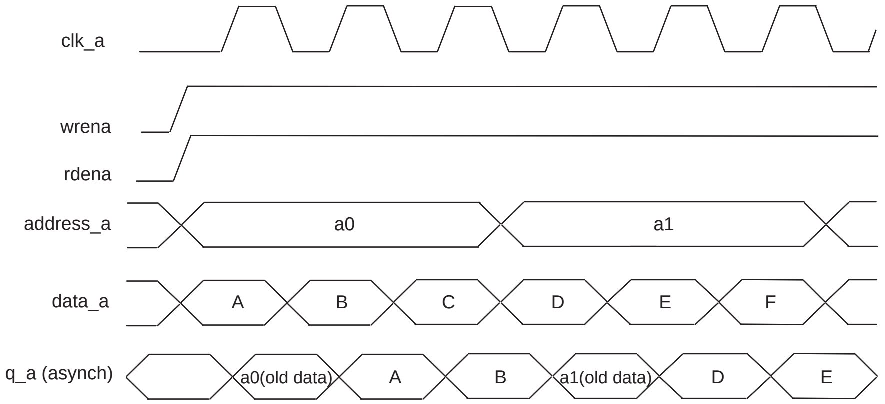


Figure 3–17 shows sample functional waveforms of same-port read-during-write behavior in old data mode.

Figure 3–17. Same Port Read-During-Write: Old Data Mode



Mixed-Port Read-During-Write Mode

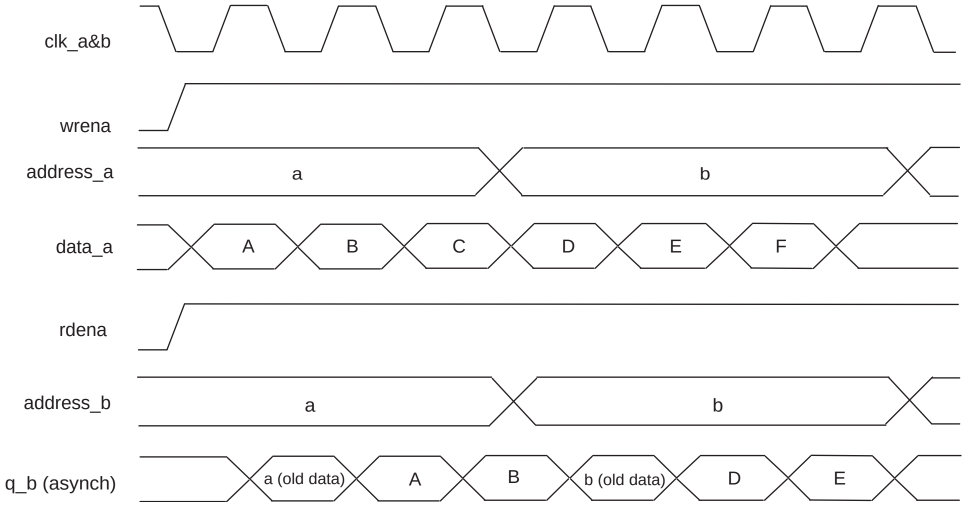
This mode applies to a RAM in simple or true dual-port mode which has one port reading from and the other port writing to the same address location with the same clock.

In this mode you also have two output choices: old data or don’t care. In old data mode, a read-during-write operation to different ports causes the RAM outputs to reflect the old data at that address location. In don’t care mode, the same operation results in a “don’t care” or “unknown” value on the RAM outputs.

Read-during-write behavior is controlled using the RAM MegaWizard Plug-In Manager. Refer to the *RAM Megafunction User Guide* for more details on how to implement the desired behavior.

Figure 3–18 shows a sample functional waveform of mixed-port read-during-write behavior for the old data mode. In don’t care mode, the old data shown in the figure is simply replaced with “don’t cares.”

Figure 3–18. Mixed Port Read During Write: Old Data Mode



Mixed-port read-during-write is not supported when two different clocks are used in a dual-port RAM. The output value is unknown during a dual-clock mixed-port read-during-write operation.

Power-Up Conditions and Memory Initialization

M9K and M144K memory block outputs power up to zero (cleared), regardless of whether the output registers are used or bypassed. MLABs power up to zero if output registers are used and power up reading the memory contents if output registers are not used. You must take this into consideration when designing logic that might evaluate the initial power-up values of the MLAB memory block. For Stratix IV devices, the Quartus II software initializes the RAM cells to 0 unless there is a **.mif** file specified.

All memory blocks support initialization using a **.mif** file. You can create **.mif** files in the Quartus II software and specify their use with the RAM MegaWizard Plug-In Manager when instantiating a memory in your design. Even if a memory is pre-initialized (for example, using a **.mif** file), it still powers up with its outputs cleared.

For more information about **.mif** files, refer to the *RAM Megafunction User Guide and the Quartus II Handbook*.

Power Management

Stratix IV memory block clock-enables allow you to control clocking of each memory block to reduce AC power consumption. Use the read-enable signal to ensure that read operations only occur when you need them to. If your design does not require read-during-write, you can reduce your power consumption by de-asserting the read-enable signal during write operations, or any period when no memory operations occur.

The Quartus II software automatically places any unused memory blocks in low power mode to reduce static power.

Conclusion

The Stratix IV TriMatrix embedded memory structure provides three different on-chip RAM block sizes to address your design needs. All memory blocks are fully customizable and can be cascaded to implement wider or deeper memories with minimal speed penalty.

You can independently configure each embedded memory block to be a single- or dual-port RAM, FIFO, ROM, or shift register using the Quartus II MegaWizard Plug-In Manager software.

Referenced Documents

This chapter references the following documents:

- *Logic Array Blocks and Adaptive Logic Modules in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Devices Handbook*
- *Quartus II Handbook*
- *RAM Megafunction User Guide*
- *Single- and Dual-Clock FIFO Megafunctions User Guide*

Document Revision History

Table 3–10 shows the revision history for this document.

Table 3–10. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	Updated "Power-Up Conditions and Memory Initialization" on page 3–20	—
May 2008 v1.0	Initial Release.	—

4. DSP Blocks in Stratix IV Devices

SIV51004-2.0

Introduction

The Stratix® IV family of devices have dedicated high-performance digital signal processing (DSP) blocks optimized for DSP applications. These DSP blocks of the Altera® Stratix device family are the fourth generation of hardwired, fixed function silicon blocks dedicated to maximizing signal processing capability, ease of use, and lowest silicon cost.

Many complex systems such as WiMAX, 3GPP WCDMA, high-performance computing (HPC), voice over Internet protocol (VoIP), H.264 video compression, medical imaging, and HDTV use sophisticated digital signal processing techniques, and this typically requires a large number of mathematical computations. Stratix IV devices are ideally suited as the DSP blocks consist of a combination of dedicated elements that perform multiplication, addition, subtraction, accumulation, summation, and dynamic shift operations. Along with the high-performance Stratix IV soft logic fabric and TriMatrix™ memory structures, you can configure these blocks to build sophisticated fixed-point and floating-point arithmetic functions. These can be manipulated easily to implement common, larger computationally intensive subsystems such as finite impulse response (FIR) filters, complex FIR filters, infinite impulse response (IIR) filters, fast Fourier transform (FFT) functions, and discrete cosine transform (DCT) functions.

This chapter contains the following sections:

- "Stratix IV DSP Block Overview"
- "Stratix IV Simplified DSP Operation" on page 4–3
- "Stratix IV Operational Modes Overview" on page 4–7
- "Stratix IV DSP Block Resource Descriptions" on page 4–8
- "Stratix IV Operational Mode Descriptions" on page 4–14
- "Software Support" on page 4–32

Stratix IV DSP Block Overview

Each Stratix IV device has two to seven columns of DSP blocks that implement multiplication, multiply-add, multiply-accumulate (MAC), and dynamic shift functions efficiently. Architectural highlights of the Stratix IV DSP block include:

- High-performance, power-optimized, fully registered, and pipelined multiplication operations
- Natively supported 9-bit, 12-bit, 18-bit, 36-bit wordlengths
- Natively supported 18-bit complex multiplications
- Efficiently supported floating-point arithmetic formats (24-bit for single precision and 53-bit for double precision)
- Signed and unsigned input support
- Built-in addition, subtraction and accumulation units to combine multiplication results efficiently
- Cascading 18-bit input bus to form tap-delay line for filtering applications
- Cascading 44-bit output bus to propagate output results from one block to the next block without external logic support
- Rich and flexible arithmetic rounding and saturation units
- Efficient barrel shifter support
- Loopback capability to support adaptive filtering

The number of DSP blocks for the Stratix IV device family is shown in Table 4–1.

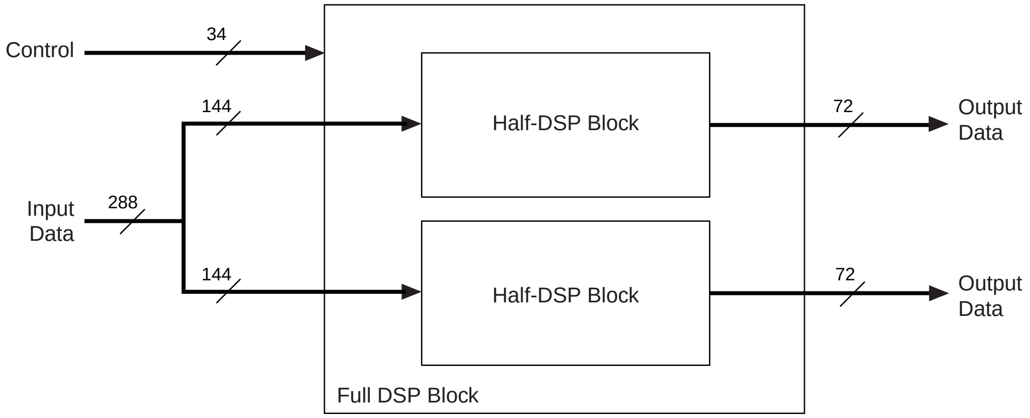
Table 4–1. Number of DSP Blocks in Stratix IV Devices

			Independent Input and Output Multiplication Operators					High Precision Multiplier Adder Mode	Four Multiplier Adder Mode
	Device	**DSP Blocks**	**9 × 9 Multipliers**	**12 × 12 Multipliers**	**18 × 18 Multipliers**	**18 × 18 Complex**	**36 × 36 Multipliers**	**18 × 36**	**18 × 18**
Stratix IV Enhanced	EP4SE110	64	512	384	256	128	128	256	512
	EP4SE230	161	1288	966	644	322	322	644	1288
	EP4SE290	104	832	624	416	208	208	416	832
	EP4SE360	130	1040	780	520	260	260	520	1040
	EP4SE530	128	1024	768	512	256	256	512	1024
	EP4SE680	170	1360	1020	680	340	340	680	1360
Stratix IV Transceiver	EP4SGX70	48	384	288	192	96	96	192	384
	EP4SGX110	64	512	384	256	128	128	256	512
	EP4SGX230	161	1288	966	644	322	322	644	1288
	EP4SGX290	104	832	624	416	208	208	416	832
	EP4SGX360	130	1040	780	520	260	260	520	1040
	EP4SGX530	128	1024	768	512	256	256	512	1024

Table 4–1 shows that the largest Stratix IV DSP-centric device provides up to 1360 18 × 18 multiplier functionality in the 36 × 36, complex 18 × 18, and summation modes.

Each DSP block occupies four LAB blocks in height and can be divided further into two half blocks that share some common clock signals, but are for all common purposes identical in functionality. The layout of each block is shown in Figure 4–1.

Figure 4–1. Overview of DSP Block Signals



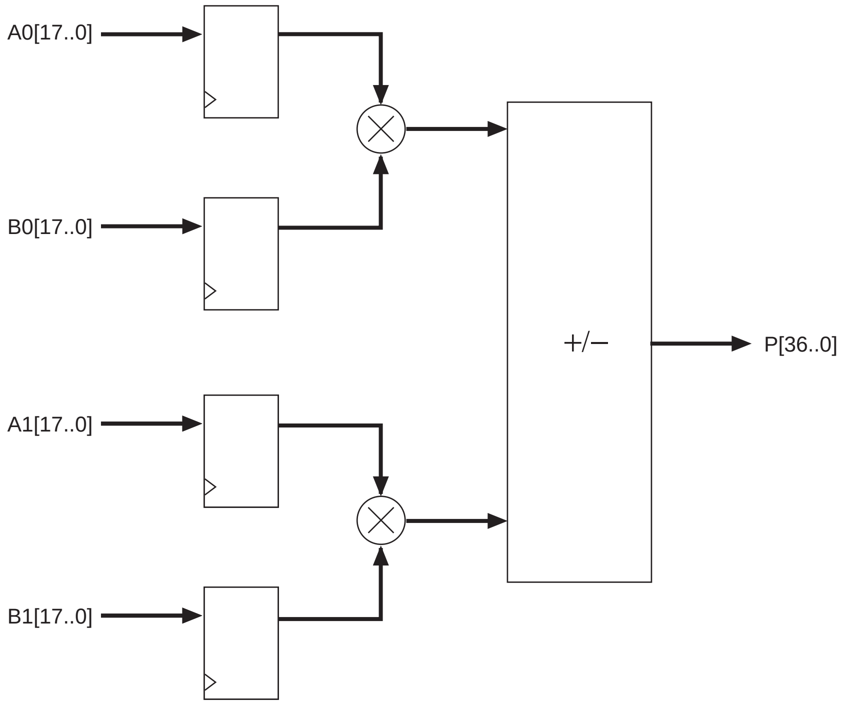
Stratix IV Simplified DSP Operation

In the Stratix IV devices, the fundamental building block is a pair of 18-bit × 18-bit multipliers followed by a first-stage 37-bit addition/subtraction unit, as shown in Equation 4–1 and Figure 4–2. Note that for all signed numbers, input and output data is represented in 2's complement format only.

Equation 4–1. Multiplier Equation

$$P[36..0] = A_0[17..0] \times B_0[17..0] \pm A_1[17..0] \times B_1[17..0]$$

Figure 4–2. Basic Two-Multiplier Adder Building Block



The structure shown in Figure 4–2 is useful for building more complex structures, such as complex multipliers and 36 × 36 multipliers, as described in later sections.

Each Stratix IV DSP block contains four two-multiplier adder units (two two-multiplier adder units per half block). Therefore, there are eight 18 × 18 multiplier functionalities per DSP block.

Following the two-multiplier adder units are the pipeline registers, the second-stage adders, and an output register stage. You can configure the second-stage adders to provide the following alternative functions per half block:

Equation 4–2. Four-Multiplier Adder Equation

$$Z[37..0] = P_0[36..0] + P_1[36..0]$$

Equation 4–3. Four-Multiplier Adder Equation (44-Bit Accumulation)

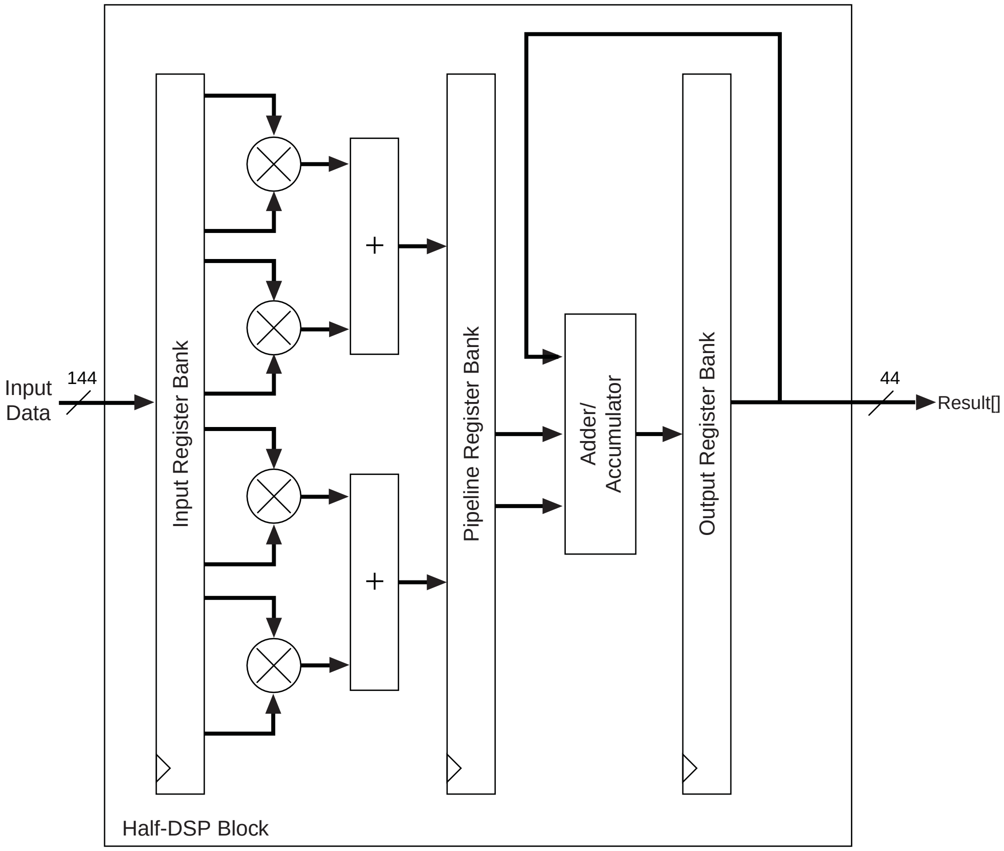
$$W_n[43..0] = W_{n-1}[43..0] \pm Z_n[37..0]$$

In these equations, *n* denotes sample time, and P[36..0] are the results from the two-multiplier adder units.

Equation 4–2 provides a sum of four 18-bit × 18-bit multiplication operations (four-multiplier adder), and Equation 4–3 provides a four 18-bit × 18-bit multiplication operation but with maximum of a 44-bit accumulation capability by feeding the output of the unit back to itself. This is shown in Figure 4–3.

You can bypass all register stages depending on which mode you select.

Figure 4–3. Four-Multiplier Adder and Accumulation Capability

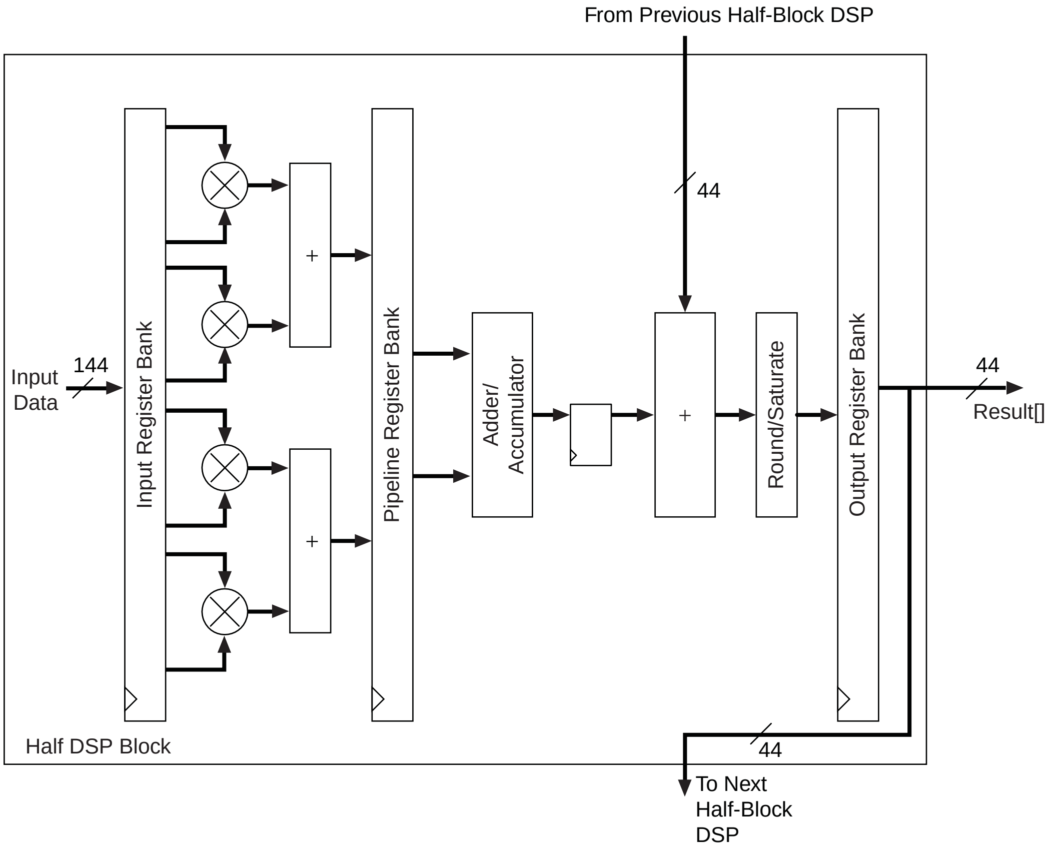


To support commonly found FIR-like structures efficiently, a major addition to the DSP block in Stratix IV is the ability to propagate the result of one half block to the next half block completely within the DSP block without additional soft logic overhead. This is achieved by the inclusion of a dedicated addition unit and routing that adds the 44-bit result of a previous half block with the 44-bit result of the current block. The 44-bit result is either fed to the next half block or out of the DSP block via the output register stage. This is shown in Figure 4–4. Detailed examples are described in later sections.

The combination of a fast, low-latency four-multiplier adder unit and the "chained cascade" capability of the output chaining adder provides an optimal FIR and vector multiplication capability.

To support single-channel type FIR filters efficiently, you can configure one of the multiplier input's registers to form a tap delay line input, saving resources and providing higher system performance.

Figure 4–4. Output Cascading Feature for FIR Structures

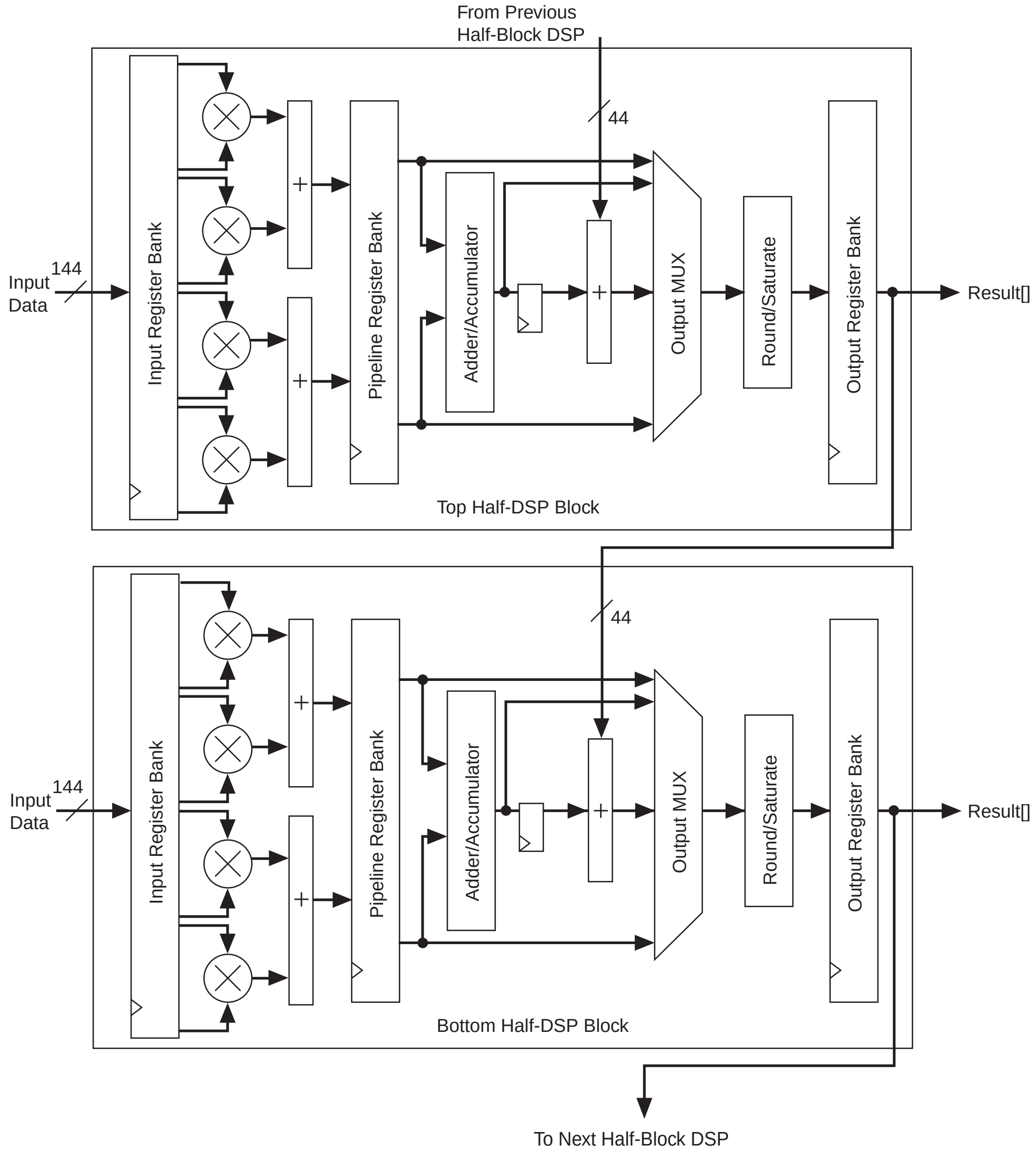


Also shown in Figure 4–4 is the optional rounding and saturation unit (RSU). This unit provides a rich set of commonly found arithmetic round and saturation functions used in signal processing.

In addition to the independent multipliers and sum modes, you can use the DSP blocks to perform shift operations. The DSP block can dynamically switch between logical shift left/right, arithmetic shift left/right, and rotation operation in one clock cycle.

A top-level view of the Stratix IV DSP block is shown in Figure 4–5.
A more detailed diagram is shown in Figure 4–6.

Figure 4–5. Stratix IV Full DSP Block Summary



Stratix IV Operational Modes Overview

Each Stratix IV DSP block can be used in one of five basic operational modes. Table 4–2 shows the five basic operational modes and the number of multipliers that can be implemented within a single DSP block, depending on the mode.

Table 4–2. Stratix IV DSP Block Operation Modes

Mode	Multiplier in Width	# of Mults	# per Block	Signed or Unsigned	RND, SAT	In Shift Register	Chainout Adder	1st Stage Add/Sub	2nd Stage Add/Acc
Independent Multiplier	9-bits	1	8	Both	No	No	No	—	—
	12-bits	1	6	Both	No	No	No	—	—
	18-bits	1	4	Both	Yes	Yes	No	—	—
	36-bits	1	2	Both	No	No	No	—	—
	Double	1	2	Both	No	No	No	—	—
Two-Multiplier Adder *(1)*	18-bits	2	4	Signed *(4)*	Yes	No	No	Both	—
Four-Multiplier Adder	18-bits	4	2	Both	Yes	Yes	Yes	Both	Add Only
Multiply Accumulate	18-bits	4	2	Both	Yes	Yes	Yes	Both	Both
Shift *(2)*	36-bits *(3)*	1	2	Both	No	No	—	—	—
High Precision Multiplier Adder	18x36	2	2	Both	No	No	No	—	Add Only

Notes to Table 4–2:

(1) This mode also supports the loopback mode. In loopback mode, the number of loopback multipliers per DSP block is two and the remaining multipliers can be used in regular two-multiplier adder mode.

(2) The dynamic shift mode supports arithmetic shift left, arithmetic shift right, logical shift left, logical shift right, and rotation operation.

(3) The dynamic shift mode operates on a 32-bit input vector but the multiplier width is configured as 36-bits.

(4) Unsigned value is also supported but you must make sure that the result can be contained within 36-bits.

The DSP block consists of two identical halves (top-half and bottom-half). Each half has four 18 × 18 multipliers.

The Quartus® II software includes megafunctions used to control the mode of operation of the multipliers. After making the appropriate parameter settings using the megafunction's MegaWizard® Plug-In Manager, the Quartus II software automatically configures the DSP block.

Stratix IV DSP blocks can operate in different modes simultaneously. Each half block is fully independent except for the sharing of the four `clock`, `ena`, and `aclr` signals. For example, you can break down a single DSP block to operate a 9 × 9 multiplier in one half block and an 18 × 18 two-multiplier adder in the other half block. This increases DSP block resource efficiency and allows you to implement more multipliers within a Stratix IV device. The Quartus II software automatically places multipliers that can share the same DSP block resources within the same block.

Stratix IV DSP Block Resource Descriptions

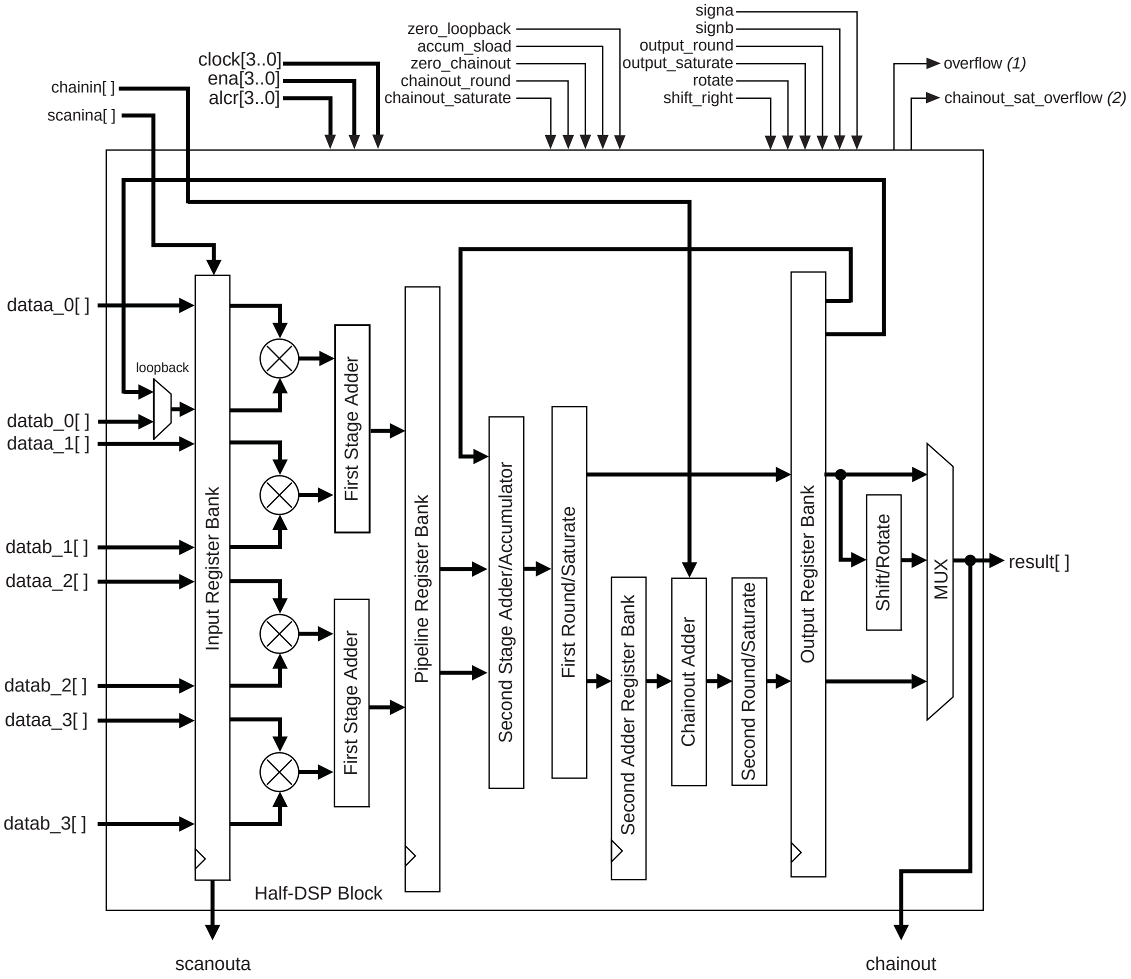
The DSP block consists of the following elements:

- Input register bank
- Four two-multiplier adders
- Pipeline register bank
- Two second-stage adders

- Four round and saturation logic units
- Second adder register and output register bank

A detailed overall architecture of the top half of the DSP block is shown in Figure 4–6, and a list of DSP block dynamic signals is shown in Table 4–9.

Figure 4–6. Half DSP Block Architecture



Notes to Figure 4–6:

(1) Block output for accumulator overflow, and saturate overflow.

(2) Block output for saturation overflow of chainout.

Input Registers

All of the DSP block registers are triggered by the positive edge of the clock signal and are cleared upon power up. Each multiplier operand can feed an input register or directly to the multiplier, bypassing the input registers. The following DSP block signals control the input registers within the DSP block:

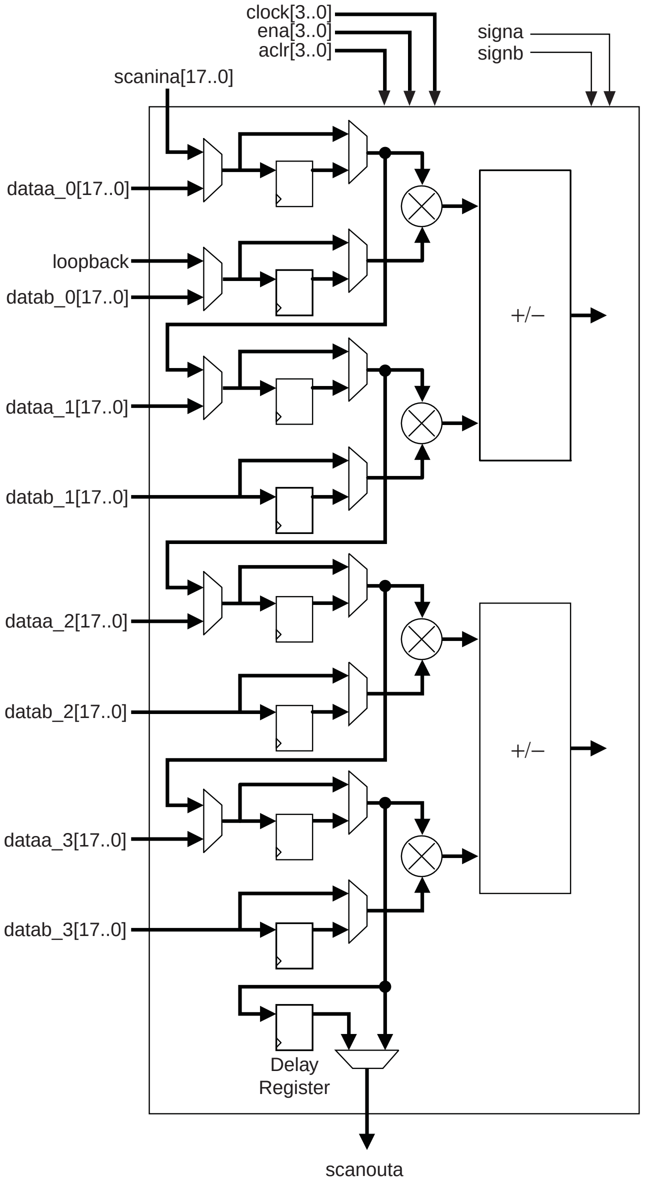
- `clock[3..0]`
- `ena[3..0]`

- `aclr[3..0]`

Every DSP block has nine 18-bit data input register banks per half DSP block. Every half DSP block has the option to use the eight data register banks as inputs to the four multipliers. The special ninth register bank is a delay register required by modes that use both the cascade and chainout features of the DSP block. Use the ninth register bank to balance the latency requirements when using the chained cascade feature.

A feature of the input register bank is to support a tap delay line. Therefore, the top leg of the multiplier input (A) can be driven from general routing or from the cascade chain, as shown in Figure 4–7 and a list of DSP block dynamic signals is shown in Table 4–9.

Figure 4–7. Input Register of Half DSP Block



You must select whether the A-input comes from general routing or from the cascade chain at compile time. In cascade mode, the dedicated shift outputs from one multiplier block directly feeds input registers of the adjacent multiplier below it (within the same half DSP block) or the first multiplier in the next half DSP block, to form an 8-tap shift register chain per DSP Block. The DSP block can increase the length of the shift register chain by cascading to the lower DSP blocks. The dedicated shift register chain spans a single column, but you can implement longer shift register chains requiring multiple columns using the regular FPGA routing resources.

Shift registers are useful in DSP functions such as FIR filters. When implementing 18 × 18 or smaller width multipliers, you do not need external logic to create the shift register chain because the input shift registers are internal to the DSP block. This implementation significantly reduces the logical element (LE) resources required, avoids routing congestion, and results in predictable timing.

The first multiplier in every half DSP block (top- and bottom-half) in Stratix IV devices has a mux for the first multiplier B-input (lower-leg input) register to select between general routing and loopback, as shown in Figure 4–6. In loopback mode, the most significant 18-bit registered outputs are connected as feedback to the multiplier input of the first top multiplier in each half DSP block. Loopback modes are used by recursive filters where the previous output is needed to compute the current output.

The loopback mode is described in detail in "Two-Multiplier Adder Sum Mode" on page 4–21.

Table 4–3 shows the summary of input register modes for the DSP block.

Table 4–3. Input Register Modes

Register Input Mode *(1)*	9 × 9	12 × 12	18 × 18	36 × 36	Double
Parallel input	✓	✓	✓	✓	✓
Shift register input *(2)*	—	—	✓	—	—
Loopback input *(3)*	—	—	✓	—	—

Notes to Table 4–3:

(1) The multiplier operand input wordlengths are statically configured at compile time.

(2) Available only on the A-operand.

(3) Only one loopback input is allowed per half block. See Figure 4–15 for details.

Multiplier and First-Stage Adder

The multiplier stage natively supports 9 × 9, 12 × 12, 18 × 18, or 36 × 36 multipliers. Other wordlengths are padded up to the nearest appropriate native wordlength; for example, 16 × 16 would be padded up to use 18 × 18. Refer to "Independent Multiplier Modes" on page 4–14 for more details. Depending on the data width of the multiplier, a single DSP block can perform many multiplications in parallel.

Each multiplier operand can be a unique signed or unsigned number. Two dynamic signals, `signa` and `signb`, control the representation of each operand, respectively. A `logic 1` value on the `signa/signb` signal indicates that `data A/data B` is a signed number; a `logic 0` value indicates an unsigned number. Table 4–4 shows the sign of the multiplication result for the various operand sign representations. The result of the multiplication is signed if any one of the operands is a signed value.

Table 4–4. Multiplier Sign Representation

Data A (signa Value)	Data B (signb Value)	Result
Unsigned (logic 0)	Unsigned (logic 0)	Unsigned
Unsigned (logic 0)	Signed (logic 1)	Signed
Signed (logic 1)	Unsigned (logic 0)	Signed
Signed (logic 1)	Signed (logic 1)	Signed

Each half block has its own `signa` and `signb` signal. Therefore, all of the `data A` inputs feeding the same DSP half block must have the same sign representation. Similarly, all of the `data B` inputs feeding the same DSP half block must have the same sign representation. The multiplier offers full precision regardless of the sign representation in all operational modes except for full precision 18 × 18 loopback and two-multiplier adder modes. Refer to "Two-Multiplier Adder Sum Mode" on page 4–21 for details.

When the `signa` and `signb` signals are unused, the Quartus II software sets the multiplier to perform unsigned multiplication by default.

The outputs of the multipliers are the only outputs that can feed into the first-stage adder, as shown in Figure 4–6. There are four first-stage adders in a DSP block (two adders per half DSP block). The first-stage adder block has the ability to perform addition and subtraction. The control signal for addition or subtraction is static and has to be configured upon compile time. The first-stage adders are used by the sum modes to compute the sum of two multipliers, 18 × 18-complex multipliers, and to perform the first stage of a 36 × 36 multiply and shift operation.

Depending on your specifications, the output of the first-stage adder has the option to feed into the pipeline registers, second-stage adder, round and saturation unit, or the output registers.

Pipeline Register Stage

The output from the first-stage adder can either feed or bypass the pipeline registers, as shown in Figure 4–6. Pipeline registers increase the DSP block's maximum performance (at the expense of extra cycles of latency), especially when using the subsequent DSP block stages. Pipeline registers split up the long signal path between the input-registers/multiplier/first-stage adder and the second-stage adder/round-and-saturation/output-registers, creating two shorter paths.

Second-Stage Adder

There are four individual 44-bit second-stage adders per DSP block (2 adders per half DSP block). You can configure the second-stage adders as follows:

- The final stage of a 36-bit multiplier
- A sum of four (18 × 18)
- An accumulator (44-bits maximum)
- A chained output summation (44-bits maximum)

☞ The chained-output adder can be used at the same time as a second-level adder in chained output summation mode.

The output of the second-stage adder has the option to go into the round and saturation logic unit or the output register.

☞ You cannot use the second-stage adder independently from the multiplier and first-stage adder.

Round and Saturation Stage

The round and saturation logic units are located at the output of the 44-bit second-stage adder (round logic unit followed by the saturation logic unit). There are two round and saturation logic units per half DSP block. The input to the round and saturation logic unit can come from one of the following stages:

- Output of the multiplier (independent multiply mode in 18 × 18)
- Output of the first-stage adder (two-multiplier adder)
- Output of the pipeline registers
- Output of the second-stage adder (four-multiplier adder, multiply-accumulate mode in 18 × 18)

These stages are discussed in detail in "Stratix IV Operational Mode Descriptions" on page 4–14.

The round and saturation logic unit is controlled by the dynamic round and saturate signals, respectively. A `logic 1` value on the round and/or saturate enables the round and/or saturate logic unit, respectively.

☞ You can use the round and saturation logic units together or independently.

Second Adder and Output Registers

The second adder register and output register banks are two banks of 44-bit registers that can also be combined to form larger 72-bit banks to support 36 × 36 output results.

The outputs of the different stages in the Stratix IV devices are routed to the output registers through an output selection unit. Depending on the operational mode of the DSP block, the output selection unit selects whether the outputs of the DSP blocks comes from the outputs of the multiplier block, first-stage adder, pipeline registers, second-stage adder, or the round and saturation logic unit. The output selection unit is set automatically by the software, based on the DSP block operational mode you specified, and has the option to either drive or bypass the output registers. The exception is when the block is used in shift mode, in which case the user dynamically controls the output-select mux directly.

When the DSP block is configured in "chained cascaded" output mode, both of the second-stage adders are used. The first one is used for performing a four-multiplier adder and the second is used for the chainout adder. The outputs of the four-multiplier adder are routed to the second-stage adder registers before they enter the chainout adder. The output of the chainout adder goes to the regular output register bank. Depending on the configuration, the chainout results can be routed to the input of the next half block's chainout adder input or to the general fabric (functioning as regular output registers). Refer to "Stratix IV Operational Mode Descriptions" on page 4–14 for details.

The second-stage and output registers are triggered by the positive edge of the clock signal and are cleared on power up.

The following DSP block signals control the output registers within the DSP block:

- `clock[3..0]`
- `ena[3..0]`
- `aclr[3..0]`

Stratix IV Operational Mode Descriptions

This section contains an explanation of different operational modes in Stratix IV devices.

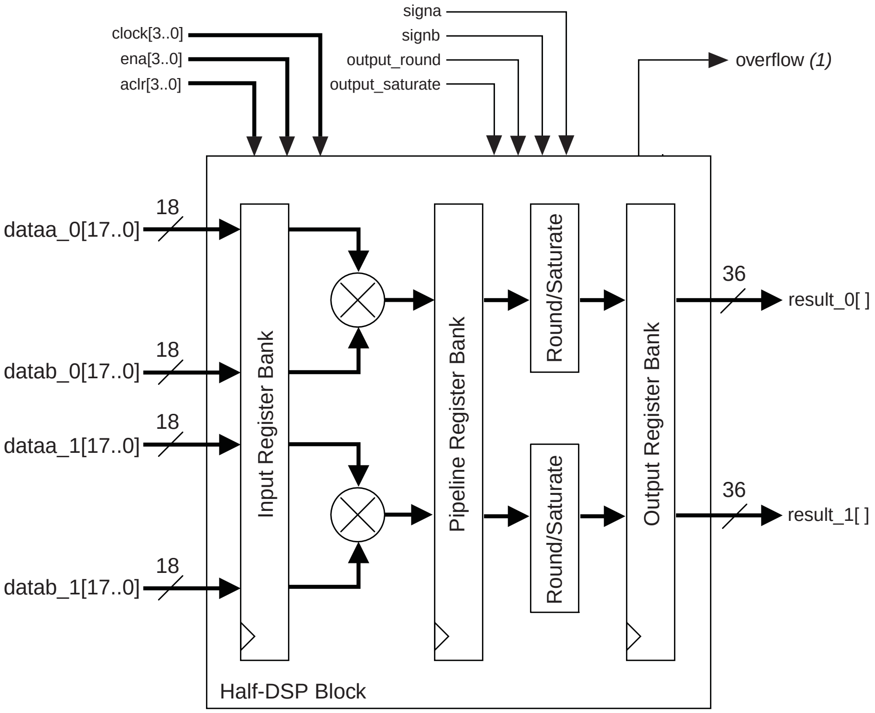
Independent Multiplier Modes

In independent input and output multiplier mode, the DSP block performs individual multiplication operations for general-purpose multipliers.

9-, 12-, and 18-Bit Multiplier

You can configure each DSP block multiplier for 9-, 12-, or 18-bit multiplication. A single DSP block can support up to eight individual 9 × 9 multipliers, six 12 × 12 multipliers, or up to four individual 18 × 18 multipliers. For operand widths up to 9 bits, a 9 × 9 multiplier is implemented. For operand widths from 10 to 12 bits, a 12 × 12 multiplier is implemented, and for operand widths from 13 to 18 bits, an 18 × 18 multiplier is implemented. This is done by the Quartus II software by zero-padding the least significant bits (LSBs). Figure 4–8, Figure 4–9, and Figure 4–10 show the DSP block in the independent multiplier operation mode, and a list of DSP block dynamic signals is shown in Table 4–9.

Figure 4–8. 18-Bit Independent Multiplier Mode Shown for Half DSP Block



Note to Figure 4–8:

(1) Block output for accumulator overflow, and saturate overflow.

Figure 4–9. 12-Bit Independent Multiplier Mode Shown for Half DSP Block

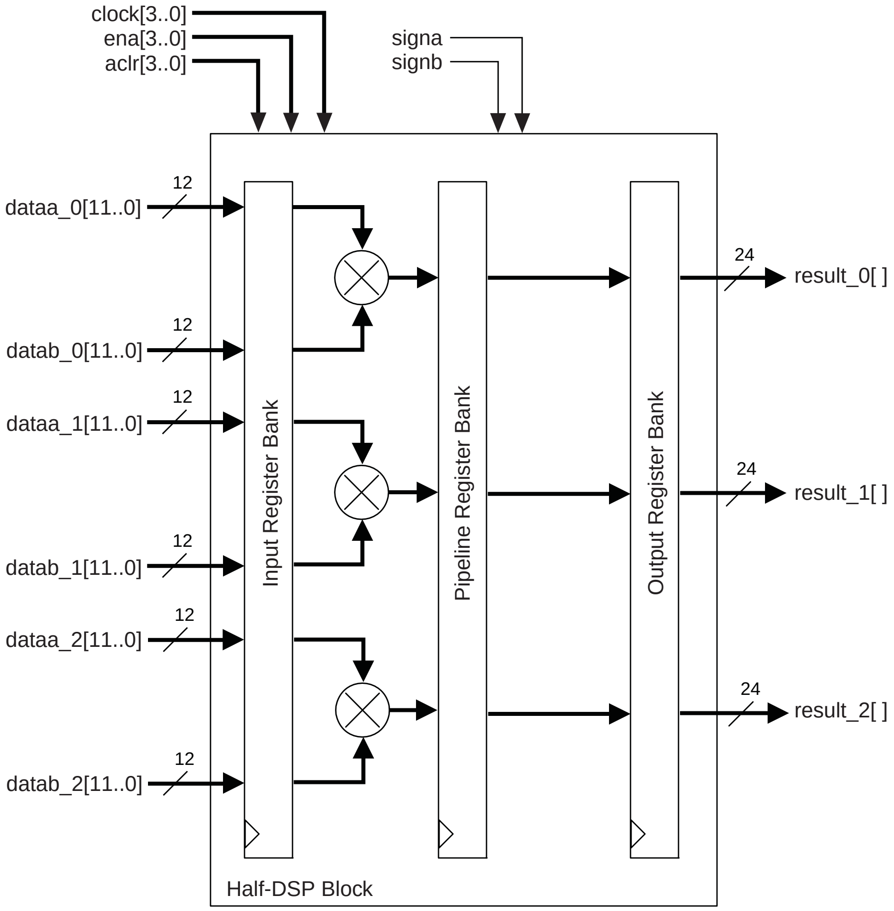
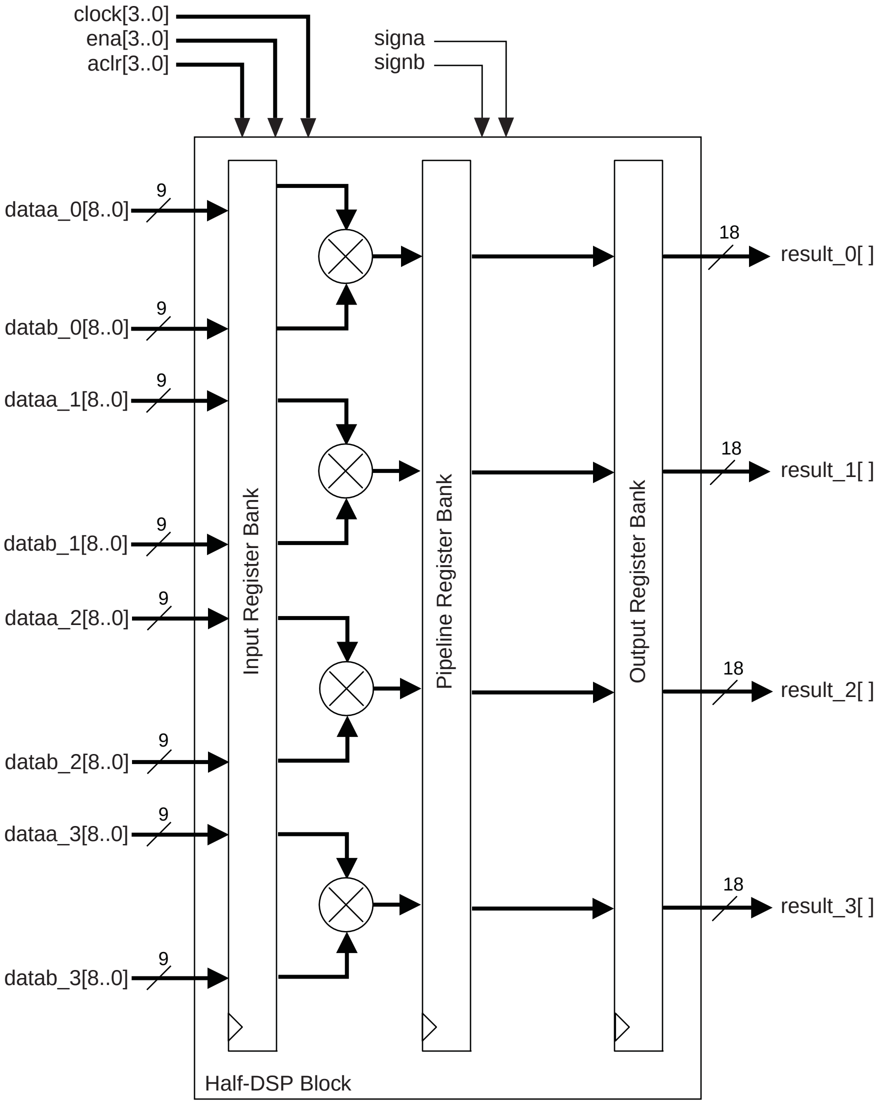


Figure 4–10. 9-Bit Independent Multiplier Mode Shown for a Half Block



The multiplier operands can accept signed integers, unsigned integers, or a combination of both. You can change the `signa` and `signb` signals dynamically and they can be registered in the DSP block. Additionally, the multiplier inputs and result can be registered independently. You can use the pipeline registers within the DSP block to pipeline the multiplier result, increasing the performance of the DSP block.

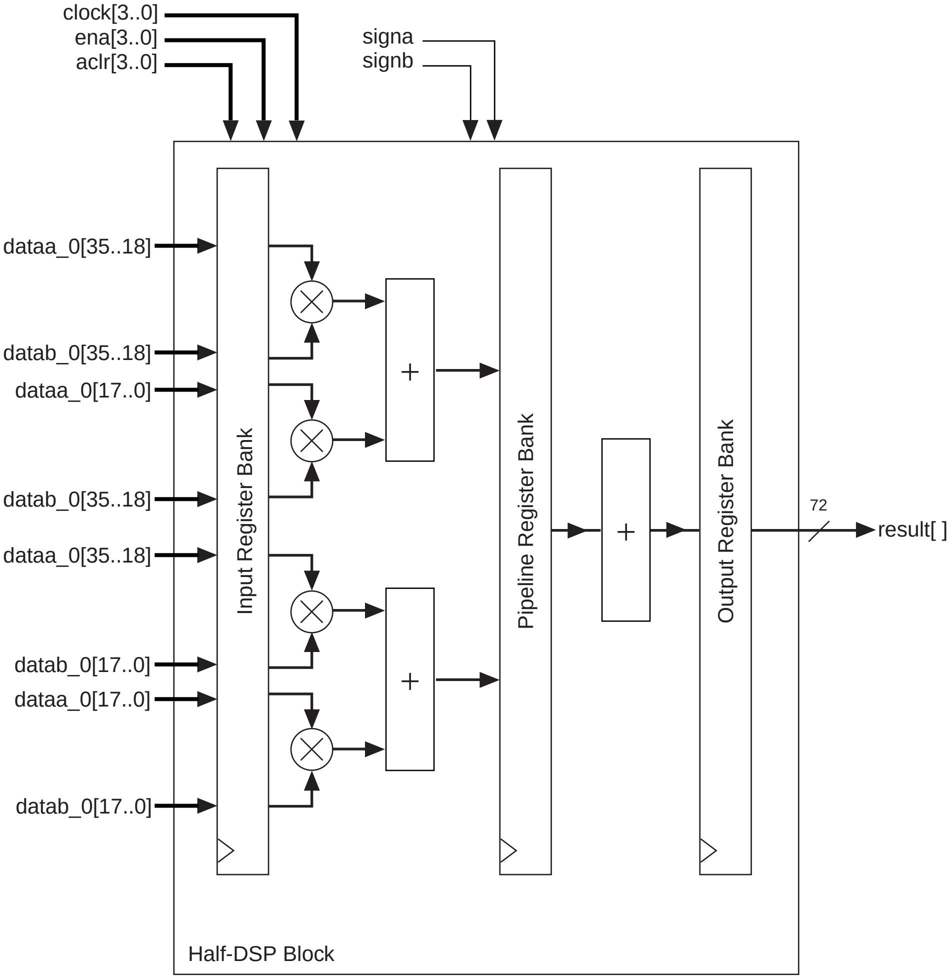
☞ The round and saturation logic unit is supported for the 18-bit independent multiplier mode only.

36-Bit Multiplier

You can efficiently construct a 36 × 36 multiplier using four 18 × 18 multipliers. This simplification fits conveniently into one half DSP block, and is implemented in the DSP block automatically by selecting the 36 × 36 mode. Stratix IV devices can have up to two 36-bit multipliers per DSP block (one 36-bit multiplier per half DSP block). The 36-bit multiplier is also under the independent multiplier mode but uses the entire half DSP block, including the dedicated hardware logic after the pipeline registers to implement the 36 × 36 bit multiplication operation, as shown in Figure 4–11.

The 36-bit multiplier is useful for applications requiring more than 18-bit precision; for example, for the mantissa multiplication portion of single precision and extended single precision floating-point arithmetic applications.

Figure 4–11. 36-Bit Independent Multiplier Mode Shown for Half DSP Block



Double Multiplier

The Stratix IV DSP block can be configured to efficiently support a signed or unsigned 54 × 54 bit multiplier that is required to compute the mantissa portion of an IEEE double precision floating point multiplication. A 54 × 54 bit multiplier can be built using basic 18 × 18 multipliers, shifters and adders. In order to efficiently utilize the Stratix IV DSP block's built in shifters and adders, a special double mode (partial 54 × 54 multiplier) is available that is a slight modification to the basic 36 × 36 multiplier mode, as shown in Figure 4–12 and Figure 4–13.

Figure 4–12. Double Mode Shown for Half DSP Block

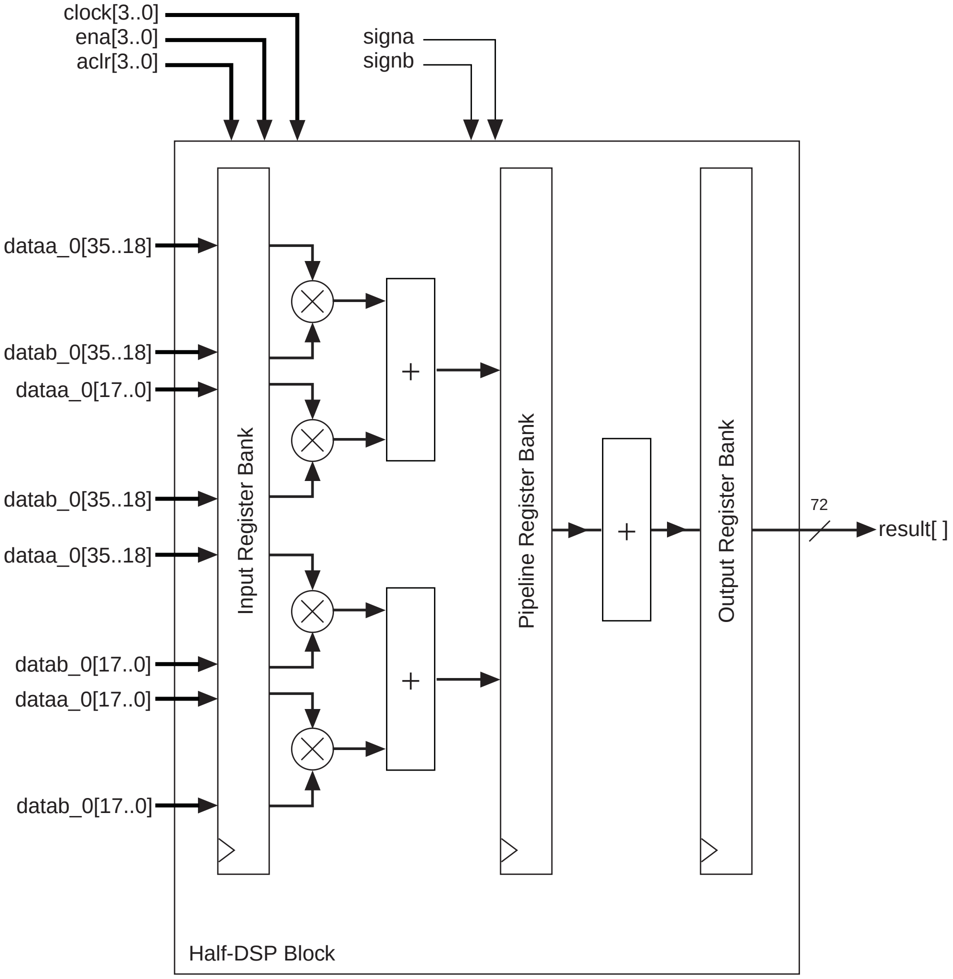
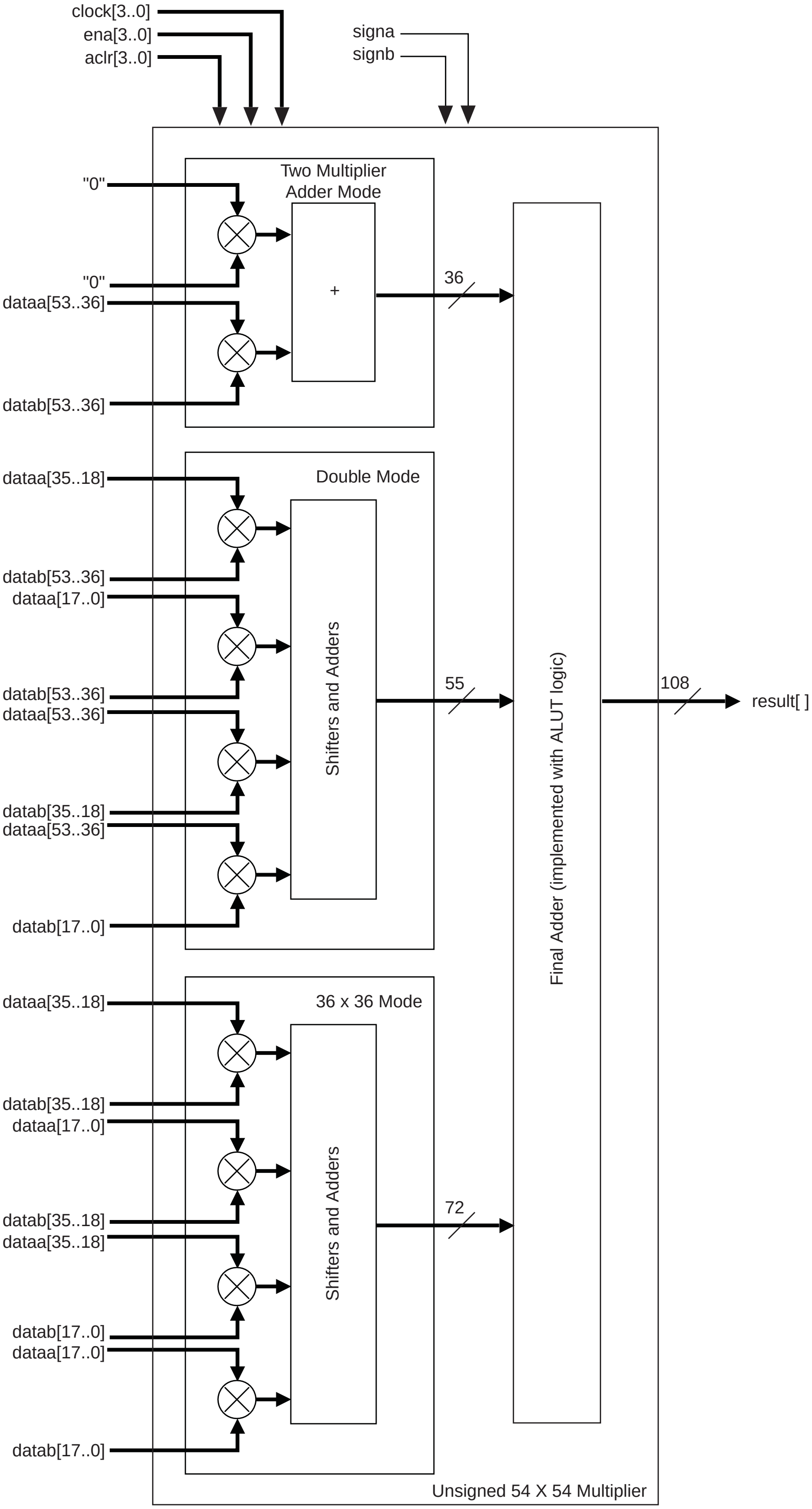


Figure 4–13. Unsigned 54 × 54 Multiplier



Two-Multiplier Adder Sum Mode

In the two-multiplier adder configuration, the DSP block can implement four 18-bit two-multiplier adders (2 two-multiplier adders per half DSP block). You can configure the adders to take the sum or difference of two multiplier outputs. Summation or subtraction has to be selected at compile time. The two-multiplier adder function is useful for applications such as FFTs, complex FIR, and IIR filters. Figure 4–14 shows the DSP block configured in the two-multiplier adder mode.

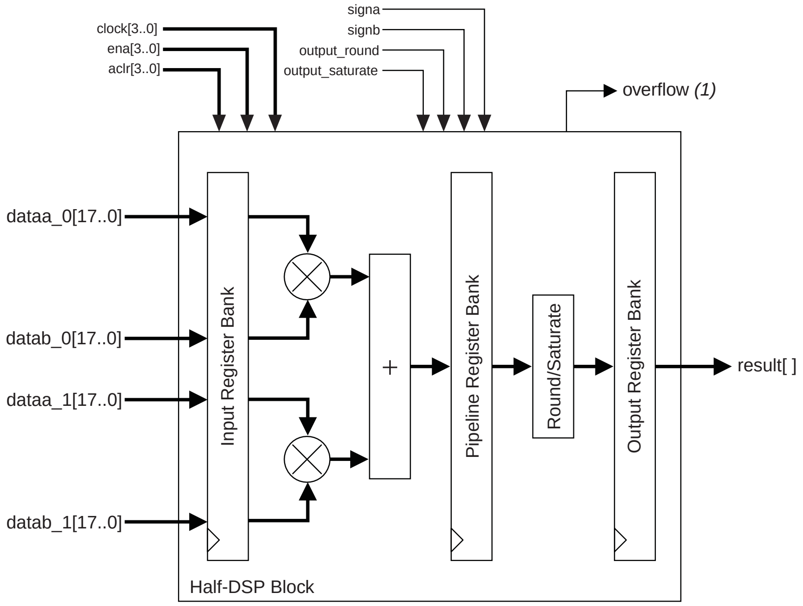
The loopback mode is the other sub-feature of the two-multiplier adder mode. Figure 4–15 shows the DSP block configured in the loopback mode. This mode takes the 36-bit summation result of the two multipliers and feeds back the most significant 18-bits to the input. The lower 18-bits are discarded. You have the option to disable or zero-out the loopback data by using the dynamic `zero_loopback` signal. A `logic 1` value on the `zero_loopback` signal selects the `zeroed` data or disables the looped back data, while a `logic 0` selects the looped back data.

☞ The option to use the loopback mode or the general two-multiplier adder mode must be selected at compile time.

For the two-multiplier adder mode, if all the inputs are full 18-bit and unsigned, the result will require 37 bits. As the output data width in two-multiplier adder mode is limited to 36 bits, this 37-bit output requirement is not allowed. Any other combination that does not violate the 36-bit maximum result is permitted; for example, two 16 × 16 signed two-multiplier adders is valid.

The two-multiplier adder mode supports the round and saturation logic unit. You can use the pipeline registers and output registers within the DSP block to pipeline the multiplier-adder result, increasing the performance of the DSP block.

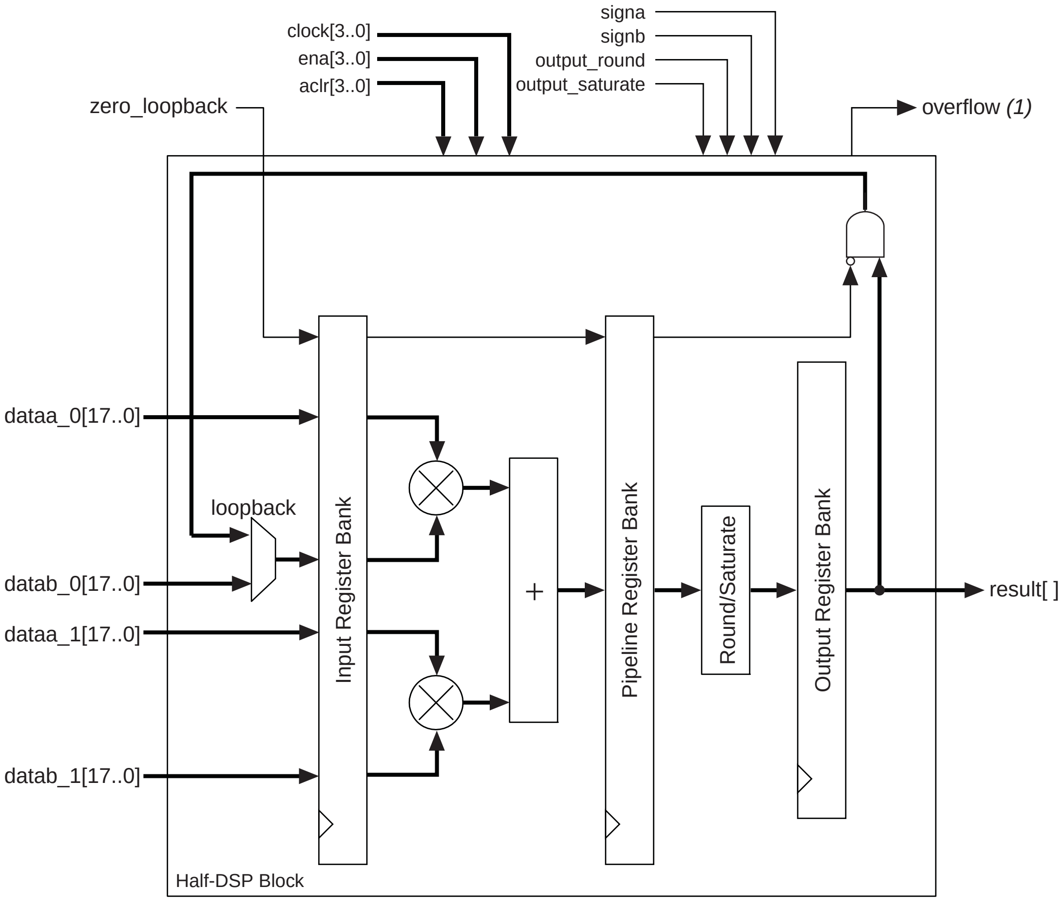
Figure 4–14. Two-Multiplier Adder Mode Shown for Half DSP Block



Note to Figure 4–14:

(1) Block output for accumulator overflow, and saturate overflow.

Figure 4–15. Loopback Mode for Half DSP Block



Note to Figure 4–15:

(1) Block output for accumulator overflow, and saturate overflow.

18 x 18 Complex Multiply

You can configure the DSP block when used in two-multiplier adder mode to implement complex multipliers using the two-multiplier adder mode. A single half DSP block can implement one 18-bit complex multiplier.

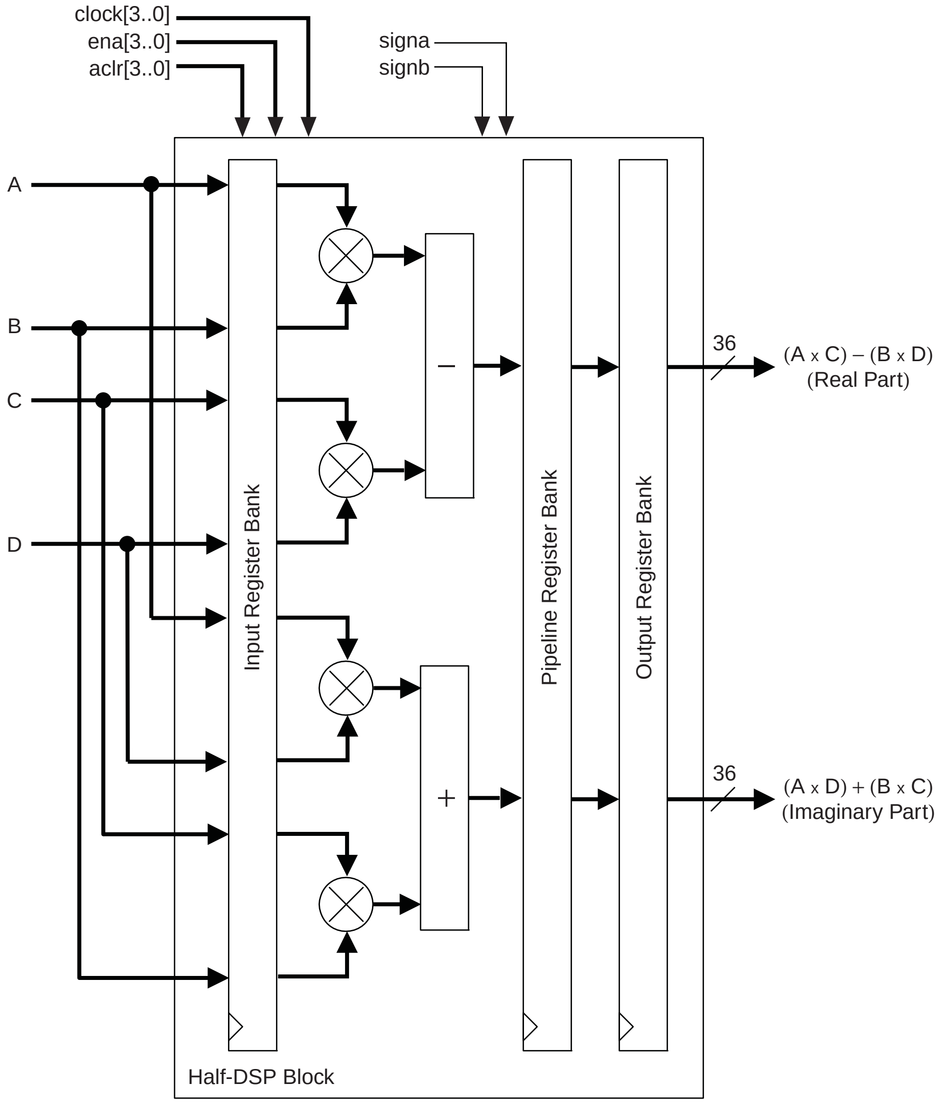
A complex multiplication can be written as shown in Equation 4–4.

Equation 4–4. Complex Multiplication Equation

$$(a + jb) \times (c + jd) = ((a \times c)) - (b \times d)) + j((a \times d) + (b \times c))$$

To implement this complex multiplication within the DSP block, the real part ((a × c) – (b × d)) is implemented using two multipliers feeding one subtractor block while the imaginary part ((a × d) + (b × c)) is implemented using another two multipliers feeding an adder block. Figure 4–16 shows an 18-bit complex multiplication. This mode automatically assumes all inputs are using signed numbers.

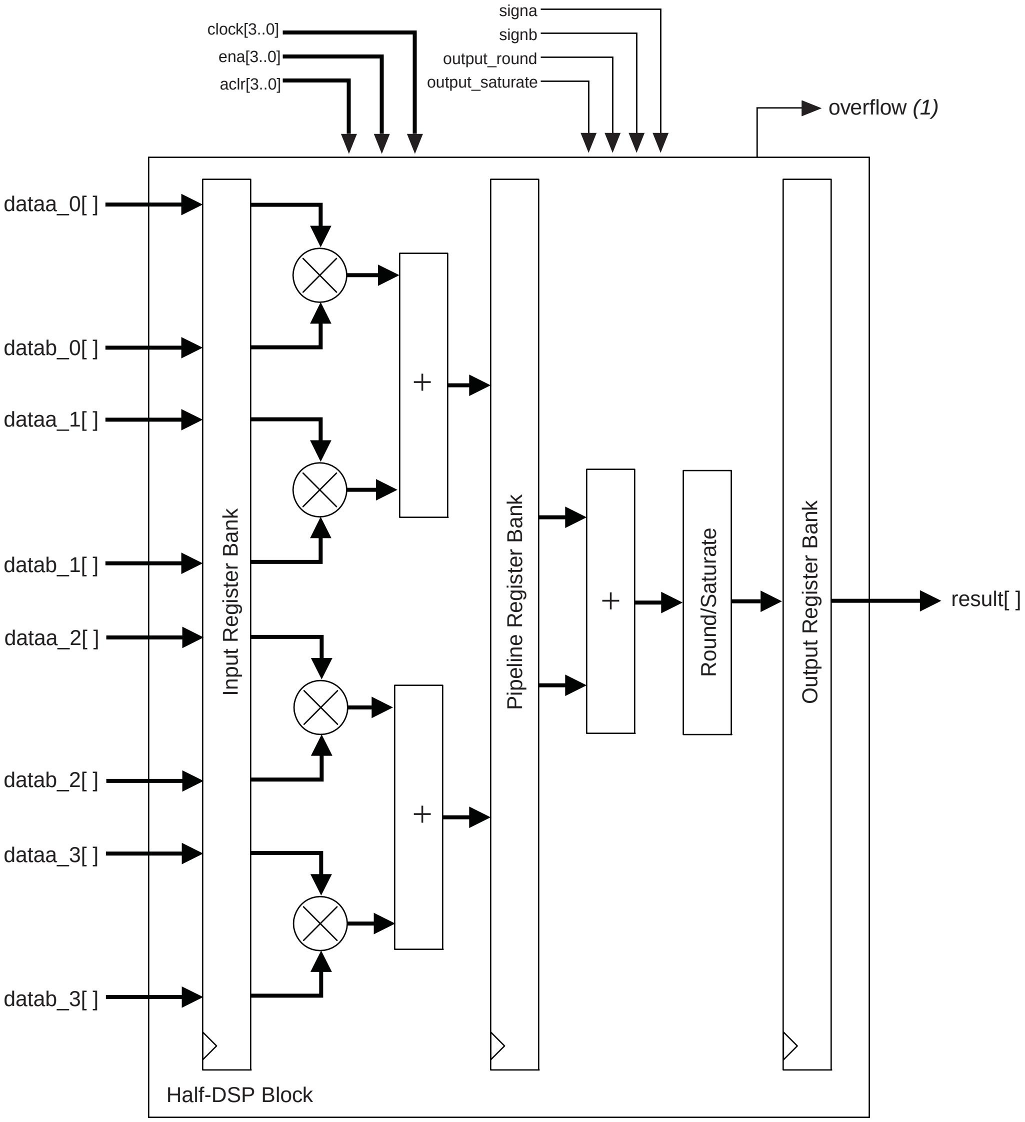
Figure 4–16. Complex Multiplier Using Two-Multiplier Adder Mode



Four-Multiplier Adder

In the four-multiplier adder configuration shown in Figure 4–17, the DSP block can implement two four-multiplier adders (one four-multiplier adder per half DSP block). These modes are useful for implementing one-dimensional and two-dimensional filtering applications. The four-multiplier adder is performed in two addition stages. The outputs of two of the four multipliers are initially summed in the two first-stage adder blocks. The results of these two adder blocks are then summed in the second-stage adder block to produce the final four-multiplier adder result, as shown by Equation 4–2 and Equation 4–3.

Figure 4–17. Four-Multiplier Adder Mode Shown for Half DSP Block



Note to Figure 4–17:

(1) Block output for accumulator overflow, and saturate overflow.

The four-multiplier adder mode supports the round and saturation logic unit. You can use the pipeline registers and output registers within the DSP block to pipeline the multiplier-adder result, increasing the performance of the DSP block.

High-Precision Multiplier Adder Mode

In the high-precision multiplier adder configuration, shown in Figure 4–18, the DSP block can implement 2 two-multiplier adders, with multiplier precision of 18 x 36 (one two-multiplier adder per DSP half block). This mode is useful in filtering or FFT applications where a data path greater than 18 bits is required, yet 18 bits is sufficient for the coefficient precision. This can occur in cases where that data has a high dynamic range. If the coefficients are fixed, as in FFT and most filter applications, the precision of 18 bits provide a dynamic range over 100 dB, if the largest coefficient is normalized to the maximum 18 bit representation.

In these situations, the data path can be up to 36 bits, allowing sufficient capacity for bit growth or gain changes in the signal source without loss of precision. This mode is also extremely useful in single precision block floating point applications.

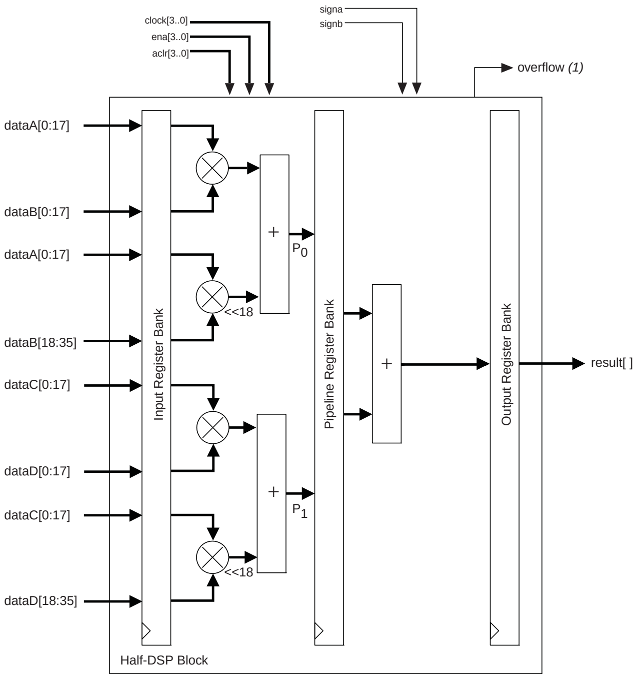
The high-precision multiplier adder is performed in two stages. The 18 × 36 multiply is divided into two 18 × 18 multipliers. The multiplier with the LSB of the data source is performed unsigned, while the multiplier with the most significant bit (MSB) of the data source can be signed or unsigned. The latter multiplier has its result left shifted by 18 bits prior to the first adder stage, creating an effective 18 x 36 multiplier. The results of these two adder blocks are then summed in the second stage adder block to produce the final result:

$Z[54..0] = P_0[53..0] + P_1[53..0]$

where:

$P_0 = A[17..0] \times B[35..0]$ and $P_1 = C[17..0] \times D[35..0]$

Figure 4–18. High-Precision Multiplier Adder Configuration *(Note 1)*



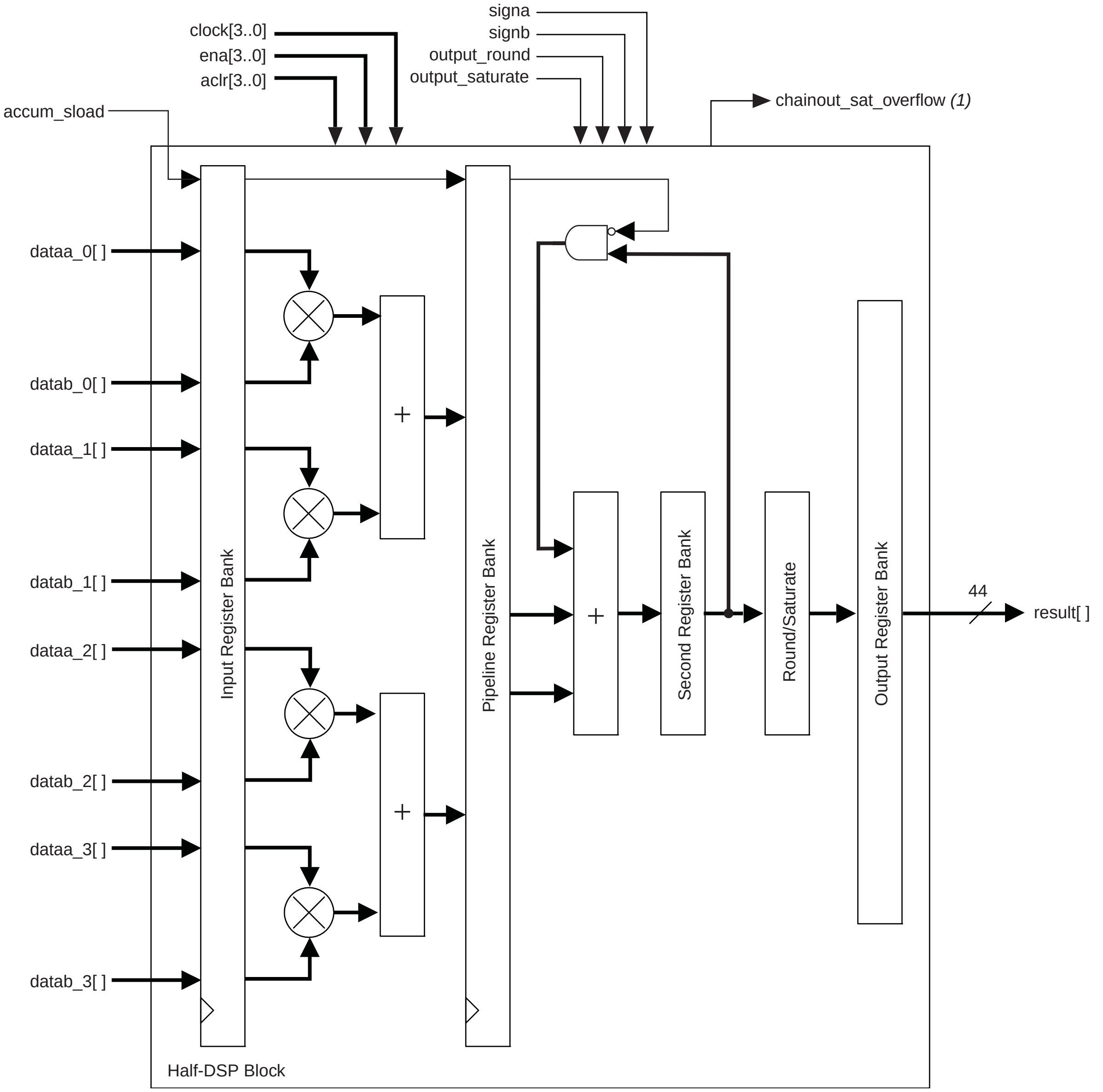
Note to Figure 4–18:

(1) Block output for accumulator overflow, and saturate overflow.

Multiply Accumulate Mode

In multiply accumulate mode, the second-stage adder is configured as a 44-bit accumulator or subtractor. The output of the DSP block is looped back to the second-stage adder and added or subtracted with the two outputs of the first-stage adder block according to Equation 4–3. Figure 4–19 shows the DSP block configured to operate in multiply accumulate mode.

Figure 4–19. Multiply Accumulate Mode Shown for Half DSP Block



Note to Figure 4–19:

(1) Block output for saturation overflow of chainout.

A single DSP block can implement up to two independent 44-bit accumulators.

The dynamic `accum_sload` control signal is used to clear the accumulation. A `logic 1` value on the `accum_sload` signal synchronously loads the accumulator with the multiplier result only, while a `logic 0` enables accumulation by adding or subtracting the output of the DSP block (accumulator feedback) to the output of the multiplier and first-stage adder.

☞ The control signal for the accumulator and subtractor is static and therefore has to be configured at compile time.

This mode supports the round and saturation logic unit as it is configured as an 18-bit multiplier accumulator. You can use the pipeline registers and output registers within the DSP block to increase the performance of the DSP block.

Shift Modes

Stratix IV devices support the following shift modes for 32-bit input only:

- Arithmetic shift left, `ASL[N]`
- Arithmetic shift right, `ASR[32-N]`
- Logical shift left, `LSL[N]`
- Logical shift right, `LSR[32-N]`
- 32-bit rotator or barrel shifter, `ROT[N]`

☞ You can switch the shift mode between these modes using the dynamic rotate and shift control signals.

The shift mode in a Stratix IV device can be easily used by the soft embedded processor such as Nios® II to perform the dynamic shift and rotate operation. Figure 4–20 shows the shift mode configuration.

The shift mode makes use of the available multipliers to logically or arithmetically shift left, right, or rotate the desired 32-bit data. The DSP block is configured like the independent 36-bit multiplier mode to perform the shift mode operations.

The arithmetic shift right requires a signed input vector. During an arithmetic shift right, the sign is extended to fill the MSB of the 32-bit vector. The logical shift right uses an unsigned input vector. During a logical shift right, zeros are padded in the most significant bits shifting the 32-bit vector to the right. The barrel shifter uses unsigned input vector and implements a rotation function on a 32-bit word length.

Two control signals `rotate` and `shift_right` together with the `signa` and `signb` signals, determining the shifting operation. Examples of shift operations are shown in Table 4–5.

Figure 4–20. Shift Operation Mode Shown for Half DSP Block

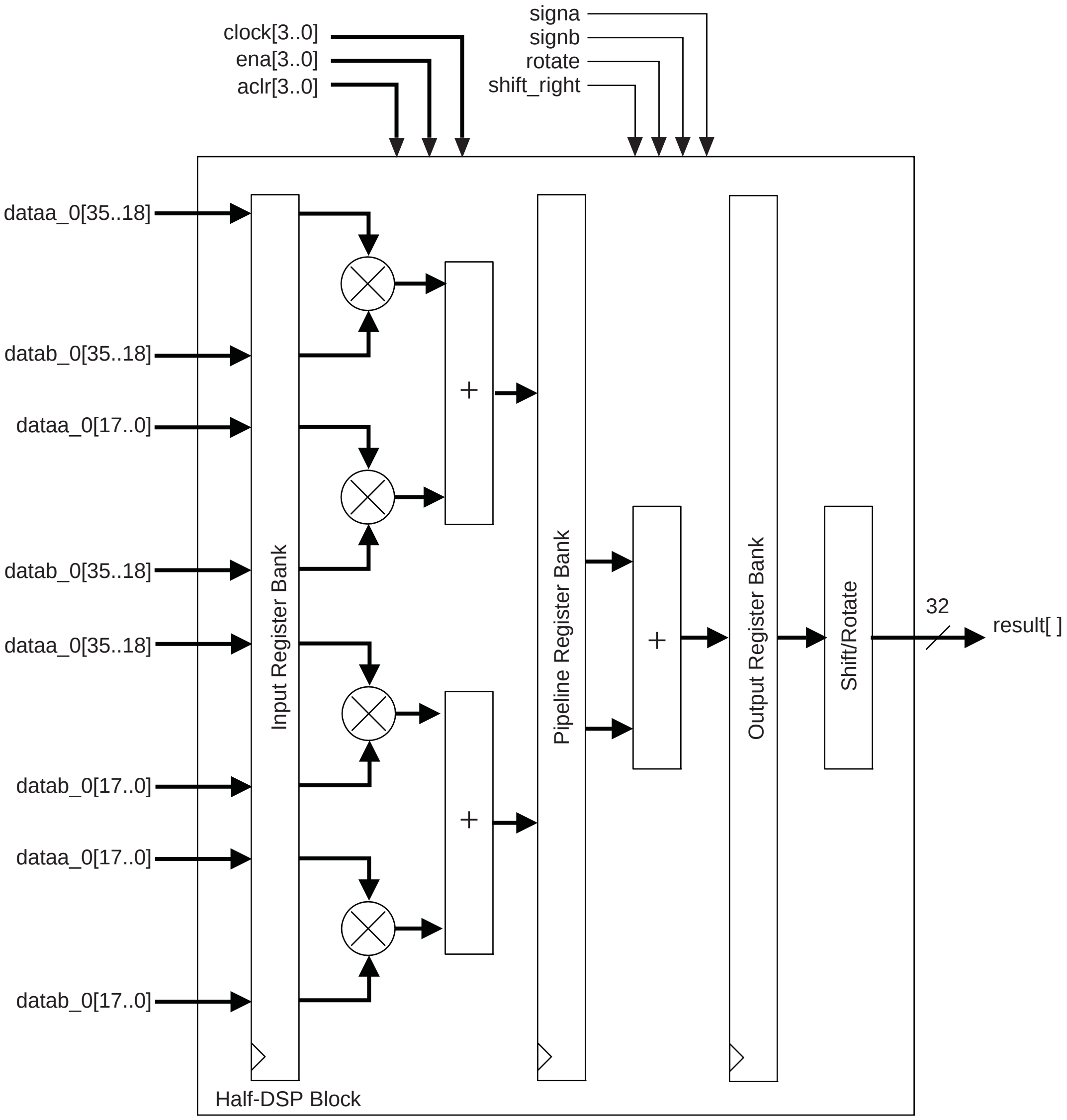


Table 4–5. Examples of Shift Operations

Example	Signa	Signb	Shift	Rotate	A-input	B-input	Result
Logical Shift Left `LSL[N]`	Unsigned	Unsigned	0	0	0xAABBCCDD	0x0000100	0xBBCCDD00
Logical Shift Right `LSR[32-N]`	Unsigned	Unsigned	1	0	0xAABBCCDD	0x0000100	0x000000AA
Arithmetic Shift Left `ASL[N]`	Signed	Unsigned	0	0	0xAABBCCDD	0x0000100	0xBBCCDD00
Arithmetic Shift Right `ASR 32-N]`	Signed	Unsigned	1	0	0xAABBCCDD	0x0000100	0xFFFFFFAA
Rotation `ROT[N]`	Unsigned	Unsigned	0	1	0xAABBCCDD	0x0000100	0xBBCCDDAA

Rounding and Saturation Mode

Round and saturation functions are often required in DSP arithmetic. Rounding is used to limit bit growth and its side effects and saturation is used to reduce overflow and underflow side effects.

Two rounding modes are supported in Stratix IV devices:

- Round-to-nearest-integer mode
- Round-to-nearest-even mode

You must select one of the two options at compile time.

Round-to-nearest-integer provides the biased rounding support and is the simplest form of rounding commonly used in DSP arithmetic. The round-to-nearest-even method provides unbiased rounding support and is used where DC offsets are a concern. Table 4–6 shows how round-to-nearest-even works. Examples of the difference between the two modes are shown in Table 4–7. In this example, a 6-bit input is rounded to 4 bits. You can observe from Table 4–7 that the main difference between the two rounding options is when the residue bits are exactly halfway between its nearest two integers and the LSB is zero (even).

Table 4–6. Example of Round-To-Nearest-Even Mode

6- to 4-bits Rounding	Odd/Even (Integer)	Fractional	Add to Integer	Result
010111	x	> 0.5 (11)	1	0110
001101	x	< 0.5 (01)	0	0011
001010	Even (0010)	= 0.5 (10)	0	0010
001110	Odd (0011)	= 0.5 (10)	1	0100
110111	x	> 0.5 (11)	1	1110
101101	x	< 0.5 (01)	0	1011
110110	Odd (1101)	= 0.5 (10)	1	1110
110010	Even (1100)	= 0.5 (10)	0	1100

Table 4–7. Comparison of Round-to-Nearest-Integer and Round-to-Nearest-Even

Round-To-Nearest-Integer	Round-To-Nearest-Even
010111 ⇨ 0110	010111 ⇨ 0110
001101 ⇨ 0011	001101 ⇨ 0011
001010 ⇨ 0011	001010 ⇨ 0010
001110 ⇨ 0100	001110 ⇨ 0100
110111 ⇨ 1110	110111 ⇨ 1110
101101 ⇨ 1011	101101 ⇨ 1011
110110 ⇨ 1110	110110 ⇨ 1110
110010 ⇨ 1101	110010 ⇨ 1100

Two saturation modes are supported in Stratix IV:

- Asymmetric saturation mode
- Symmetric saturation mode

You must select one of the two options at compile time.

In 2’s complement format, the maximum negative number that can be represented is $-2^{(n-1)}$ while the maximum positive number is $2^{(n-1)} - 1$. Symmetrical saturation will limit the maximum negative number to $-2^{(n-1)} + 1$. For example, for 32 bits:

- Asymmetric 32-bit saturation: Max = 0x7FFFFFFF, Min = 0x80000000
- Symmetric 32-bit saturation: Max = 0x7FFFFFFF, Min = 0x80000001

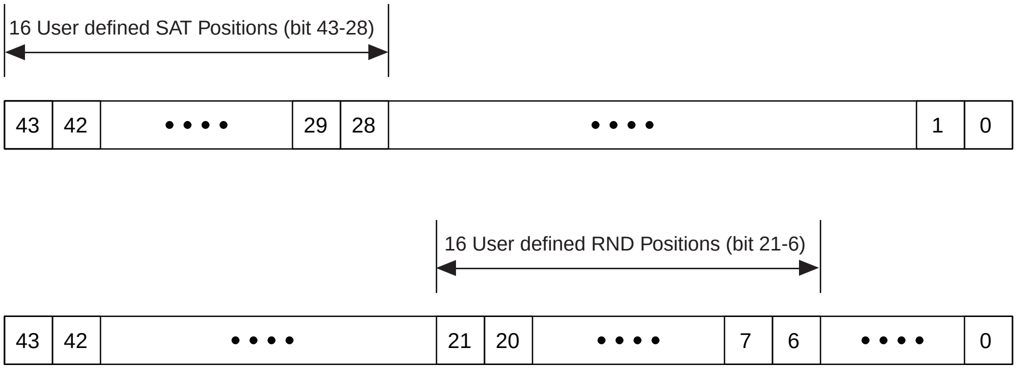
Table 4–8 shows how the saturation works. In this example, a 44-bit input is saturated to 36-bits.

Table 4–8. Examples of Saturation

44 to 36 Bits Saturation	Symmetric SAT Result	Asymmetric SAT Result
5926AC01342h	7FFFFFFFFh	7FFFFFFFFh
ADA38D2210h	800000001h	800000000h

Stratix IV devices have up to 16 configurable bit positions out of the 44-bit bus (`[43:0]`) for the round and saturate logic unit providing higher flexibility. You must select the 16 configurable bit positions at compile time. These 16-bit positions are located at bits `[21:6]` for rounding and `[43:28]` for saturation, as shown in Figure 4–21.

Figure 4–21. Round and Saturation Locations


16 User defined SAT Positions (bit 43-28)
43
42
• • • •
29
28
• • • •
1
0
16 User defined RND Positions (bit 21-6)
43
42
• • • •
21
20
• • • •
7
6
• • • •
0

For symmetric saturation, the RND bit position is also used to determine where the LSP for the saturated data is located.

You can use the rounding and saturation function described above in regular supported multiplication operations as specified in Table 4–2. However, for accumulation type operations, the following convention is used:

The functionality of the round logic unit is in the format of:

Result = RND[S(A × B)], when used for an accumulation type of operation.

Likewise, the functionality of the saturation logic unit is in the format of:

Result = SAT[S(A × B)], when used for an accumulation type of operation.

If both the round and saturation logic units are used for an accumulation type of operation, the format is:

Result = SAT[RND[S(A × B)]]

DSP Block Control Signals

The Stratix IV DSP block is configured using a set of static and dynamic signals. The DSP block dynamic signals are user configurable and can be set to toggled or not at run time. This list of dynamic signals is shown in Table 4–9 for the DSP block.

Table 4–9. DSP Block Dynamic Signals (Part 1 of 2)

Signal Name	Function	Count
■ `signa` ■ `signb`	Signed/unsigned control for all multipliers and adders. `signa` for “multiplicand” input bus to `dataa[17:0]` each multiplier. `signb` for “multiplier” input bus `datab[17:0]` to each multiplier. `signa` = 1, `signb` = 1 for signed-signed multiplication `signa` = 1, `signb` = 0 for signed-unsigned multiplication `signa` = 0, `signb` = 1 for unsigned-signed multiplication `signa` = 0, `signb` = 0 for unsigned-unsigned multiplication	2
`output_round`	Round control for first stage round/saturation block. `output_round` = 1 for rounding on multiply output `output_round` = 0 for normal multiply output	1
`chainout_round`	Round control for second stage round/saturation block. `chainout_round` = 1 for rounding on multiply output `chainout_round` = 0 for normal multiply output	1
`output_saturate`	Saturation control for first stage round/saturation block for Q-format multiply. If both rounding and saturation is enabled, saturation is done on the rounded result. `output_saturate` = 1 for saturation support `output_saturate` = 0 for no saturation support	1
`chainout_saturate`	Saturation control for second stage round/saturation block for Q-format multiply. If both rounding and saturation’s enabled, saturation is done on the rounded result. `chainout_saturate` = 1 for saturation support `chainout_saturate` = 0 for no saturation support	1
`accum_sload`	Dynamically specifies whether the accumulator value is zero. `accum_sload` = 0, accumulation input is from the output registers `accum_sload` = 1, accumulation input is set to be zero	1
`zero_chainout`	Dynamically specifies whether the chainout value is zero.	1
`zero_loopback`	Dynamically specifies whether the loopback value is zero.	1
`rotate`	`rotation` = 1, rotation feature is enabled	1
`shift_right`	`shift_right` = 1, shift right feature is enabled	1
	Total Signals per half block	11

Table 4–9. DSP Block Dynamic Signals (Part 2 of 2)

Signal Name	Function	Count

`clock0`
`clock1`
`clock2`
`clock3`
	DSP-block-wide clock signals	4

`ena0`
`ena1`
`ena2`
`ena3`
	Input and Pipeline Register enable signals	4

`aclr0`
`aclr1`
`aclr2`
`aclr3`
	DSP block-wide asynchronous clear signals (active low).	4

	Total Count per Full Block	34

Software Support

Altera provides two distinct methods for implementing various modes of the DSP block in a design: instantiation and inference. Both methods use the following Quartus II megafunctions:

- `lpm_mult`
- `altmult_add`
- `altmult_accum`
- `altfp_mult`

You can instantiate the megafunctions in the Quartus II software to use the DSP block. Alternatively, with inference, you can create an HDL design and synthesize it using a third-party synthesis tool (such as LeonardoSpectrum, Synplify, or Quartus II Native Synthesis) that infers the appropriate megafunction by recognizing multipliers, multiplier adders, multiplier accumulators, and shift functions. Using either method, the Quartus II software maps the functionality to the DSP blocks during compilation.

Refer to the *Quartus II Software Help* for instructions about using the megafunctions and the MegaWizard Plug-In Manager.

For more information, refer to the *Synthesis* section in volume 1 of the *Quartus II Development Software Handbook*.

Conclusion

The Stratix IV device DSP blocks are optimized to support DSP applications requiring high data throughput, such as FIR filters, IIR filters, FFT functions, and encoders. These DSP blocks are flexible and can be configured to implement one of several operational modes to suit a particular application. The built-in shift register chain, multipliers, and adders/subtractors minimize the amount of external logic required to implement these functions, resulting in efficient resource utilization and improved performance and data throughput for DSP applications. The Quartus II software, used with the LeonardoSpectrum and Synplify software, provide a complete and easy-to-use flow for implementing these multiplier functions in the DSP blocks.

Referenced Documents

This chapter references the following documents:

- *Synthesis* section in volume 1 of the *Quartus II Development Software Handbook*

Document Revision History

Table 4–10 shows the revision history for this document.

Table 4–10. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	■ Updated Table 4–2 ■ Updated Figure 4–16 ■ Updated Figure 4–18	—
May 2008 v1.0	Initial Release.	—

5. Clock Networks and PLLs in Stratix IV Devices

SIV51005-2.0

Introduction

Stratix® IV devices provide a hierarchical clock structure and multiple PLLs with advanced features. The large number of clocking resources, in combination with the clock synthesis precision provided by the PLLs, provides a complete clock management solution. Stratix IV devices provide dedicated global clock networks (GCLKs), regional clock networks (RCLKs), and periphery clock networks (PCLKs). These clocks are organized into a hierarchical clock structure that provides up to 236 unique clock domains (16 GCLK + 88 RCLK + 132 PCLK) within the Stratix IV device and allows up to 71 unique GCLK, RCLK, and PCLK clock sources (16 GCLK + 22 RCLK + 33 PCLK) per device quadrant. The Altera® Quartus® II software compiler automatically turns off clock networks not used in the design, thereby reducing the overall power consumption of the device.

Stratix IV devices deliver abundant PLL resources with up to 12 PLLs per device and up to 10 outputs per PLL. You can independently program every output, creating a unique, customizable clock frequency with no fixed relation to any other input or output clock. Inherent jitter filtration and fine granularity control over multiply, divide ratios, and dynamic phase shift reconfiguration provide the high performance precision required in today's high-speed applications. Stratix IV device PLLs are feature rich, supporting advanced capabilities such as clock switchover, dynamic phase shifting, PLL reconfiguration, and reconfigurable bandwidth. Stratix IV PLLs also support external feedback mode, spread-spectrum tracking, and post-scale counter-cascading features.

The Quartus II software enables the PLLs and their features without requiring any external devices. The following sections describe the Stratix IV clock networks and PLLs in detail.

This chapter contains the following sections:

- "Clock Networks in Stratix IV Devices" on page 5–1
- "PLLs in Stratix IV Devices" on page 5–16

Clock Networks in Stratix IV Devices

The GCLKs, RCLKs, and PCLKs available in Stratix IV devices are organized into hierarchical clock structures that provide up to 236 unique clock domains (16 GCLK + 88 RCLK + 132 PCLK) within the Stratix IV device and allows up to 71 unique GCLK, RCLK, and PCLK clock sources (16 GCLK + 22 RCLK + 33 PCLK) per device quadrant. Table 5–1 shows the clock resources available in Stratix IV devices.

Table 5–1. Clock Resources in Stratix IV Devices *(Note 1)*, *(2)*, *(3)*, *(4)* (Part 1 of 2)

Clock Resource	# of Resources Available	Source of Clock Resource
Clock input pins	32 Single-ended (16 Differential)	`CLK[0..15]p` and `CLK[0..15]n` pins
Global clock networks	16	`CLK[0..15]p/n` pins, PLL clock outputs, and logic array

Table 5–1. Clock Resources in Stratix IV Devices *(Note 1)*, *(2)*, *(3)*, *(4)* (Part 2 of 2)

Clock Resource	# of Resources Available	Source of Clock Resource
Regional clock networks	64/88 *(1)*	`CLK[0..15]p/n` pins, PLL clock outputs, and logic array
Peripheral clock networks	132 (33 per device quadrant) *(2)*	DPA clock outputs, horizontal I/O pins, and logic array
GCLKs/RCLKs per quadrant	32/38 *(3)*	16 GCLKs + 16 RCLKs/ 16 GCLKs + 22 RCLKs
GCLKs/RCLKs per device	80/104 *(4)*	16 GCLKs + 64 RCLKs / 16 GCLKs + 88 RCLKs

Notes to Table 5–1:

(1) There are 64 RCLKs in EP4SE110, EP4SE230, EP4SGX70, EP4SGX110, EP4SGX230 devices. There are 88 RCLKs in EP4SE290, EP4SE360, EP4SE530, EP4SE680, EP4SGX290, EP4SGX360, EP4SGX530 devices.

(2) There are 56 PCLKs in EP4SE110, EP4SGX70, EP4SGX110 devices. There are 88 PCLKs in EP4SE230, EP4SE290, EP4SE360, EP4SGX230, EP4SGX290, EP4SGX360 devices. There are 112 PCLKs in EP4SE530, EP4SGX530 devices. There are 132 PCLKs in the EP4SE680 device.

(3) There are 32 GCLKs/RCLKs per quadrant in EP4SE110, EP4SE230, EP4SGX70, EP4SGX110, EP4SGX230 devices. There are 38 GCLKs/RCLKs per quadrant in EP4SE290, EP4SE360, EP4SE530, EP4SE680, EP4SGX290, EP4SGX360, EP4SGX530 devices.

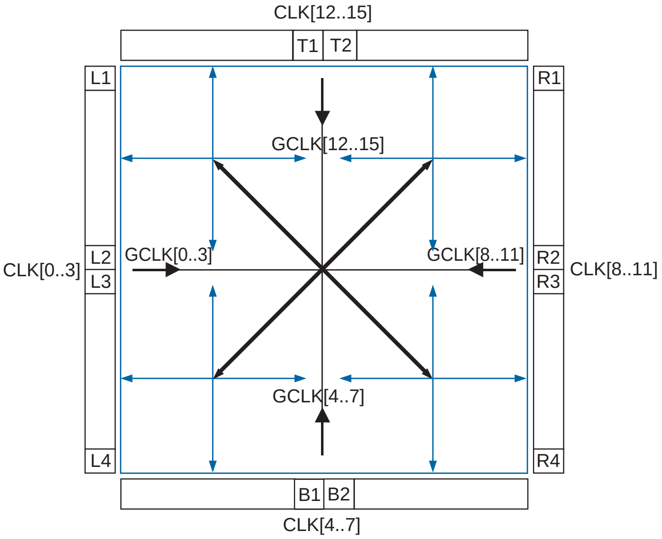
(4) There are 80 GCLKs/RCLKs per entire device in EP4SE110,EP4SE230, EP4SGX70, EP4SGX110, EP4SGX230 devices. There are 104 GCLKs/RCLKS per entire device in EP4SE290, EP4SE360, EP4SE530, EP4SE680, EP4SGX290, EP4SGX360, EP4SGX530 devices.

Stratix IV devices have up to 32 dedicated single-ended clock pins or 16 dedicated differential clock pins (`CLK[0:15]p` and `CLK[0:15]n`) that can drive either the GCLK or RCLK networks. These clock pins are arranged on the four sides of the Stratix IV device, as shown in Figure 5–1 to Figure 5–4.

Global Clock Networks

Stratix IV devices provide up to 16 GCLKs that can drive throughout the entire device, serving as low-skew clock sources for functional blocks like adaptive logic modules (ALMs), digital signal processing (DSP) blocks, TriMatrix memory blocks, and PLLs. Stratix IV device I/O elements (IOEs) and internal logic can also drive GCLKs to create internally generated global clocks and other high fan-out control signals; for example, synchronous or asynchronous clears and clock enables. Figure 5–1 shows CLK pins and PLLs that can drive GCLK networks in Stratix IV devices.

Figure 5–1. Global Clock Networks



Regional Clock Networks

The regional clock (RCLK) networks only pertain to the quadrant they drive into. The RCLK networks provide the lowest clock delay and skew for logic contained within a single device quadrant. Stratix IV device I/O elements and internal logic within a given quadrant can also drive RCLKs to create internally generated regional clocks and other high fan-out control signals; for example, synchronous or asynchronous clears and clock enables. Figure 5–2 to Figure 5–4 show CLK pins and PLLs that can drive RCLK networks in Stratix IV devices. The EP4SE110, EP4SE230, EP4SGX70, EP4SGX110, and EP4SGX230 devices contain 64 RCLKs; the EP4SE230, EP4SE290, EP4SE360, EP4SGX230, EP4SGX290, and EP4SGX360 devices contain 88 RCLKs.

Figure 5–2. Regional Clock Networks (EP4SE110, EP4SGX70, EP4SGX110 Devices)

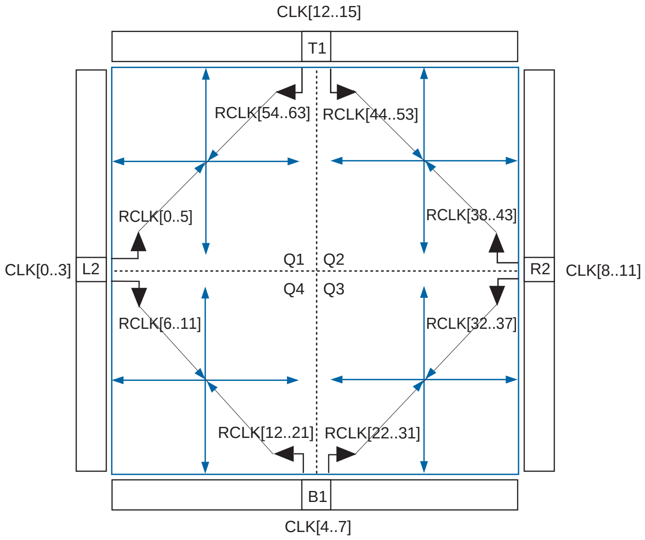


Figure 5–3. Regional Clock Networks (EP4SE230, EP4SE290, EP4SGX230, EP4SGX290 Devices)

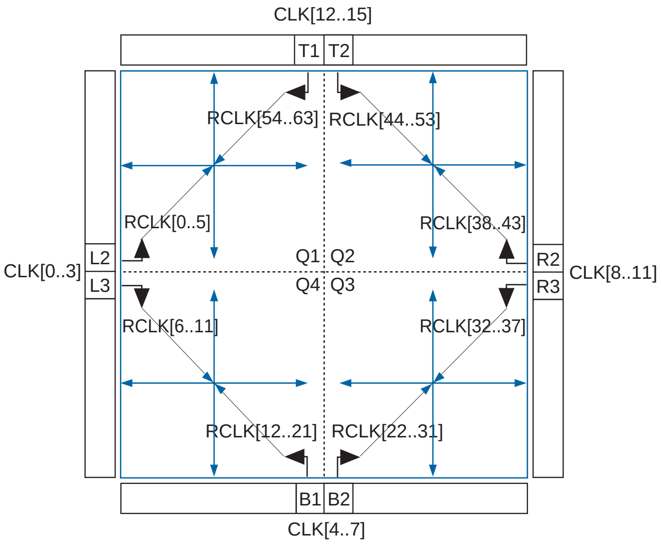
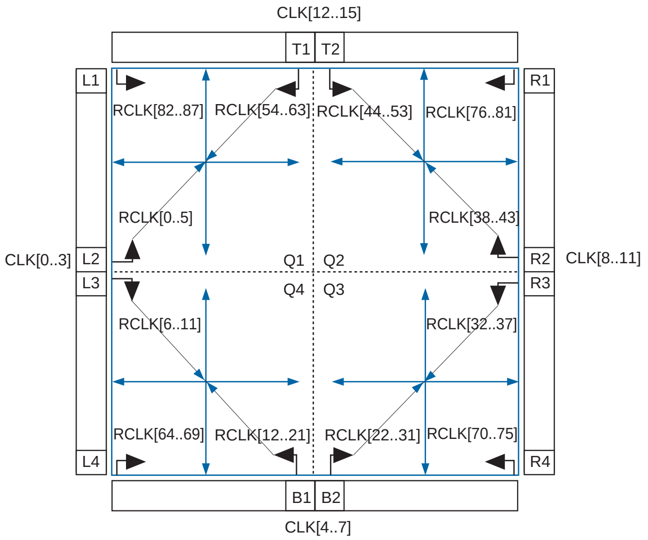


Figure 5–4. Regional Clock Networks (EP4SE360, EP4SE530, EP4SE680, EP4SGX360, EP4SGX530 Devices) *(Note 1)*



Notes to Figure 5–4:

(1) The corner RCLKs [64..87] can only be fed by their respective corner PLL outputs. See Table 5–6 for connectivity.

Periphery Clock Networks

Periphery clock (PCLK) networks are a collection of individual clock networks driven from the periphery of the Stratix IV device. Clock outputs from the DPA block, PLD-transceiver interface clocks, horizontal I/O pins, and internal logic can drive the PCLK networks. The EP4SE110, EP4SGX70, and EP4SGX110 devices contain 56 PCLKs; the EP4SE230, EP4SE290, EP4SE360, EP4SGX230, EP4SGX290, and EP4SGX360 devices contain 88 PCLKs; the EP4SE530 and EP4SGX530 devices contain 112 PCLKs, while the EP4SE680 device contains 132 PCLKs. These PCLKs have higher skew compared to GCLK and RCLK networks and can be used instead of general purpose routing to drive signals into and out of the Stratix IV device.

☞ The legal clock sources for PCLK networks are clock outputs from the DPA block, PLD-Transceiver interface clocks, horizontal I/O pins, and internal logic.

Clocking Regions

Stratix IV devices provide up to 104 distinct clock domains (16 GCLKs + 88 RCLKs) in the entire device. You can utilize these clock resources to form the following four different types of clock regions:

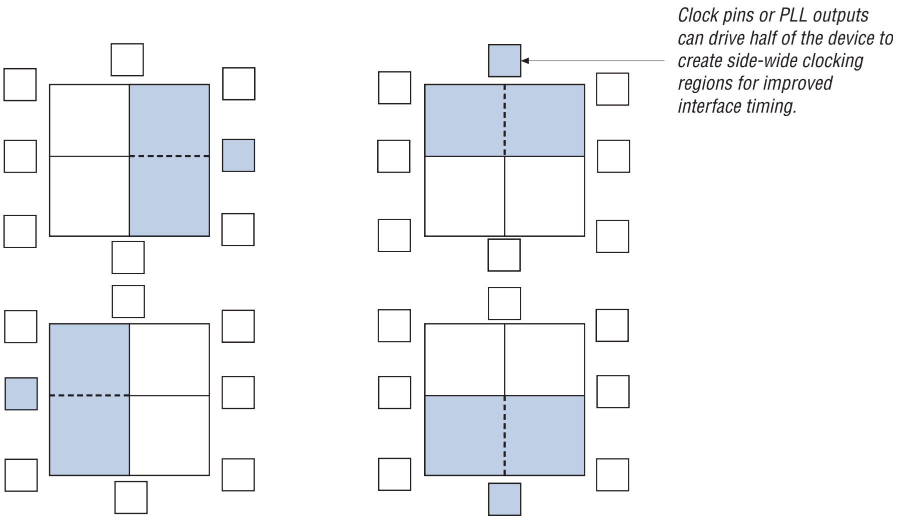
- Entire device clock region
- Regional clock region
- Dual-regional clock region
- Sub-regional clock region

In order to form the entire device clock region, a source (not necessarily a clock signal) drives a global clock network that can be routed through the entire device. This clock region has the maximum delay compared to other clock regions, but allows the signal to reach every destination within the device. This is a good option for routing global reset/clear signals or routing clocks throughout the device.

In order to form a regional clock region, a source drives a single-quadrant of the device. This clock region provides the lowest skew within a quadrant and is a good option if all destinations are within a single device quadrant.

To form a dual-regional clock region, a single source (a clock pin or PLL output) generates a dual-regional clock by driving two regional clock networks (one from each quadrant). This technique allows destinations across two device quadrants to utilize the same low-skew clock. The routing of this signal on an entire side has approximately the same delay as in a regional clock region. Internal logic can also drive a dual-regional clock network. Corner PLL outputs only span one quadrant and hence cannot generate a dual-regional clock network. Figure 5–5 shows the dual-regional clock region.

Figure 5–5. Stratix IV Dual-Regional Clock Region


Clock pins or PLL outputs can drive half of the device to create side-wide clocking regions for improved interface timing.

The sub-regional clock scheme allows the formation of independent sub-regional clock regions for optimal and efficient use of global and regional clock resources. You can partition the device into a maximum of 16 sub-regional clock regions. Each region is driven by a global or regional clock or by an adjacent adaptive logic module (ALM). This technique allows the formation of optimally-sized synchronous clock regions for the best utilization of clock network resources. Figure 5–6 to Figure 5–8 show that you can divide the device into 16, 8, or 12 independent sub-regions.

Figure 5–6. Sixteen Independent Sub-Regional Clock Regions

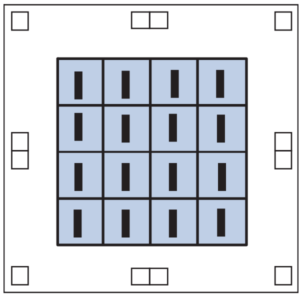


Figure 5–7. Eight Independent Sub-Regional + One Dual-Regional Clock Region

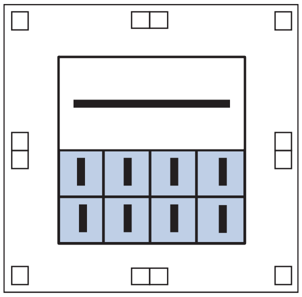
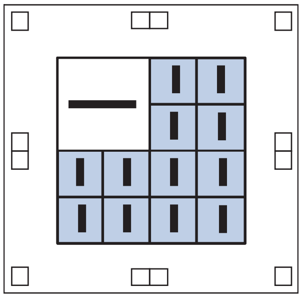


Figure 5–8. Twelve Independent Sub-Regional + One Regional Clock Region



Clock Network Sources

In Stratix IV devices, clock input pins, PLL outputs, and internal logic can drive the global and regional clock networks. See Table 5–2 and Table 5–3 for the connectivity between dedicated `CLK[0..15]` pins and the global and regional clock networks.

Dedicated Clock Inputs Pins

The `CLK` pins can either be differential clocks or single-ended clocks. Stratix IV devices support 16 differential clock inputs or 32 single-ended clock inputs. You can also use the dedicated clock input pins `CLK[15..0]` for high fan-out control signals such as asynchronous clears, presets, and clock enables for protocol signals such as `TRDY` and `IRDY` for PCI through global or regional clock networks.

Logic Array Blocks (LABs)

You can also drive each global and regional clock network using LAB-routing to enable internal logic to drive a high fan-out, low-skew signal.

☞ Stratix IV PLLs cannot be driven by internally generated GCLKs or RCLKs. The input clock to the PLL has to come from dedicated clock input pins or pin/PLL-fed GCLKs or RCLKs only.

PLL Clock Outputs

Stratix IV PLLs can drive both GCLK and RCLK networks, as shown in Table 5–5 and Table 5–6.

Table 5–2 shows the connection between the dedicated clock input pins and GCLKs.

Table 5–2. Clock Input Pin Connectivity to Global Clock Networks (Part 1 of 2)

Clock Resources	CLK (p/n Pins)															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
GCLK0	✓	✓	✓	✓	—	—	—	—	—	—	—	—	—	—	—	—
GCLK1	✓	✓	✓	✓	—	—	—	—	—	—	—	—	—	—	—	—
GCLK2	✓	✓	✓	✓	—	—	—	—	—	—	—	—	—	—	—	—
GCLK3	✓	✓	✓	✓	—	—	—	—	—	—	—	—	—	—	—	—
GCLK4	—	—	—	—	✓	✓	✓	✓	—	—	—	—	—	—	—	—
GCLK5	—	—	—	—	✓	✓	✓	✓	—	—	—	—	—	—	—	—

Table 5–2. Clock Input Pin Connectivity to Global Clock Networks (Part 2 of 2)

Clock Resources	CLK (p/n Pins)															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
GCLK6	—	—	—	—	✓	✓	✓	✓	—	—	—	—	—	—	—	—
GCLK7	—	—	—	—	✓	✓	✓	✓	—	—	—	—	—	—	—	—
GCLK8	—	—	—	—	—	—	—	—	✓	✓	✓	✓	—	—	—	—
GCLK9	—	—	—	—	—	—	—	—	✓	✓	✓	✓	—	—	—	—
GCLK10	—	—	—	—	—	—	—	—	✓	✓	✓	✓	—	—	—	—
GCLK11	—	—	—	—	—	—	—	—	✓	✓	✓	✓	—	—	—	—
GCLK12	—	—	—	—	—	—	—	—	—	—	—	—	✓	✓	✓	✓
GCLK13	—	—	—	—	—	—	—	—	—	—	—	—	✓	✓	✓	✓
GCLK14	—	—	—	—	—	—	—	—	—	—	—	—	✓	✓	✓	✓
GCLK15	—	—	—	—	—	—	—	—	—	—	—	—	✓	✓	✓	✓

Table 5–3 shows the connectivity between the dedicated clock input pins and RCLKs in devices. A given clock input pin can drive two adjacent regional clock networks to create a dual-regional clock network.

Table 5–3. *Clock Input Pin Connectivity to Regional Clock Networks (Part 1 of 2)*

Clock Resource	CLK (p/n Pins)															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
RCLK [0, 4, 6, 10]	✓	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RCLK [1, 5, 7, 11]	—	✓	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RCLK [2, 8]	—	—	✓	—	—	—	—	—	—	—	—	—	—	—	—	—
RCLK [3, 9]	—	—	—	✓	—	—	—	—	—	—	—	—	—	—	—	—
RCLK [12, 16, 20, 22, 26, 30]	—	—	—	—	✓	—	—	—	—	—	—	—	✓	—	—	—
RCLK [13, 17, 21, 23, 27, 31]	—	—	—	—	—	✓	—	—	—	—	—	—	—	✓	—	—
RCLK [14, 18, 24, 28]	—	—	—	—	—	—	✓	—	—	—	—	—	—	—	✓	—
RCLK [15, 19, 25, 29]	—	—	—	—	—	—	—	✓	—	—	—	—	—	—	—	✓
RCLK [32, 36, 38, 42]	—	—	—	—	—	—	—	—	✓	—	—	—	✓	—	—	—
RCLK [33, 37, 39, 43]	—	—	—	—	—	—	—	—	—	✓	—	—	—	✓	—	—
RCLK [34, 40]	—	—	—	—	—	—	—	—	—	—	✓	—	—	—	✓	—
RCLK [35, 41]	—	—	—	—	—	—	—	—	—	—	—	✓	—	—	—	✓
RCLK [44, 48, 52, 54, 58, 62]	—	—	—	—	—	—	—	—	—	—	—	—	✓	—	—	—
RCLK [45, 49, 53, 55, 59, 63]	—	—	—	—	—	—	—	—	—	—	—	—	—	✓	—	—

Table 5–3. *Clock Input Pin Connectivity to Regional Clock Networks (Part 2 of 2)*

Clock Resource	CLK (p/n Pins)															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
`RCLK [46, 50, 56, 60]`	—	—	—	—	—	—	—	—	—	—	—	—	—	—	✓	—
`RCLK [47, 51, 57, 61]`	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	✓

Clock Input Connections to PLLs

Dedicated clock input pin connectivity to Stratix IV PLLs is shown in Table 5–4.

Table 5–4. *Stratix IV Device PLLs and PLL Clock Pin Drivers*

Dedicated Clock Input Pin (CLKp/n pins)	PLL Number											
	L1	L2	L3	L4	B1	B2	R1	R2	R3	R4	T1	T2
`CLK0`	✓	✓	✓	✓	—	—	—	—	—	—	—	—
`CLK1`	✓	✓	✓	✓	—	—	—	—	—	—	—	—
`CLK2`	✓	✓	✓	✓	—	—	—	—	—	—	—	—
`CLK3`	✓	✓	✓	✓	—	—	—	—	—	—	—	—
`CLK4`	—	—	—	—	✓	✓	—	—	—	—	—	—
`CLK5`	—	—	—	—	✓	✓	—	—	—	—	—	—
`CLK6`	—	—	—	—	✓	✓	—	—	—	—	—	—
`CLK7`	—	—	—	—	✓	✓	—	—	—	—	—	—
`CLK8`	—	—	—	—	—	—	✓	✓	✓	✓	—	—
`CLK9`	—	—	—	—	—	—	✓	✓	✓	✓	—	—
`CLK10`	—	—	—	—	—	—	✓	✓	✓	✓	—	—
`CLK11`	—	—	—	—	—	—	✓	✓	✓	✓	—	—
`CLK12`	—	—	—	—	—	—	—	—	—	—	✓	✓
`CLK13`	—	—	—	—	—	—	—	—	—	—	✓	✓
`CLK14`	—	—	—	—	—	—	—	—	—	—	✓	✓
`CLK15`	—	—	—	—	—	—	—	—	—	—	✓	✓

Clock Output Connections

PLLs in Stratix IV devices can drive up to 20 regional clock networks and four global clock networks. Refer to Table 5–5 for Stratix IV PLL connectivity to GCLK networks. The Quartus II software automatically assigns PLL clock outputs to regional or global clock networks.

Table 5–5 shows how the PLL clock outputs connect to GCLK networks.

Table 5–5. Stratix IV PLL Connectivity to GCLKs

Clock Network	PLL Number											
	L1	L2	L3	L4	B1	B2	R1	R2	R3	R4	T1	T2
`GCLK0`	✓	✓	✓	✓	—	—	—	—	—	—	—	—
`GCLK1`	✓	✓	✓	✓	—	—	—	—	—	—	—	—
`GCLK2`	✓	✓	✓	✓	—	—	—	—	—	—	—	—
`GCLK3`	✓	✓	✓	✓	—	—	—	—	—	—	—	—
`GCLK4`	—	—	—	—	✓	✓	—	—	—	—	—	—
`GCLK5`	—	—	—	—	✓	✓	—	—	—	—	—	—
`GCLK6`	—	—	—	—	✓	✓	—	—	—	—	—	—
`GCLK7`	—	—	—	—	✓	✓	—	—	—	—	—	—
`GCLK8`	—	—	—	—	—	—	✓	✓	✓	✓	—	—
`GCLK9`	—	—	—	—	—	—	✓	✓	✓	✓	—	—
`GCLK10`	—	—	—	—	—	—	✓	✓	✓	✓	—	—
`GCLK11`	—	—	—	—	—	—	✓	✓	✓	✓	—	—
`GCLK12`	—	—	—	—	—	—	—	—	—	—	✓	✓
`GCLK13`	—	—	—	—	—	—	—	—	—	—	✓	✓
`GCLK14`	—	—	—	—	—	—	—	—	—	—	✓	✓
`GCLK15`	—	—	—	—	—	—	—	—	—	—	✓	✓

Table 5–6 shows how the PLL clock outputs connect to RCLK networks.

Table 5–6. Stratix IV Regional clock Outputs From PLLs

Clock Resource	PLL Number											
	L1	L2	L3	L4	B1	B2	R1	R2	R3	R4	T1	T2
`RCLK[0..11]`	—	✓	✓	—	—	—	—	—	—	—	—	—
`RCLK[12..31]`	—	—	—	—	✓	✓	—	—	—	—	—	—
`RCLK[32..43]`	—	—	—	—	—	—	—	✓	✓	—	—	—
`RCLK[44..63]`	—	—	—	—	—	—	—	—	—	—	✓	✓
`RCLK[64..69]`	—	—	—	✓	—	—	—	—	—	—	—	—
`RCLK[70..75]`	—	—	—	—	—	—	—	—	—	✓	—	—
`RCLK[76..81]`	—	—	—	—	—	—	✓	—	—	—	—	—
`RCLK[82..87]`	✓	—	—	—	—	—	—	—	—	—	—	—

Clock Control Block

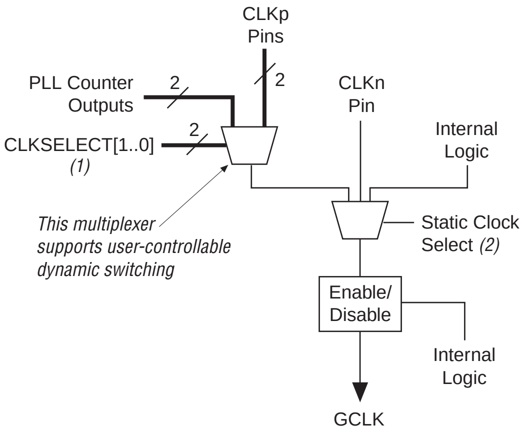
Every global and regional clock network has its own clock control block. The control block provides the following features:

- Clock source selection (dynamic selection for global clocks)
- Global clock multiplexing
- Clock power down (static or dynamic clock enable or disable)

Figure 5–9 and Figure 5–10 show the global clock and regional clock select blocks, respectively.

You can select the clock source for the global clock select block either statically or dynamically. You can statically select the clock source using a setting in the Quartus II software, or you can dynamically select the clock source using internal logic to drive the multiplexer select inputs. When selecting the clock source dynamically, you can either select PLL outputs (such as `C0` or `C1`), or a combination of clock pins or PLL outputs.

Figure 5–9. Stratix IV Global Clock Control Block



Notes to Figure 5–9:

(1) These clock select signals can only be dynamically controlled through internal logic when the device is operating in user mode.

(2) These clock select signals can only be set through a configuration file (**.sof** or **.pof**) and cannot be dynamically controlled during user mode operation.

The mapping between input clock pins, pll counter outputs, and clock control block inputs is as follows:

- inclk[0], inclk[1]

  can be fed by any of the 4 dedicated clock pins on the same side

- inclk[2]

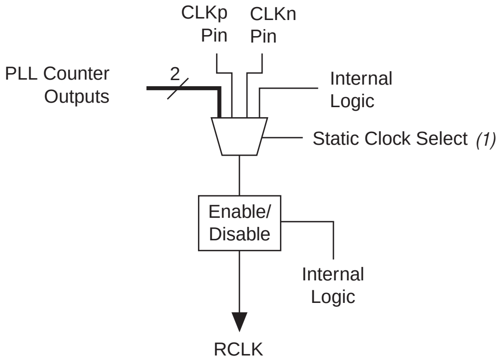
  can be fed by PLL counters C0 and C2 from the two center PLLs on the same side of the Stratix IV device

- inclk[3]

  can be fed by PLL counters C1 and C3 from the two center PLLs on the same side of the Stratix IV device

The corner PLLs (L1, L4, R1, R4) and corresponding clock input pins (PLL_L1_CLK, etc.) do not support dynamic selection for the global clock network. The clock source selection for the GCLK/RCLK networks from corner PLLs (L1, L4, R1, R4) and corresponding clock input pins (PLL_L1_CLK, etc.) is controlled statically using configuration bit settings in the configuration file (**.sof** or **.pof**) generated by the Quartus II software.

Figure 5–10. Regional Clock Control Block



Note to Figure 5–10:

(1) This clock select signal can only be controlled through a configuration file (**.sof** or **.pof**) and cannot be dynamically controlled during user mode operation.

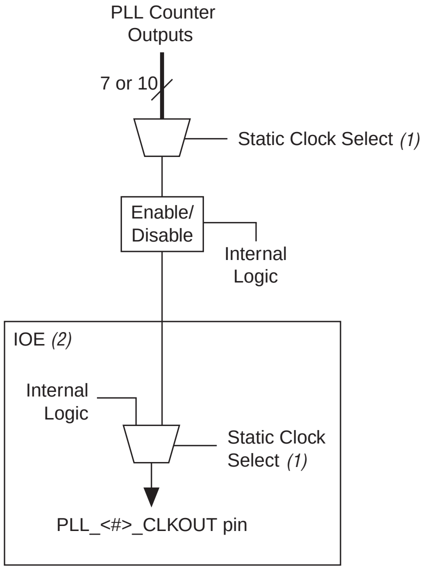
The clock source selection for the regional clock select block can only be controlled statically using configuration bit settings in the configuration file (**.sof** or **.pof**) generated by the Quartus II software.

The Stratix IV clock networks can be powered down by both static and dynamic approaches. When a clock network is powered down, all the logic fed by the clock network is in an off-state, thereby reducing the overall power consumption of the device. The unused global and regional clock networks are automatically powered down through configuration bit settings in the configuration file (**.sof** or **.pof**) generated by the Quartus II software. The dynamic clock enable or disable feature allows the internal logic to control power-up or power-down synchronously on GCLK and RCLK networks, including dual-regional clock regions. This function is independent of the PLL and is applied directly on the clock network, as shown in Figure 5–9 and Figure 5–10.

You can set the input clock sources and the `clkena` signals for the global and regional clock network multiplexers through the Quartus II software using the ALTCLKCTRL megafunction. You can also enable or disable the dedicated external clock output pins using the ALTCLKCTRL megafunction. Figure 5–11 shows the external PLL output clock control block.

☞ When using the ALTCLKCTRL megafunction to implement dynamic clock source selection, the inputs from the clock pins feed the `inclk[0..1]` ports of the multiplexer, while the PLL outputs feed the `inclk[2..3]` ports. You can choose from among these inputs using the `CLKSELECT[1..0]` signal.

Figure 5–11. Stratix IV External PLL Output Clock Control Block



Notes to Figure 5–11:

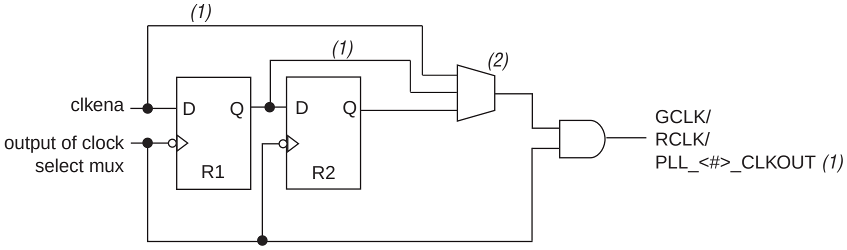
(1) This clock select signal can only be set through a configuration file (**.sof** or **.pof**) and cannot be dynamically controlled during user mode operation.

(2) The clock control block feeds to a multiplexer within the `PLL_<#>_CLKOUT` pin's IOE. The `PLL_<#>_CLKOUT` pin is a dual-purpose pin. Therefore, this multiplexer selects either an internal signal or the output of the clock control block.

Clock Enable Signals

Figure 5–12 shows how the clock enable/disable circuit of the clock control block is implemented in Stratix IV devices.

Figure 5–12. clkena Implementation



Notes to Figure 5–12:

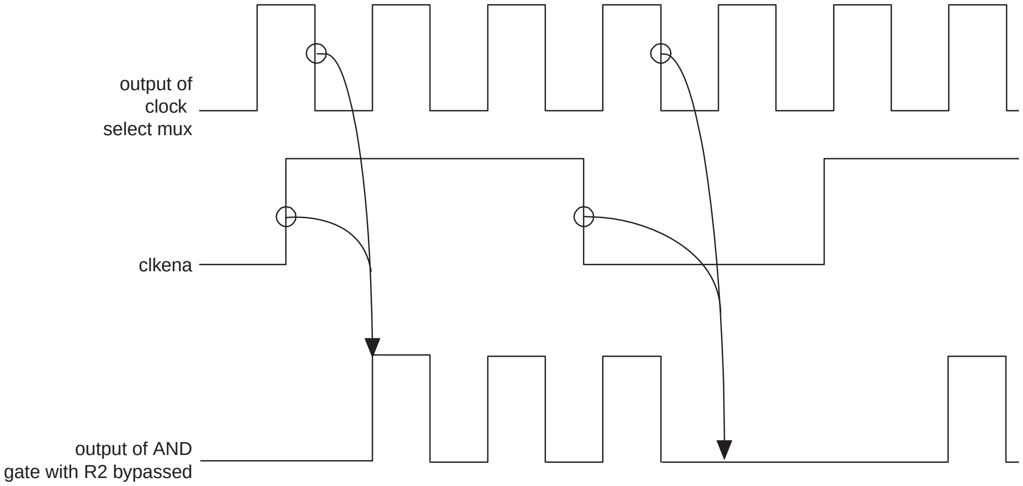
(1) The R1 and R2 bypass paths are not available for PLL external clock outputs.

(2) The select line is statically controlled by a bit setting in the configuration file (**.sof** or **.pof**).

In Stratix IV devices, the `clkena` signals are supported at the clock network level instead of at the PLL output counter level. This allows you to gate off the clock even when a PLL is not being used. You can also use the `clkena` signals to control the dedicated external clocks from the PLLs. Figure 5–13 shows the waveform example for a clock output enable. `clkena` is synchronous to the falling edge of the clock output.

Stratix IV devices also have an additional metastability register that aids in asynchronous enable/disable of the GCLK/RCLK networks. This register can be optionally bypassed in the Quartus II software.

Figure 5–13. clkena Signals *(Note 1)*



***Note to* Figure 5–13:**

(1) You can use the `clkena` signals to enable or disable the global and regional networks or the `PLL_<#>_CLKOUT` pins.

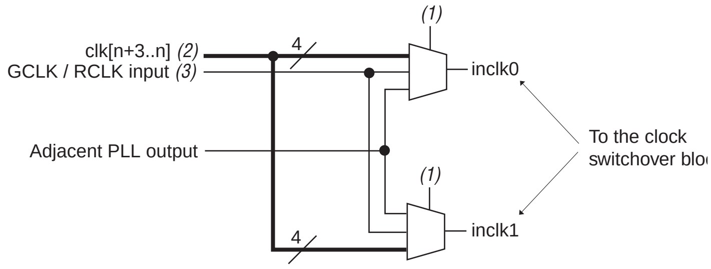
The PLL can remain locked independent of the `clkena` signals since the loop-related counters are not affected. This feature is useful for applications that require a low power or sleep mode. The `clkena` signal can also disable clock outputs if the system is not tolerant of frequency over-shoot during resynchronization.

Clock Source Control for PLLs

The clock input to Stratix IV PLLs comes from clock input multiplexers. The clock multiplexer inputs come from dedicated clock input pins, PLLs through the GCLK and RCLK networks, or from dedicated connections between adjacent Top/Bottom and Left/Right PLLs. The clock input sources to Top/Bottom and Left/Right PLLs (L2, L3, T1, T2, B1, B2, R2, and R3) are shown in Figure 5–14; the corresponding clock input sources to Left/Right PLLs (L1, L4, R1, and R4) are shown in Figure 5–15.

The multiplexer select lines are set in the configuration file (SRAM object file [**.sof**] or programmer object file [**.pof**]) only. Once programmed, this block cannot be changed without loading a new configuration file (**.sof** or **.pof**). The Quartus II software automatically sets the multiplexer select signals depending on the clock sources selected in the design.

Figure 5–14. Clock Input Multiplexer Logic for L2, L3, T1, T2, B1, B2, R2, and R3 PLLs



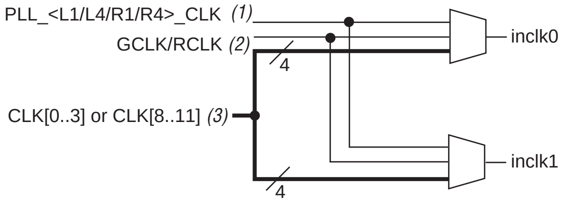
***Notes to* Figure 5–14:**

(1) The input clock multiplexing is controlled through a configuration file (**.sof** or **.pof**) only and cannot be dynamically controlled in user mode operation.

(2) n=0 for L2 and L3 PLLs; n=4 for B1 and B2 PLLs; n=8 for R2 and R3 PLLs, and n=12 for T1 and T2 PLLs.

(3) The global (GCLK) or regional (RCLK) clock input can be driven by an output from another PLL, a pin-driven global or regional clock, or through a clock control block provided the clock control block is fed by an output from another PLL or a pin-driven dedicated global or regional clock. An internally generated global signal or general purpose I/O pin cannot drive the PLL.

Figure 5–15. Clock Input Multiplexer Logic for L1, L4, R1, and R4 PLLs



Notes to Figure 5–15:

(1) Dedicated clock input pins to PLLs - L1, L4, R1 and R4, respectively. For example, `PLL_L1_CLK` is the dedicated clock input for `PLL_L1`.

(2) The global (GCLK) or regional (RCLK) clock input can be driven by an output from another PLL, a pin-driven global or regional clock, or through a clock control block provided the clock control block is fed by an output from another PLL or a pin driven dedicated global or regional clock. An internally-generated global signal or general purpose I/O pin cannot drive the PLL.

(3) The center clock pins can feed the corner PLLs on the same side directly, through a dedicated path. However, these paths may not be fully compensated.

Cascading PLLs

The Left/Right and Top/Bottom PLLs can be cascaded through the GCLK/RCLK networks. In addition, where two Left/Right or Top/Bottom PLLs exist next to each other, there is a direct connection between them that does not require the GCLK/RCLK network. Using this path reduces clock jitter when cascading PLLs. Stratix IV GX devices allow cascading of Left/Right PLLs to transceiver PLLs (transceiver CMU and receiver CDR).

For more information, refer to the *FPGA Fabric PLLs -Transceiver PLLs Cascading* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*.

PLLs in Stratix IV Devices

Stratix IV devices offer up to 12 PLLs that provide robust clock management and synthesis for device clock management, external system clock management, and high-speed I/O interfaces. The nomenclature for the PLLs follows their geographical location in the device floor plan. The PLLs that reside on the top and bottom sides of the device are named PLL_T1, PLL_T2, PLL_B1 and PLL_B2; the PLLs that reside on the left and right sides of the device are named PLL_L1, PLL_L2, PLL_L3, PLL_L4, PLL_R1, PLL_R2, PLL_R3, and PLL_R4.

Table 5–7 shows the number of PLLs available in the Stratix IV device family.

Table 5–7. Stratix IV Device PLL Availability *(Note 1)*, *(2)*, *(3)*, *(4)*, *(5)*, *(6)*, *(7)*

Device	L1	L2	L3	L4	T1	T2	B1	B2	R1	R2	R3	R4
EP4SE110	—	✓	—	—	✓	—	✓	—	—	✓	—	—
EP4SE230	—	✓	—	—	✓	—	✓	—	—	✓	—	—
EP4SE290	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EP4SE360	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EP4SE530	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EP4SE680	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EP4SGX70	—	✓	—	—	✓	—	✓	—	—	—	—	—
EP4SGX110	—	✓	—	—	✓	—	✓	—	—	✓	—	—
EP4SGX230	—	✓	✓	—	✓	✓	✓	✓	—	✓	✓	—
EP4SGX290	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EP4SGX360	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EP4SGX530	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Notes to Table 5–7:

(1) EP4SGX70 and EP4SGX110 devices in the F780 package support 3 PLLs - L2/T1/B1

(2) EP4SE110, EP4SE230, EP4SE290, EP4SE360, EP4SGX290, and EP4SGX360 devices in the F780 package support 4 PLLs - T1/T2/B1/B2

(3) The EP4SGX110 device in the F1152 package supports 4 PLLs - L2/T1/B1/R2

(4) EP4SGX230, EP4SGX290, EP4SGX360, and EP4SGX530 devices in the F1152 package support 6 PLLs - L2/T1/T2/B1/B2/R2

(5) EP4SE290, EP4SE360, EP4SE530, and EP4SE680 devices in the F1152 package support 8 PLLs - L1/L2/T1/T2/B1/B2/R1/R2

(6) EP4SGX230, EP4SGX290, EP4SGX360, and EP4SGX530 devices in the F1517 package support 8 PLLs - L2/L3/T1/T2/B1/B2/R2/R3

(7) EP4SE290, EP4SE360, EP4SE530, and EP4SE680 devices in the F1517 package and EP4SE530, EP4SE680, and EP4SGX530 devices in the F1932 package support all 12 PLLs.

All Stratix IV PLLs have the same core analog structure with only minor differences in the features that are supported. Table 5–8 highlights the features of Top/Bottom and Left/Right PLLs in Stratix IV devices.

Table 5–8. Stratix IV PLL Features (Part 1 of 2)

Feature	Stratix IV Top/Bottom PLLs	Stratix IV Left/Right PLLs
C (output) counters	10	7
M, N, C counter sizes	1 to 512	1 to 512
Dedicated clock outputs	6 single-ended or 4 single-ended and 1 differential pair	2 single-ended or 1 differential pair
Clock input pins	8 single-ended or 4 differential pin pairs	8 single-ended or 4 differential pin pairs

Table 5–8. Stratix IV PLL Features (Part 2 of 2)

Feature	Stratix IV Top/Bottom PLLs	Stratix IV Left/Right PLLs
External feedback input pin	Single-ended or differential	Single-ended only
Spread-spectrum input clock tracking	Yes *(1)*	Yes *(1)*
PLL cascading	Through GCLK and RCLK and dedicated path between adjacent PLLs	Through GCLK and RCLK and dedicated path between adjacent PLLs *(2)*
Compensation modes	All except LVDS clock network compensation	All except external feedback mode when using differential I/Os
PLL drives LVDSCLK and LOADEN	No	Yes
VCO output drives DPA clock	No	Yes
Phase shift resolution	Down to 96.125 ps *(3)*	Down to 96.125 ps *(3)*
Programmable duty cycle	Yes	Yes
Output counter cascading	Yes	Yes
Input clock switchover	Yes	Yes

Notes to Table 5–8:

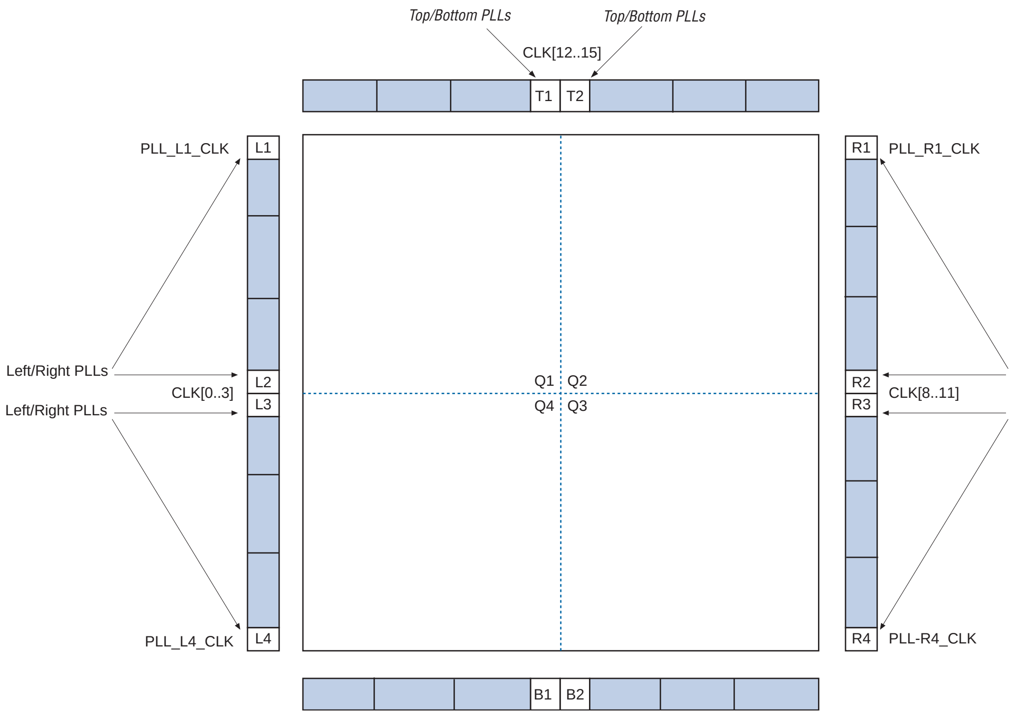
(1) Provided input clock jitter is within input jitter tolerance specifications.

(2) The dedicated path between adjacent PLLs is not available on L1, L4, R1, and R4 PLLs.

(3) The smallest phase shift is determined by the voltage-controlled oscillator (VCO) period divided by eight. For degree increments, the Stratix IV device can shift all output frequencies in increments of at least 45 degrees. Smaller degree increments are possible depending on the frequency and divide parameters.

Figure 5–16 shows the location of PLLs in Stratix IV devices.

Figure 5–16. Stratix IV PLL Locations



Stratix IV PLL Hardware Overview

Stratix IV devices contain up to 12 PLLs with advanced clock management features. The main goal of a PLL is to synchronize the phase and frequency of an internal or external clock to an input reference clock. There are a number of components that comprise a PLL to achieve this phase alignment.

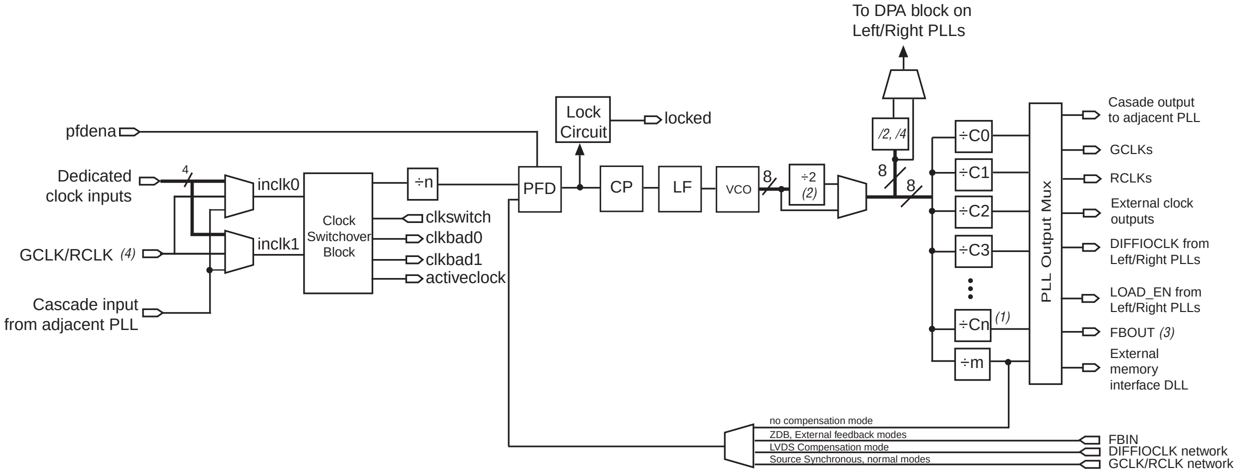
Stratix IV PLLs align the rising edge of the input reference clock to a feedback clock using the phase-frequency detector (PFD). The falling edges are determined by the duty-cycle specifications. The PFD produces an up or down signal that determines whether the voltage-controlled oscillator (VCO) needs to operate at a higher or lower frequency. The output of the PFD feeds the charge pump and loop filter, which produces a control voltage for setting the VCO frequency. If the PFD produces an up signal, then the VCO frequency increases. A down signal decreases the VCO frequency. The PFD outputs these up and down signals to a charge pump. If the charge pump receives an up signal, current is driven into the loop filter. Conversely, if the charge pump receives a down signal, current is drawn from the loop filter.

The loop filter converts these up and down signals to a voltage that is used to bias the VCO. The loop filter also removes glitches from the charge pump and prevents voltage over-shoot, which filters the jitter on the VCO. The voltage from the loop filter determines how fast the VCO operates. A divide counter (m) is inserted in the feedback loop to increase the VCO frequency above the input reference frequency. VCO frequency (f_{VCO}) is equal to (m) times the input reference clock (f_{REF}). The input reference clock (f_{REF}) to the PFD is equal to the input clock (f_{IN}) divided by the pre-scale counter (N). Therefore, the feedback clock (f_{FB}) applied to one input of the PFD is locked to the f_{REF} that is applied to the other input of the PFD.

The VCO output from Left/Right PLLs can feed seven post-scale counters (C[0..6]), while the corresponding VCO output from Top/Bottom PLLs can feed ten post-scale counters (C[0..9]). These post-scale counters allow a number of harmonically related frequencies to be produced by the PLL.

Figure 5–17 shows a simplified block diagram of the major components of the Stratix IV PLL.

Figure 5–17. Stratix IV PLL Block Diagram



Notes to Figure 5–17:

(1) The number of post-scale counters is 7 for Left/Right PLLs and 10 for Top/Bottom PLLs.

(2) This is the VCO post-scale counter K.

(3) The FBOUT port is fed by the M counter in Stratix IV PLLs.

The global (GCLK) or regional (RCLK) clock inputs can be driven by an output from another PLL, a pin-driven global or regional clock, or through a clock control block provided the clock control block is fed by an output from another PLL or a pin-driven dedicated global or regional clock. An internally generated global signal or general purpose I/O pin cannot drive the PLL.

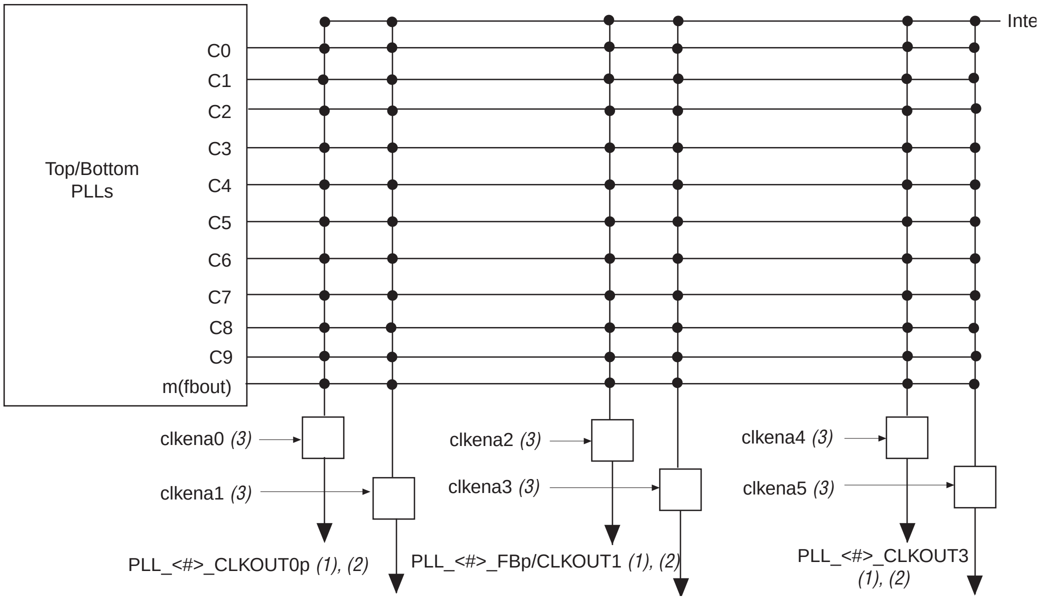
PLL Clock I/O Pins

Each Top/Bottom PLL supports six clock I/O pins, organized as three pairs of pins:

- 1st pair: 2 single-ended I/O or 1 differential I/O
- 2nd pair: 2 single-ended I/O or 1 differential external feedback input (FBp/FBn)
- 3rd pair: 2 single-ended I/O or 1 differential input

Figure 5–18 shows the clock I/O pins associated with Top/Bottom PLLs.

Figure 5–18. External Clock Outputs for Top/Bottom PLLs



Notes to Figure 5–18:

(1) These clock output pins can be fed by any one of the `C[9..0]`, m counters.

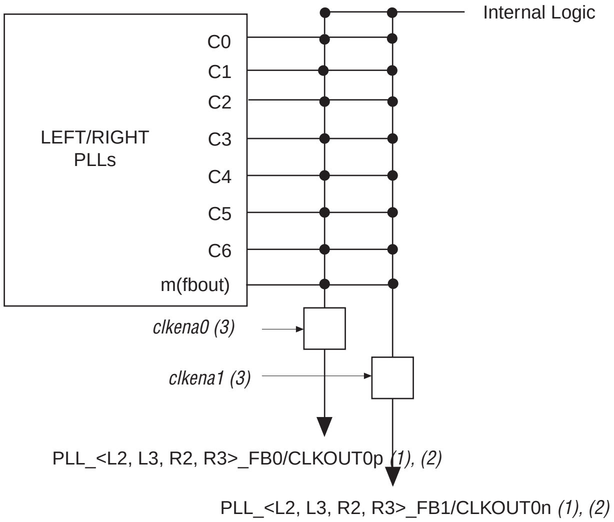
(2) The `CLKOUT0p` and `CLKOUT0n` pins can be either single-ended or differential clock outputs. `CLKOUT1` and `CLKOUT2` pins are dual-purpose I/O pins that can be used as two single-ended outputs or one differential external feedback input pin. `CLKOUT3` and `CLKOUT4` pins are two single-ended output pins.

(3) These external clock enable signals are available only when using the ALTCLKCTRL megafunction.

Any of the output counters (`C[9..0]` on Top/Bottom PLLs and `C[6..0]` on Left/Right PLLs) or the `M` counter can feed the dedicated external clock outputs, as shown in Figure 5–18 and Figure 5–19. Therefore, one counter or frequency can drive all output pins available from a given PLL.

Each Left/Right PLL supports two clock I/O pins, configured as either two single-ended I/Os or one differential I/O pair. When using both pins as single-ended I/Os, one of them can be the clock output while the other pin is the external feedback input (FB) pin. Hence, Left/Right PLLs only support external feedback mode for single-ended I/O standards only.

Figure 5–19. External Clock Outputs for Left/Right PLLs


Internal Logic
LEFT/RIGHT PLLs
C0
C1
C2
C3
C4
C5
C6
m(fbout)
clkena0 (3)
clkena1 (3)
PLL_<L2, L3, R2, R3>_FB0/CLKOUT0p (1), (2)
PLL_<L2, L3, R2, R3>_FB1/CLKOUT0n (1), (2)

Notes to Figure 5–19:

(1) These clock output pins can be fed by any one of the `C[6..0],` m counters.

(2) The `CLKOUT0p` and `CLKOUT0n` pins are dual-purpose I/O pins that can be used as two single-ended outputs or one single-ended output and one external feedback input pin.

(3) These external clock enable signals are available only when using the ALTCLKCTRL megafunction.

Each pin of a single-ended output pair can either be in-phase or 180-degrees out-of-phase. The Quartus II software places the NOT gate in the design into the IOE to implement 180-degree phase with respect to the other pin in the pair. The clock output pin pairs support the same I/O standards as standard output pins (in the top and bottom banks) as well as LVDS, LVPECL, differential HSTL, and differential SSTL.

Refer to the *I/O Features in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook* to determine which I/O standards are supported by the PLL clock input and output pins.

Stratix IV PLLs can also drive out to any regular I/O pin through the global or regional clock network. You can also use the external clock output pins as user I/O pins if external PLL clocking is not needed.

PLL Control Signals

You can use the three signals, `pfdena`, `areset`, and `locked`, to observe and control the PLL operation and resynchronization.

pfdena

Use the `pfdena` signal to maintain the most recent locked frequency so your system has time to store its current settings before shutting down. The pfdena signal controls the PFD output with a programmable gate. If you disable PFD, the VCO operates at its most recent set value of control voltage and frequency, with some long-term drift to a lower frequency. The PLL continues running even if it goes out-of-lock or the input clock is disabled. You can use either your own control signal or the control signals available from the clock switchover circuit (`activeclock`, `clkbad[0]`, or `clkbad[1]`) to control `pfdena`.

areset

The `areset` signal is the reset or resynchronization input for each PLL. The device input pins or internal logic can drive these input signals. When `areset` is driven high, the PLL counters reset, clearing the PLL output and placing the PLL out-of-lock. The VCO is then set back to its nominal setting. When `areset` is driven low again, the PLL will resynchronize to its input as it re-locks.

You should assert the areset signal every time the PLL loses lock to guarantee the correct phase relationship between the PLL input and output clocks. You can set up the PLL to automatically reset (self reset) upon a loss-of-lock condition using the Quartus II MegaWizard. You should include the `areset` signal in designs if any of the following conditions are true:

- PLL reconfiguration or clock switchover is enabled in the design.
- Phase relationships between the PLL input and output clocks need to be maintained after a loss-of-lock condition.

☞ If the input clock to the PLL is not toggling or is unstable upon power up, assert the `areset` signal after the input clock is stable and within specifications.

locked

The locked output of the PLL indicates that the PLL has locked onto the reference clock and the PLL clock outputs are operating at the desired phase and frequency set in the Quartus II software MegaWizard. Without any additional circuitry, the lock signal may toggle as the PLL begins the locking process. The lock detection circuit provides a signal to the core logic that gives an indication when the feedback clock has locked onto the reference clock both in phase and frequency.

☞ Altera recommends that you use the areset and locked signals in your designs to control and observe the status of your PLL.

Clock Feedback Modes

Stratix IV PLLs support up to six different clock feedback modes. Each mode allows clock multiplication and division, phase shifting, and programmable duty cycle. Table 5–9 shows the clock feedback modes supported by Stratix IV device PLLs.

Table 5–9. Clock Feedback Mode Availability *(Note 1)*

	Availability	
Clock Feedback Mode	**Top/Bottom PLLs**	**Left/Right PLLs**
Source-synchronous mode	Yes	Yes
No-compensation mode	Yes	Yes
Normal mode	Yes	Yes
Zero-delay buffer (ZDB) mode	Yes	Yes
External-feedback mode	Yes	Yes *(1)*
LVDS compensation	No	Yes

Note to Table 5–9:

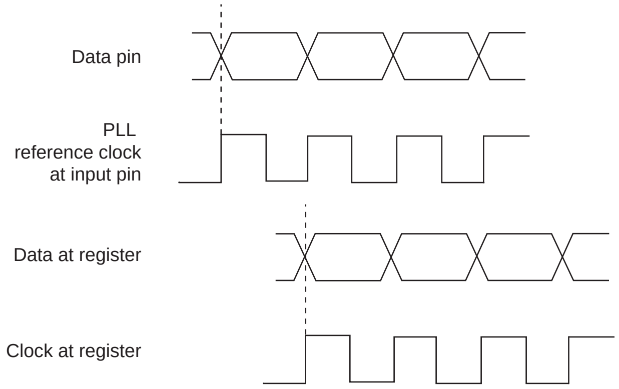
(1) External feedback mode supported for single-ended inputs and outputs only on Left/Right PLLs.

☞ The input and output delays are fully compensated by a PLL only when using the dedicated clock input pins associated with a given PLL as the clock source. For example, when using `PLL_T1` in normal mode, the clock delays from the input pin to the PLL clock output-to-destination register are fully compensated, provided the clock input pin is one of the following four pins: `CLK12`, `CLK13`, `CLK14`, or `CLK15`. When an RCLK or GCLK network drives the PLL, the input and output delays may not be fully compensated in the Quartus II software. Another example is when `PLL_T2` is configured in zero delay buffer mode, and the PLL input is driven by a dedicated clock input pin, a fully compensated clock path results in zero delay between the clock input and one of the output clocks from the PLL. If the PLL input is instead fed by a non-dedicated input (via the GCLK network), then the output clock may not be perfectly aligned with the input clock.

Source Synchronous Mode

If data and clock arrive at the same time on the input pins, the same phase relationship is maintained at the clock and data ports of any IOE input register. Figure 5–20 shows an example waveform of the clock and data in this mode. This mode is recommended for source-synchronous data transfers. Data and clock signals at the IOE experience similar buffer delays as long as you use the same I/O standard.

Figure 5–20. Phase Relationship Between Clock and Data in Source-Synchronous Mode



The source-synchronous mode compensates for the delay of the clock network used plus any difference in the delay between these two paths:

- Data pin to IOE register input
- Clock input pin to the PLL PFD input

The Stratix IV PLL can compensate multiple pad-to-input-register paths, such as a data bus when it is set to use source-synchronous compensation mode. You can use the "PLL Compensation" assignment in the Quartus II software Assignment Editor to select which input pins will be used as the PLL compensation targets. You can include your entire data bus, provided the input registers are clocked by the same output of a source-synchronous compensated PLL. In order for the clock delay to be properly compensated, all of the input pins should be on the same side of the device. The PLL will compensate for the input pin with the longest pad-to-register delay among all input pins in the compensated bus.

If you do not assign the "PLL Compensation" assignment, the Quartus II software will automatically select all of the pins driven by the compensated output of the PLL as the compensation target.

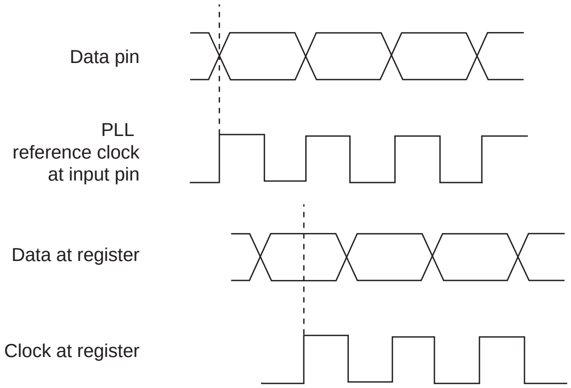
Source-Synchronous Mode for LVDS Compensation

The goal of this mode is to maintain the same data and clock timing relationship seen at the pins of the internal SERDES capture register, except that the clock is inverted (180-degree phase shift). Thus, this mode ideally compensates for the delay of the LVDS clock network plus any difference in delay between these two paths:

- Data pin-to-SERDES capture register
- Clock input pin-to-SERDES capture register. In addition, the output counter needs to provide the 180-degree phase shift.

Figure 5–21 shows an example waveform of the clock and data in LVDS mode.

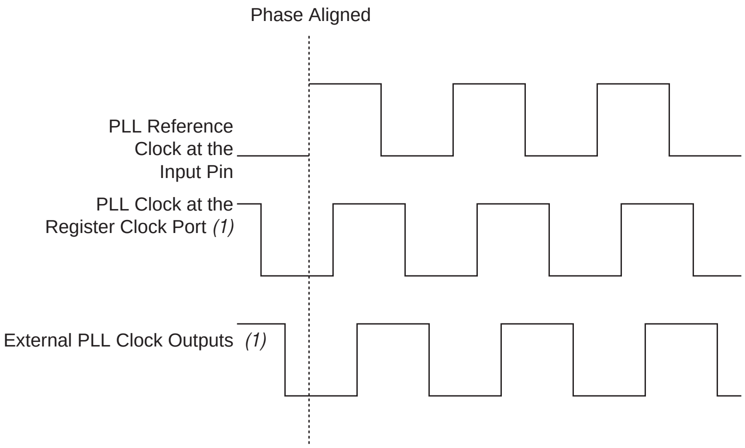
Figure 5–21. Phase Relationship Between Clock and Data in LVDS Mode



No-Compensation Mode

In the no-compensation mode, the PLL does not compensate for any clock networks. This mode provides better jitter performance because the clock feedback into the PFD passes through less circuitry. Both the PLL internal- and external-clock outputs are phase-shifted with respect to the PLL clock input. Figure 5–22 shows an example waveform of the PLL clocks' phase relationship in this mode.

Figure 5–22. Phase Relationship Between PLL Clocks in No Compensation Mode


Phase Aligned
PLL Reference Clock at the Input Pin
PLL Clock at the Register Clock Port *(1)*
External PLL Clock Outputs *(1)*

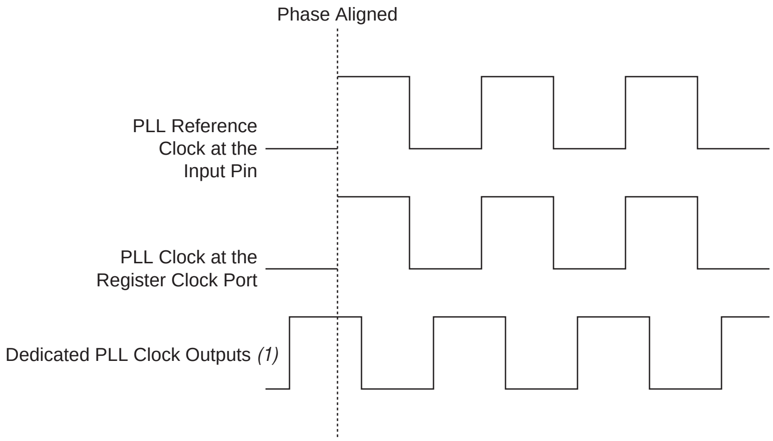
Note to Figure 5–22

(1) The PLL clock outputs will lag the PLL input clocks depending on routine delays.

Normal Mode

An internal clock in normal mode is phase-aligned to the input clock pin. The external clock-output pin has a phase delay relative to the clock input pin if connected in this mode. The Quartus II software timing analyzer reports any phase difference between the two. In normal mode, the delay introduced by the GCLK or RCLK network is fully compensated. Figure 5–23 shows an example waveform of the PLL clocks' phase relationship in this mode.

Figure 5–23. Phase Relationship Between PLL Clocks in Normal Mode


Phase Aligned
PLL Reference Clock at the Input Pin
PLL Clock at the Register Clock Port
Dedicated PLL Clock Outputs *(1)*

Note to Figure 5–23:

(1) The external clock output can lead or lag the PLL internal clock signals.

Zero-Delay Buffer Mode

In zero-delay buffer (ZDB) mode, the external clock output pin is phase-aligned with the clock input pin for zero delay through the device. When using this mode, you must use the same I/O standard on the input clocks and output clocks in order to guarantee clock alignment at the input and output pins. This mode is supported on all Stratix IV PLLs.

When using Stratix IV PLLs in ZDB mode, along with single-ended I/O standards, to ensure phase alignment between the clock input pin (`CLK`) and the external clock output (`CLKOUT`) pin, you are required to instantiate a bidirectional I/O pin in the design to serve as the feedback path connecting the `FBOUT` and `FBIN` ports of the PLL. The PLL uses this bidirectional I/O pin to mimic, and hence compensate for, the output delay from the clock output port of the PLL to the external clock output pin. Figure 5–24 shows ZDB mode implementation in Stratix IV PLLs. You cannot use differential I/O standards on the PLL clock input or output pins when using ZDB mode.

☞ The bidirectional I/O pin that you instantiate in your design should always be assigned a single-ended I/O standard.

Figure 5–24. Zero-Delay Buffer Mode in Stratix IV PLLs

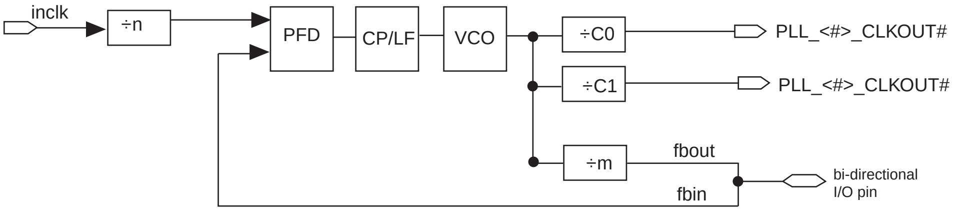
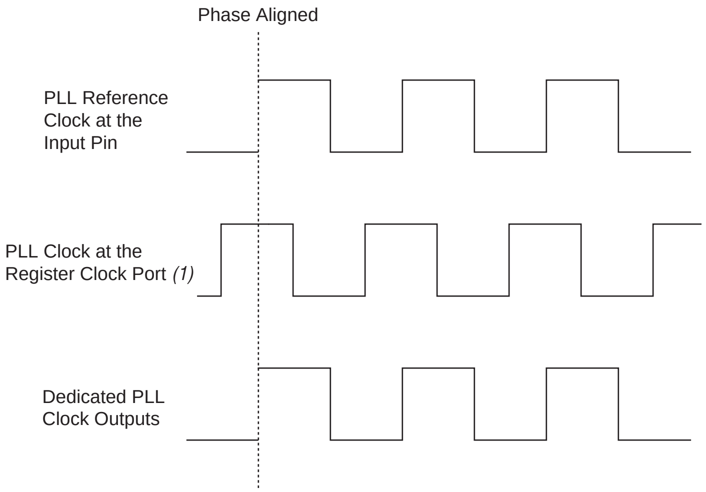


Figure 5–25 shows an example waveform of the PLL clocks' phase relationship in ZDB mode.

Figure 5–25. Phase Relationship Between PLL Clocks in Zero Delay Buffer Mode



Note to Figure 5–25:

(1) The internal PLL clock output can lead or lag the external PLL clock outputs.

External Feedback Mode

In external-feedback (EFB) mode, the external-feedback input pin (`fbin`) is phase-aligned with the clock input pin, as shown in Figure 5–27. Aligning these clocks allows you to remove clock delay and skew between devices. This mode is supported on all Stratix IV PLLs.

In this mode, the output of the `M` counter (`FBOUT`) feeds back to the PLL `fbin` input (using a trace on the board) becoming part of the feedback loop. Also, you use one of the dual-purpose external clock outputs as the `fbin` input pin in EFB mode.

When using this mode, you must use the same I/O standard on the input clock, feedback input, and output clocks. Left/Right PLLs support EFB mode when using single-ended I/O standards only. Figure 5–26 shows EFB mode implementation in Stratix IV devices.

Figure 5–26. External Feedback Mode in Stratix IV Devices

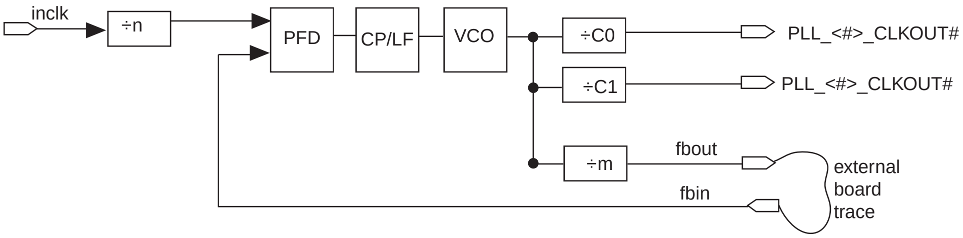
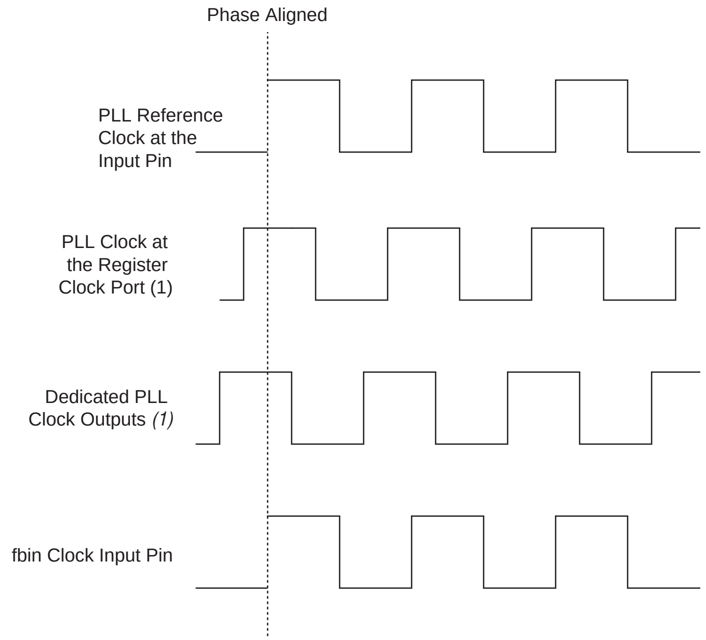


Figure 5–27 shows an example waveform of the phase relationship between PLL clocks in EFB mode.

Figure 5–27. Phase Relationship Between PLL Clocks in External-Feedback Mode


Phase Aligned
PLL Reference Clock at the Input Pin
PLL Clock at the Register Clock Port (1)
Dedicated PLL Clock Outputs (1)
fbin Clock Input Pin

Note to Figure 5–27:

(1) The PLL clock outputs can lead or lag the fbin clock input.

Clock Multiplication and Division

Each Stratix IV PLL provides clock synthesis for PLL output ports using *m*/(n* post-scale counter) scaling factors. The input clock is divided by a pre-scale factor, n, and is then multiplied by the *m* feedback factor. The control loop drives the VCO to match f_{in} (m/n). Each output port has a unique post-scale counter that divides down the high-frequency VCO. For multiple PLL outputs with different frequencies, the VCO is set to the least common multiple of the output frequencies that meets its frequency specifications. For example, if the output frequencies required from one PLL are 33 and 66 MHz, then the Quartus II software sets the VCO to 660 MHz (the least common multiple of 33 and 66 MHz within the VCO range). Then the post-scale counters scale down the VCO frequency for each output port.

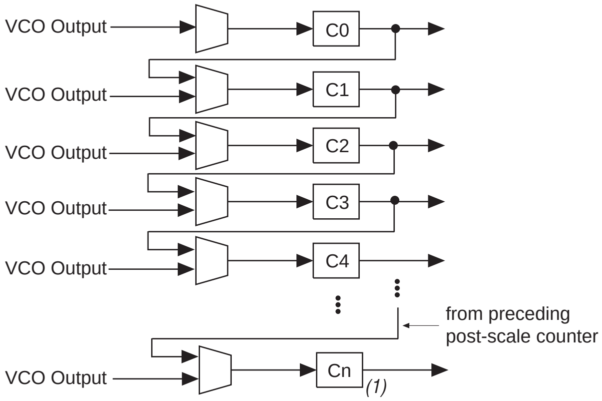
Each PLL has one pre-scale counter, n, and one multiply counter, m, with a range of 1 to 512 for both m and n. The n counter does not use duty-cycle control because the only purpose of this counter is to calculate frequency division. There are seven generic post-scale counters per Left/Right PLL and ten post-scale counters per Top/Bottom PLL that can feed GCLKs, RCLKs, or external clock outputs. These post-scale counters range from 1 to 512 with a 50% duty cycle setting. The high- and low-count values for each counter range from 1 to 256. The sum of the high- and low-count values chosen for a design selects the divide value for a given counter.

The Quartus II software automatically chooses the appropriate scaling factors according to the input frequency, multiplication, and division values entered into the ALTPLL megafunction.

Post-Scale Counter Cascading

The Stratix IV PLLs support post-scale counter cascading to create counters larger than 512. This is automatically implemented in the Quartus II software by feeding the output of one `C` counter into the input of the next `C` counter, as shown in Figure 5–28.

Figure 5–28. Counter Cascading



Note to Figure 5–28:

(1) n = 6 or n = 9

When cascading post-scale counters to implement a larger division of the high-frequency VCO clock, the cascaded counters behave as one counter with the product of the individual counter settings. For example, if C0 = 40 and C1 = 20, then the cascaded value is C0 × C1 = 800.

☞ Post-scale counter cascading is set in the configuration file. It cannot be done using PLL reconfiguration.

Programmable Duty Cycle

The programmable duty cycle allows PLLs to generate clock outputs with a variable duty cycle. This feature is supported on the PLL post-scale counters. The duty-cycle setting is achieved by a low and high time-count setting for the post-scale counters. The Quartus II software uses the frequency input and the required multiply or divide rate to determine the duty cycle choices. The post-scale counter value determines the precision of the duty cycle. The precision is defined as 50% divided by the post-scale counter value. For example, if the `C0` counter is 10, then steps of 5% are possible for duty-cycle choices between 5% to 90%.

If the PLL is in external feedback mode, you must set the duty cycle for the counter driving the `fbin` pin to 50%. Combining the programmable duty cycle with programmable phase shift allows the generation of precise non-overlapping clocks.

Programmable Phase Shift

Phase shift is used to implement a robust solution for clock delays in Stratix IV devices. Phase shift is implemented by using a combination of the VCO phase output and the counter starting time. The VCO phase output and counter starting time is the most accurate method of inserting delays, since it is purely based on counter settings, which are independent of process, voltage, and temperature.

You can phase-shift the output clocks from the Stratix IV PLLs in either of these two resolutions:

- Fine resolution using VCO phase taps
- Coarse resolution using counter starting time

Fine-resolution phase shifts are implemented by allowing any of the output counters (`C[n..0]`) or the `m` counter to use any of the eight phases of the VCO as the reference clock. This allows you to adjust the delay time with a fine resolution. The minimum delay time that you can insert using this method is defined by:

Equation 5–1.

$$\Phi_{fine} = \frac{1}{8}T_{VCO} = \frac{1}{8f_{VCO}} = \frac{N}{8Mf_{REF}}$$

where f_{REF} is the input reference clock frequency.

For example, if f_{REF} is 100 MHz, n is 1, and m is 8, then f_{VCO} is 800 MHz and Φ_{fine} equals 156.25 ps. This phase shift is defined by the PLL operating frequency, which is governed by the reference clock frequency and the counter settings.

Coarse-resolution phase shifts are implemented by delaying the start of the counters for a predetermined number of counter clocks. You can express coarse phase shift as:

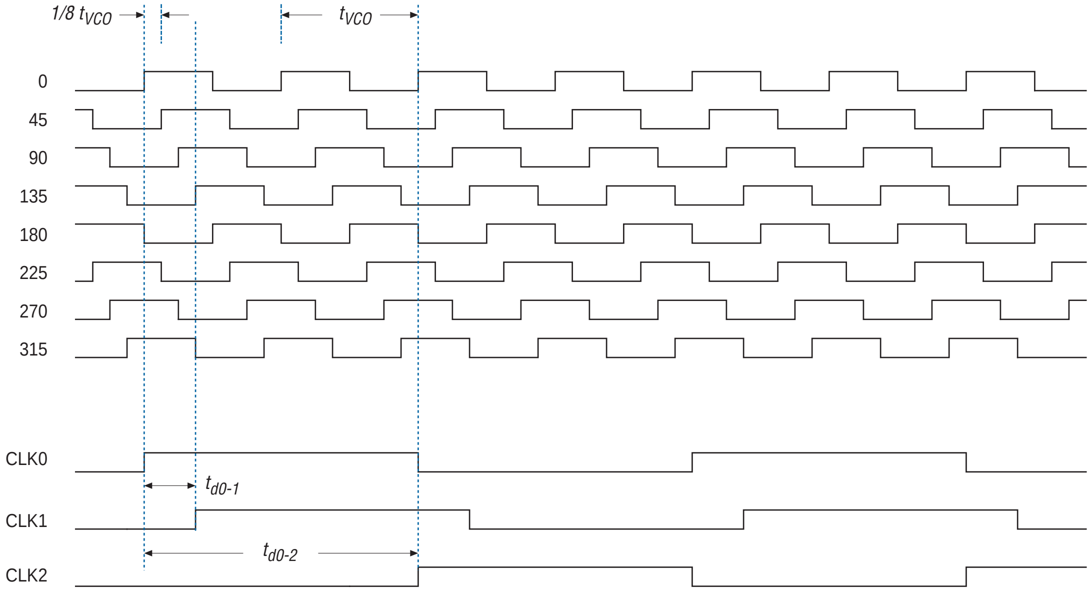
Equation 5–2.

$$\Phi_{coarse} = \frac{C-1}{f_{VCO}} = \frac{(C-1)N}{Mf_{REF}}$$

where C is the count value set for the counter delay time, (this is the initial setting in the PLL usage section of the compilation report in the Quartus II software). If the initial value is 1, C – 1 = 0° phase shift.

Figure 5–29 shows an example of phase-shift insertion with the fine resolution using the VCO phase-taps method. The eight phases from the VCO are shown and labeled for reference. For this example, `CLK0` is based off the `0phase` from the VCO and has the C value for the counter set to one. The `CLK1` signal is divided by four, two VCO clocks for high time and two VCO clocks for low time. `CLK1` is based off the 135° phase tap from the VCO and also has the C value for the counter set to one. In this case, the two clocks are offset by 3 Φ_{FINE}. `CLK2` is based off the `0phase` from the VCO but has the C value for the counter set to three. This arrangement creates a delay of 2 Φ_{COARSE} (two complete VCO periods).

Figure 5–29. Delay Insertion Using VCO Phase Output and Counter Delay Time



You can use the coarse- and fine-phase shifts to implement clock delays in Stratix IV devices.

Stratix IV devices support dynamic phase-shifting of VCO phase taps only. The phase shift is reconfigurable any number of times, and each phase shift takes about one `SCANCLK` cycle, allowing you to implement large phase shifts quickly.

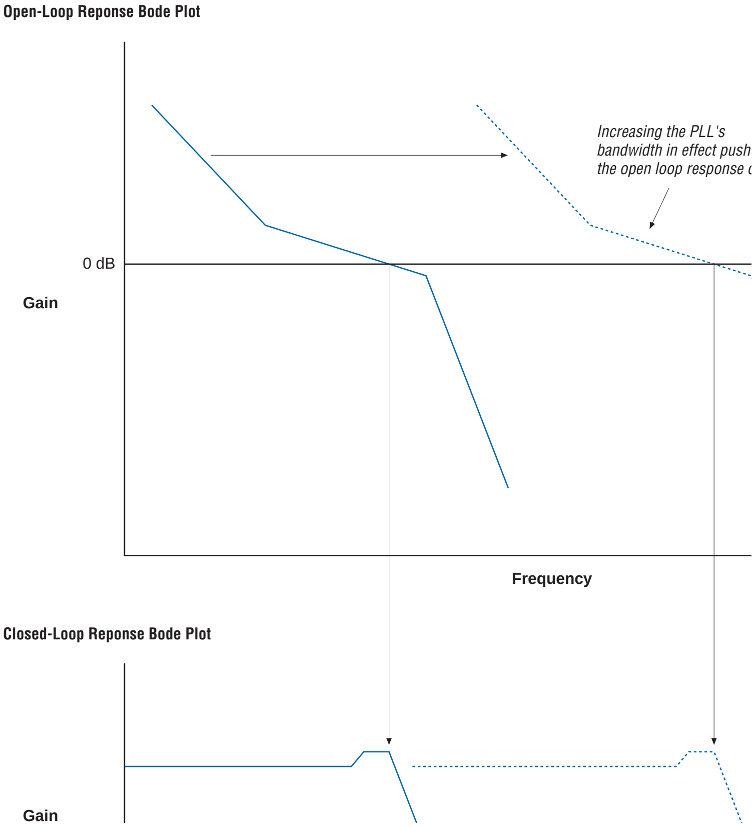
Programmable Bandwidth

Stratix IV PLLs provide advanced control of the PLL bandwidth using the PLL loop's programmable characteristics, including loop filter and charge pump.

Background

PLL bandwidth is the measure of the PLL's ability to track the input clock and its associated jitter. The closed-loop gain 3-dB frequency in the PLL determines the PLL bandwidth. The bandwidth is approximately the unity gain point for open loop PLL response. As Figure 5–30 shows, these points correspond to approximately the same frequency. Stratix IV PLLs provide three bandwidth settings—low, medium (default), and high.

Figure 5–30. Open- and Closed-Loop Response Bode Plots



A high-bandwidth PLL provides a fast lock time and tracks jitter on the reference clock source, passing it through to the PLL output. A low-bandwidth PLL filters out reference clock jitter but increases lock time. Stratix IV PLLs allow you to control the bandwidth over a finite range to customize the PLL characteristics for a particular application. The programmable bandwidth feature in Stratix IV PLLs benefits applications requiring clock switchover.

A high-bandwidth PLL can benefit a system that needs to accept a spread-spectrum clock signal. Stratix IV PLLs can track a spread-spectrum clock by using a high-bandwidth setting. Using a low-bandwidth in this case could cause the PLL to filter out the jitter on the input clock.

A low-bandwidth PLL can benefit a system using clock switchover. When the clock switchover happens, the PLL input temporarily stops. A low-bandwidth PLL reacts more slowly to changes on its input clock and takes longer to drift to a lower frequency (caused by the input stopping) than a high-bandwidth PLL.

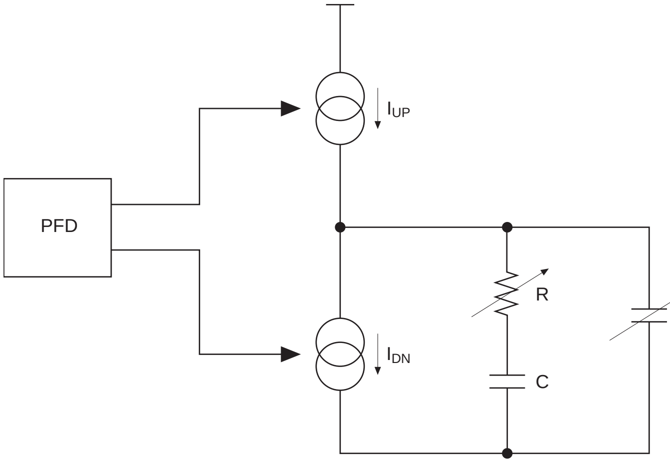
Implementation

Traditionally, external components such as the VCO or loop filter control a PLL's bandwidth. Most loop filters consist of passive components such as resistors and capacitors that take up unnecessary board space and increase cost. With Stratix IV PLLs, all the components are contained within the device to increase performance and decrease cost.

When you specify the bandwidth setting (low, medium, or high) in the ALTPLL Megawizard Plug-in Manager, the Quartus II software automatically sets the corresponding charge pump and loop filter (Icp, R, C) values to achieve the desired bandwidth range.

Figure 5–31 shows the loop filter and the components that you can set using the Quartus II software. The components are the loop filter resistor, R, the high frequency capacitor, CH, and the charge pump current, I_{UP} or I_{DN}.

Figure 5–31. Loop Filter Programmable Components



Spread-Spectrum Tracking

Stratix IV devices can accept a spread-spectrum input with typical modulation frequencies. However, the device cannot automatically detect that the input is a spread-spectrum signal. Instead, the input signal looks like deterministic jitter at the input of the PLL. Stratix IV PLLs can track a spread-spectrum input clock as long as it is within the input-jitter tolerance specifications. Stratix IV devices cannot internally generate spread-spectrum clocks.

Clock Switchover

The clock switchover feature allows the PLL to switch between two reference input clocks. Use this feature for clock redundancy or for a dual-clock domain application such as in a system that turns on the redundant clock if the previous clock stops running. The design can perform clock switchover automatically, when the clock is no longer toggling or based on a user control signal, clkswitch.

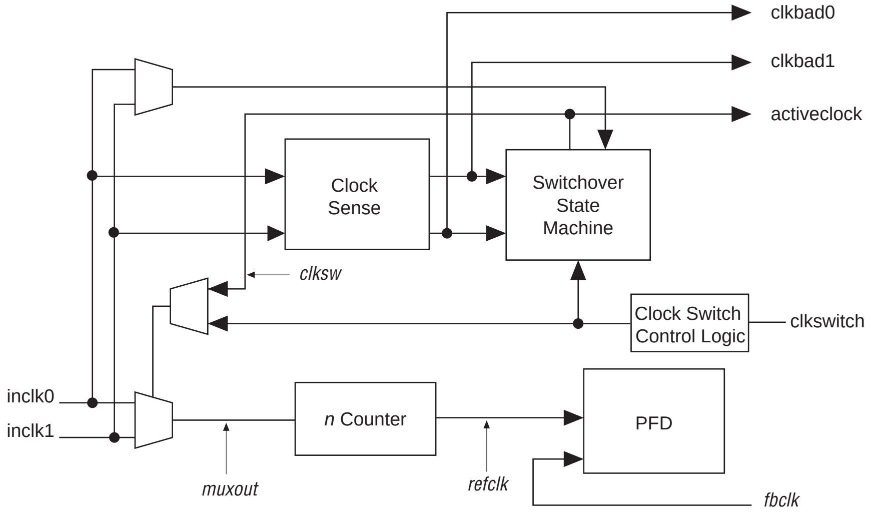
The following clock switchover modes are supported in Stratix IV PLLs:

- Automatic switchover: The clock sense circuit monitors the current reference clock and if it stops toggling, automatically switches to the other clock inclk0 or inclk1.
- Manual clock switchover: Clock switchover is controlled via the clkswitch signal in this mode. When the clkswitch signal goes from logic low to logic high, and stays high for at least three clock cycles, the reference clock to the PLL is switched from inclk0 to inclk1, or vice-versa.

- Automatic switchover with manual override: This mode combines Modes 1 and 2. When the `clkswitch` signal goes high, it overrides automatic clock switchover mode.

Stratix IV PLLs support a fully configurable clock switchover capability. Figure 5–32 shows the block diagram of the automatic switchover circuit built into the PLL. When the current reference clock is not present, the clock sense block automatically switches to the backup clock for PLL reference. The clock switchover circuit also sends out three status signals—`clkbad[0]`, `clkbad[1]`, and `activeclock`—from the PLL to implement a custom switchover circuit in the logic array. You can select a clock source as the backup clock by connecting it to the `inclk1` port of the PLL in your design.

Figure 5–32. Automatic Clock Switchover Circuit Block Diagram



Automatic Clock Switchover

Use the switchover circuitry to automatically switch between `inclk0/inclk1` when the current reference clock to the PLL stops toggling. For example, in applications that require a redundant clock with the same frequency as the reference clock, the switchover state machine generates a signal (`clksw`) that controls the multiplexer select input as shown in Figure 5–32. In this case, `inclk1` becomes the reference clock for the PLL. When using the automatic switchover mode, you can switch back and forth between `inclk0` and `inclk1` clocks any number of times, when one of the two clocks fails and the other clock is available.

When using the automatic clock switchover mode, the following requirements need to be satisfied:

- Both clock inputs need to be running.
- The period of the two clock inputs can differ by no more than 100% (2×).

If the current clock input stops toggling while the other clock is also not toggling, switchover will not be initiated and the `clkbad[0:1]` signals will not be valid. Also, if both clock inputs are not the same frequency, but their period difference is within 100%, the clock sense block will detect when a clock stops toggling, but the PLL may lose lock after the switchover is completed and need time to relock.

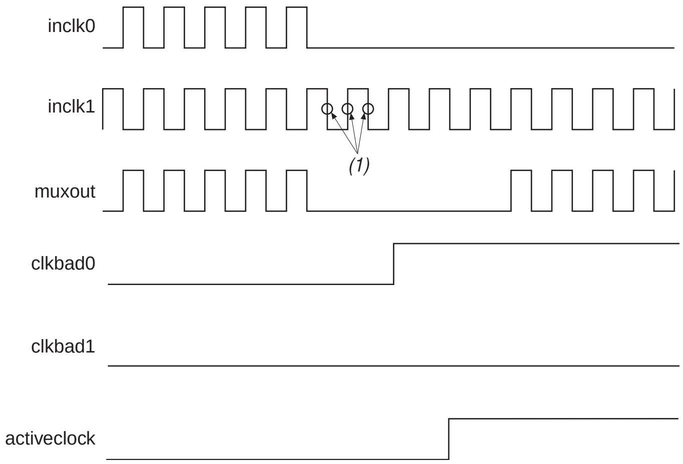
☞ Altera recommends resetting the PLL using the `areset` signal to maintain the phase relationships between the PLL input and output clocks when using clock switchover.

When using automatic switchover mode, the `clkbad[0]` and `clkbad[1]` signals indicate the status of the two clock inputs. When they are asserted, the clock sense block has detected that the corresponding clock input has stopped toggling. These two signals are not valid if the frequency difference between `inclk0` and `inclk1` is greater than 20%.

The `activeclock` signal indicates which of the two clock inputs (`inclk0` or `inclk1`) is being selected as the reference clock to the PLL. When the frequency difference between the two clock inputs is more than 20%, the `activeclock` signal is the only valid status signal.

Figure 5–33 shows an example waveform of the switchover feature when using the automatic switchover mode. In this example, the `inclk0` signal is stuck low. After the `inclk0` signal is stuck at low for approximately two clock cycles, the clock sense circuitry drives the `clkbad[0]` signal high. Also, because the reference clock signal is not toggling, the switchover state machine controls the multiplexer through the `clksw` signal to switch to the backup clock, `inclk1`.

Figure 5–33. Automatic Switchover Upon Loss of Clock Detection



Note to Figure 5–33:

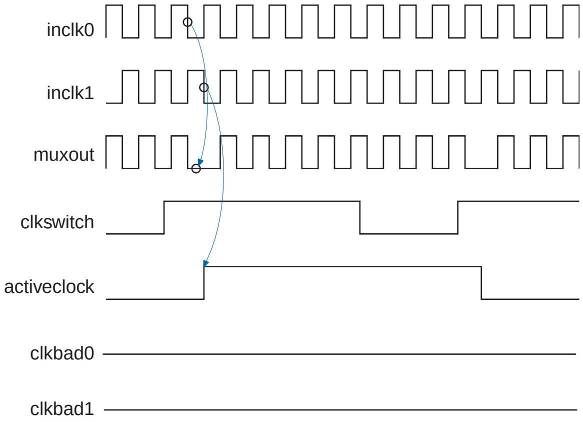
(1) Switchover is enabled on the falling edge of `inclk0` or `inclk1`, depending on which clock is available. In this figure, switchover is enabled on the falling edge of `inclk1`.

Manual Override

In the automatic switchover with manual override mode, you can use the `clkswitch` input for user- or system-controlled switch conditions. You can use this mode for same-frequency switchover, or to switch between inputs of different frequencies. For example, if `inclk0` is 66 MHz and `inclk1` is 200 MHz, you must control the switchover using `clkswitch` because the automatic clock-sense circuitry cannot monitor clock input (`inclk0`, `inclk1`) frequencies with a frequency difference of more than 100% (2×). This feature is useful when the clock sources originate from multiple cards on the backplane, requiring a system-controlled switchover between the frequencies of operation. You should choose the backup clock frequency and set the m, n, c, and k counters accordingly so the VCO operates within the recommended operating frequency range of 600 to 1,300 MHz. The ALTPLL Megawizard Plug-in Manager notifies users if a given combination of inclk0 and inclk1 frequencies cannot meet this requirement.

Figure 5–34 shows an example of a waveform illustrating the switchover feature when controlled by clkswitch. In this case, both clock sources are functional and inclk0 is selected as the reference clock; clkswitch goes high, which starts the switchover sequence. On the falling edge of inclk0, the counter's reference clock, muxout, is gated off to prevent any clock glitching. On the falling edge of inclk1, the reference clock multiplexer switches from inclk0 to inclk1 as the PLL reference, and the activeclock signal changes to indicate which clock is currently feeding the PLL.

Figure 5–34. Clock Switchover Using the clkswitch (Manual) Control

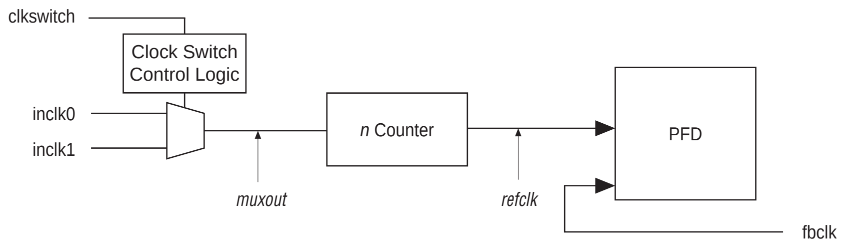


In this mode, the activeclock signal mirrors the clkswitch signal. As both clocks are still functional during the manual switch, neither clkbad signal goes high. Since the switchover circuit is positive-edge sensitive, the falling edge of the clkswitch signal does not cause the circuit to switch back from inclk1 to inclk0. When the clkswitch signal goes high again, the process repeats. clkswitch and automatic switch only work if the clock being switched to is available. If the clock is not available, the state machine waits until the clock is available.

Manual Clock Switchover

In manual clock switchover mode, the clkswitch signal controls whether inclk0 or inclk1 is selected as the input clock to the PLL. By default, inclk0 is selected. A low-to-high transition on clkswitch and clkswitch being held high for at least three inclk cycles initiates a clock switchover event. You must bring clkswitch back low again in order to perform another switchover event in the future. If you do not require another switchover event in the future, you can leave clkswitch in a logic high state after the initial switch. Pulsing clkswitch high for at least three inclk cycles performs another switchover event. If inclk0 and inclk1 are different frequencies and are always running, the clkswitch minimum high time must be greater than or equal to three of the slower frequency inclk0/inclk1 cycles. Figure 5–35 shows the block diagram of the manual switchover circuit.

Figure 5–35. Manual Clock Switchover Circuitry in Stratix IV PLLs



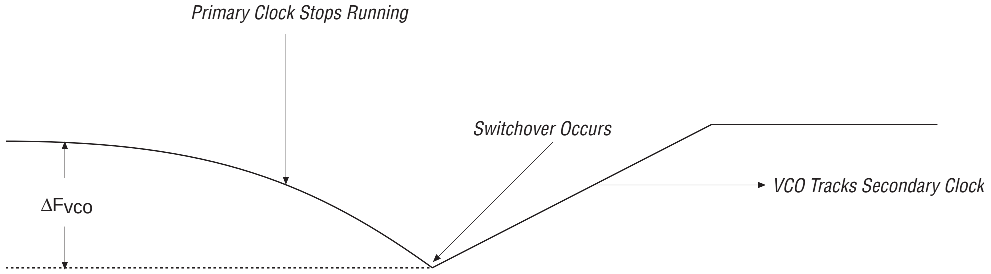
For more information on PLL software support in the Quartus II software, refer to the *ALTPLL Megafunction User Guide*.

Guidelines

Use the following guidelines when implementing clock switchover in Stratix IV PLLs.

- Automatic clock switchover requires that the `inclk0` and `inclk1` frequencies be within 100% (2×) of each other. Failing to meet this requirement causes the `clkbad[0]` and `clkbad[1]` signals to not function properly.
- When using manual clock switchover, the difference between `inclk0` and `inclk1` can be more than 100% (2×). However, differences in frequency and/or phase of the two clock sources will likely cause the PLL to lose lock. Resetting the PLL ensures that the correct phase relationships are maintained between input and output clocks.
- Applications that require a clock switchover feature and a small frequency drift should use a low-bandwidth PLL. The low-bandwidth PLL reacts more slowly than a high-bandwidth PLL to reference input clock changes. When the switchover happens, a low-bandwidth PLL propagates the stopping of the clock to the output more slowly than a high-bandwidth PLL. However, be aware that the low-bandwidth PLL also increases lock time.
- After a switchover occurs, there may be a finite resynchronization period for the PLL to lock onto a new clock. The exact amount of time it takes for the PLL to re-lock depends on the PLL configuration.
- The phase relationship between the input clock to the PLL and the output clock from the PLL is important in your design. Assert `areset` for at least 10 ns after performing a clock switchover. Wait for the locked signal to go high and be stable before re-enabling the output clocks from the PLL.
- Figure 5–36 shows how the VCO frequency gradually decreases when the current clock is lost and then increases as the VCO locks on to the backup clock.

Figure 5–36. VCO Switchover Operating Frequency



- Disable the system during clock switchover if it is not tolerant of frequency variations during the PLL resynchronization period. You can use the `clkbad[0]` and `clkbad[1]` status signals to turn off the PFD (`PFDENA = 0`) so the VCO maintains its most recent frequency. You can also use the state machine to switch over to the secondary clock. When the PFD is re-enabled, output clock-enable signals (`clkena`) can disable clock outputs during the switchover and resynchronization period. Once the lock indication is stable, the system can re-enable the output clock(s).

PLL Reconfiguration

Phase-locked loops (PLLs) use several divide counters and different voltage-controlled oscillator (VCO) phase taps to perform frequency synthesis and phase shifts. In Stratix IV PLLs, you can reconfigure both the counter settings and phase-shift the PLL output clock in real time. You can also change the charge pump and loop-filter components, which dynamically affects the PLL bandwidth. You can use these PLL components to update the output-clock frequency and the PLL bandwidth and to phase-shift in real time, without reconfiguring the entire Stratix IV device.

The ability to reconfigure the PLL in real time is useful in applications that operate at multiple frequencies. It is also useful in prototyping environments, allowing you to sweep PLL output frequencies and adjust the output-clock phase dynamically. For instance, a system generating test patterns is required to generate and transmit patterns at 75 or 150 MHz, depending on the requirements of the device under test. Reconfiguring the PLL components in real time allows you to switch between two such output frequencies within a few microseconds. You can also use this feature to adjust clock-to-out (`tCO`) delays in real time by changing the PLL output clock phase shift. This approach eliminates the need to regenerate a configuration file with the new PLL settings.

PLL Reconfiguration Hardware Implementation

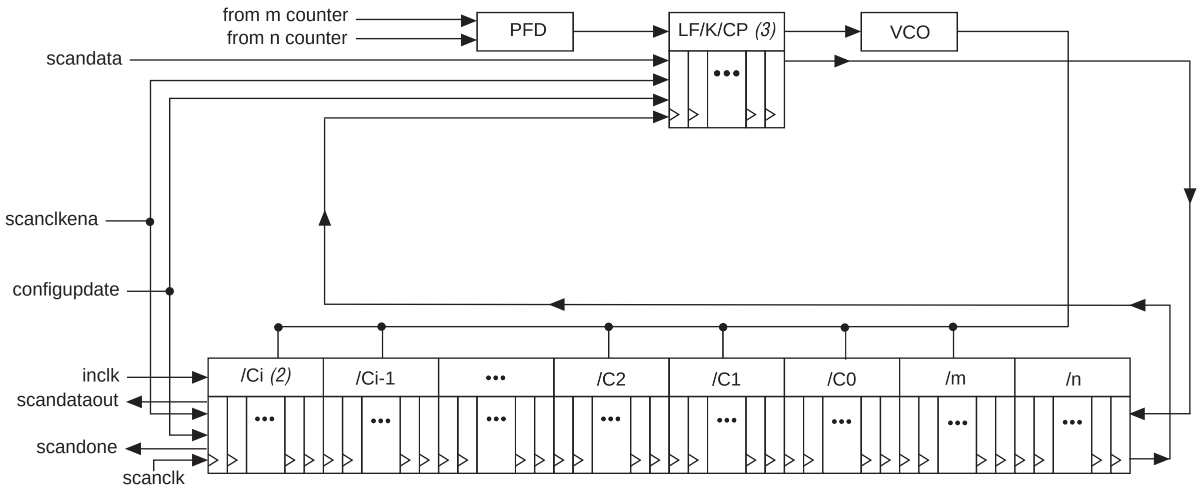
The following PLL components are reconfigurable in real time:

- Pre-scale counter (`n`)
- Feedback counter (`m`)
- Post-scale output counters (`C0 - C9`)
- Post VCO Divider (`K`)

- Dynamically adjust the charge-pump current (`Icp`) and loop-filter components (`R`, `C`) to facilitate reconfiguration of the PLL bandwidth

Figure 5–37 shows how PLL counter settings can be dynamically adjusted by shifting their new settings into a serial shift-register chain or scan chain. Serial data is input to the scan chain via the `scandataport` and shift registers are clocked by `scanclk`. The maximum `scanclk` frequency is 100 MHz. Serial data is shifted through the scan chain as long as the `scanclkena` signal stays asserted. After the last bit of data is clocked, asserting the `configupdate` signal for at least one `scanclk` clock cycle causes the PLL configuration bits to be synchronously updated with the data in the scan registers.

Figure 5–37. PLL Reconfiguration Scan Chain *(Note 1)*



Notes to Figure 5–37:

(1) The Stratix IV Left/Right PLLs support `C0 - C6` counters.

(2) i = 6 or i = 9.

(3) This figure shows the corresponding scan register for the K counter in between the scan registers for the charge pump and loop filter. The K counter is physically located after the VCO.

The counter settings are updated synchronously to the clock frequency of the individual counters. Therefore, all counters are not updated simultaneously.

Table 5–10 shows how these signals can be driven by the programmable logic device (PLD) logic array or I/O pins.

Table 5–10. Real-Time PLL Reconfiguration Ports (Part 1 of 2)

PLL Port Name	Description	Source	Destination
`scandata`	Serial input data stream to scan chain.	Logic array or I/O pin	PLL reconfiguration circuit
`scanclk`	Serial clock input signal. This clock can be free running.	GCLK/RCLK or I/O pins	PLL reconfiguration circuit
`scanclkena`	Enables `scanclk` and allows the `scandata` to be loaded in the scan chain. Active high	Logic array or I/O pin	PLL reconfiguration circuit

Table 5–10. Real-Time PLL Reconfiguration Ports (Part 2 of 2)

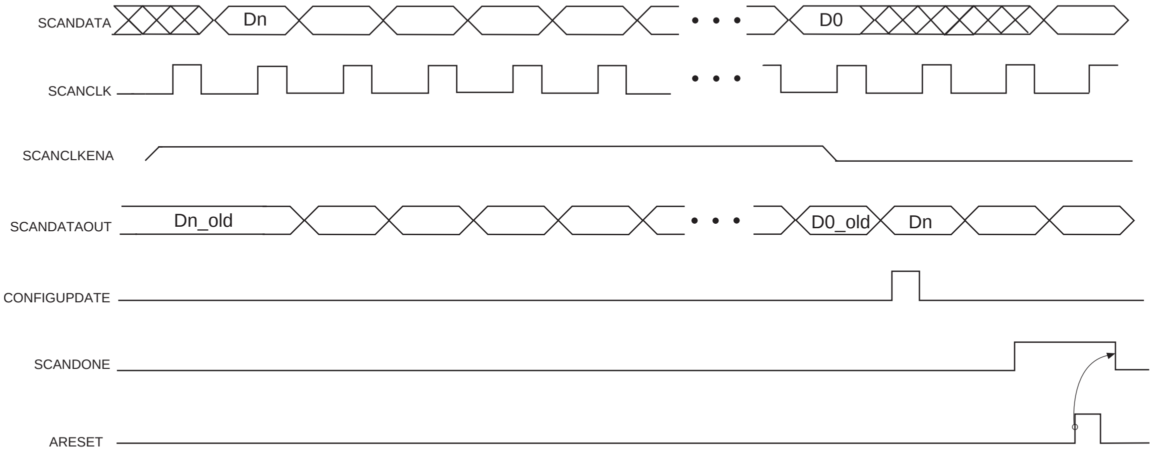
PLL Port Name	Description	Source	Destination
`configupdate`	Writes the data in the scan chain to the PLL. Active high	Logic array or I/O pin	PLL reconfiguration circuit
`scandone`	Indicates when the PLL has finished reprogramming. A rising edge indicates the PLL has begun reprogramming. A falling edge indicated the PLL has finished reprogramming.	PLL reconfiguration circuit	Logic array or I/O pins
`scandataout`	Used to output the contents of the scan chain.	PLL reconfiguration circuit	Logic array or I/O pins

The procedure to reconfigure the PLL counters is shown below:

1. The `scanclkena` signal is asserted at least one `scanclk` cycle prior to shifting in the first bit of `scandata` (`Dn`).
2. Serial data (`scandata`) is shifted into the scan chain on the 2nd rising edge of `scanclk`.
3. After all 234 bits (Top/Bottom PLLs) or 180 bits (Left/Right PLLs) have been scanned into the scan chain, the `scanclkena` signal is de-asserted to prevent inadvertent shifting of bits in the scan chain.
4. The `configupdate` signal is asserted for one `scanclk` cycle to update the PLL counters with the contents of the scan chain.
5. The `scandone` signal goes high indicating the PLL is being reconfigured. A falling edge indicates the PLL counters have been updated with new settings.
6. Reset the PLL using the `areset` signal if you make any changes to the `M` or `N` counters or to the `Icp`, `R`, or `C` settings.
7. Steps 1-5 can be repeated to reconfigure the PLL any number of times.

Figure 5–38 shows a functional simulation of the PLL reconfiguration feature.

Figure 5–38. PLL Reconfiguration Waveform



☞ When you reconfigure the counter clock frequency, you cannot reconfigure the corresponding counter phase shift settings using the same interface. Instead, reconfigure the phase shifts in real time using the dynamic phase shift reconfiguration interface. If you reconfigure the counter frequency, but wish to keep the same non-zero phase shift setting (for example, 90 degrees) on the clock output, you must reconfigure the phase shift immediately after reconfiguring the counter clock frequency.

Post-Scale Counters (C0 to C9)

The multiply or divide values and duty cycle of post-scale counters can be reconfigured in real time. Each counter has an 8-bit high-time setting and an 8-bit low-time setting. The duty cycle is the ratio of output high- or low-time to the total cycle time, which is the sum of the two. Additionally, these counters have two control bits, `rbypass`, for bypassing the counter, and `rselodd`, to select the output clock duty cycle.

When the `rbypass` bit is set to 1, it bypasses the counter, resulting in a divide by 1. When this bit is set to 0, the high- and low-time counters are added to compute the effective division of the VCO output frequency. For example, if the post-scale divide factor is 10, the high- and low-count values could be set to 5 and 5, respectively, to achieve a 50-50% duty cycle. The PLL implements this duty cycle by transitioning the output clock from high to low on the rising edge of the VCO output clock. However, a 4 and 6 setting for the high- and low-count values, respectively, would produce an output clock with 40-60% duty cycle.

The `rselodd` bit indicates an odd divide factor for the VCO output frequency along with a 50% duty cycle. For example, if the post-scale divide factor is 3, the high- and low-time count values could be set to 2 and 1, respectively, to achieve this division. This implies a 67% - 33% duty cycle. If you need a 50% - 50% duty cycle, you can set the `rselodd` control bit to 1 to achieve this duty cycle despite an odd division factor. The PLL implements this duty cycle by transitioning the output clock from high to low on a falling edge of the VCO output clock. When you set `rselodd` = 1, you subtract 0.5 cycles from the high time and you add 0.5 cycles to the low time. For example:

- High-time count = 2 cycles
- Low-time count = 1 cycle
- `rselodd` = 1 effectively equals:
  - High-time count = 1.5 cycles
  - Low-time count = 1.5 cycles
  - Duty cycle = (1.5/3) % high-time count and (1.5/3) % low-time count

Scan Chain Description

The length of the scan chain varies for different Stratix IV PLLs. The Top/Bottom PLLs have 10 post-scale counters and a 234-bit scan chain, while the Left/Right PLLs have 7 post-scale counters and a 180-bit scan chain. Table 5–11 shows the number of bits for each component of a Stratix IV PLL.

Table 5–11. Top/Bottom PLL Reprogramming Bits

Block Name	Number of Bits		Total
	Counter	**Other** *(1)*	
C9 *(2)*	16	2	18
C8	16	2	18
C7	16	2	18
C6 *(3)*	16	2	18
C5	16	2	18
C4	16	2	18
C3	16	2	18
C2	16	2	18
C1	16	2	18
C0	16	2	18
M	16	2	18
N	16	2	18
Charge Pump Current	0	3	3
VCO Post-Scale divider (K)	1	0	1
Loop Filter Capacitor *(4)*	0	2	2
Loop Filter Resistor	0	5	5
Unused CP/LF	0	7	7
Total number of bits	—	—	234

Notes to Table 5–11:

(1) Includes two control bits, `rbypass`, for bypassing the counter, and `rselodd`, to select the output clock duty cycle.

(2) LSB bit for C9 low-count value is the first bit shifted into the scan chain for Top/Bottom PLLs.

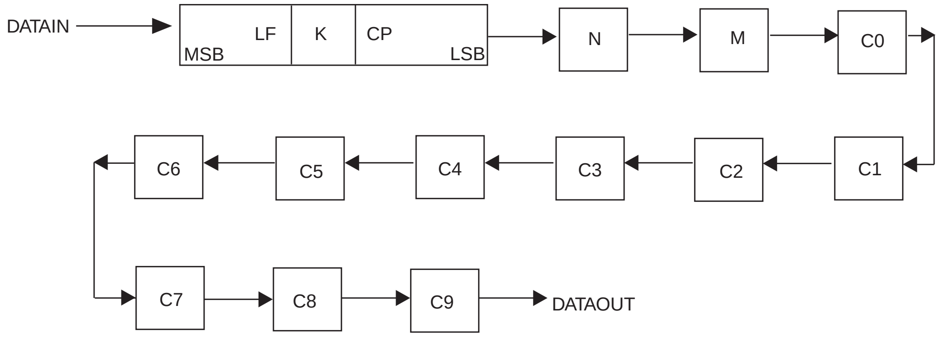
(3) LSB bit for C6 low-count value is the first bit shifted into the scan chain for Left/Right PLLs.

(4) MSB bit for loop filter is the last bit shifted into the scan chain.

Table 5–11 shows the scan chain order of PLL components for Top/Bottom PLLs, which have 10 post-scale counters. The order of bits is the same for the Left/Right PLLs, but the reconfiguration bits start with the C6 post-scale counter.

Figure 5–39 shows the scan-chain order of PLL components for the Top/Bottom PLLs.

Figure 5–39. Scan-Chain Order of PLL Components for Top/Bottom PLLs *(Note 1)*

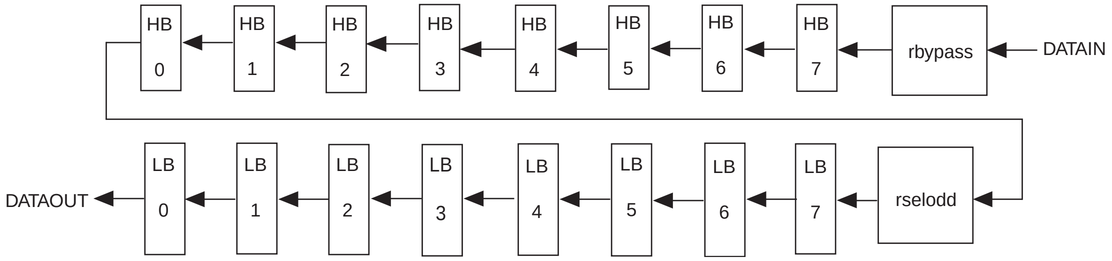


Note to Figure 5–39:

(1) Left/Right PLLs have the same scan-chain order. The post-scale counters end at C6.

Figure 5–40 shows the scan-chain bit-order sequence for post-scale counters in all Stratix IV PLLs.

Figure 5–40. Scan-Chain Bit-Order Sequence for Post-Scale Counters in Stratix IV PLLs



Charge Pump and Loop Filter

You can reconfigure the charge-pump and loop-filter settings to update the PLL bandwidth in real time. Table 5–12, Table 5–13, and Table 5–14 show the possible settings for charge pump current (`Icp`), loop-filter resistor (`R`), and capacitor (`C`) values for Stratix IV PLLs.

Table 5–12. charge_pump_current Bit Settings

CP[2]	CP[1]	CP[0]	Decimal Value for Setting
0	0	0	0
0	0	1	1
0	1	1	3
1	1	1	7

Table 5–13. loop_filter_r Bit Settings

LFR[4]	LFR[3]	LFR[2]	LFR[1]	LFR[0]	Decimal Value for Setting
0	0	0	0	0	0
0	0	0	1	1	3
0	0	1	0	0	4
0	1	0	0	0	8
1	0	0	0	0	16
1	0	0	1	1	19
1	0	1	0	0	20
1	1	0	0	0	24
1	1	0	1	1	27
1	1	1	0	0	28
1	1	1	1	0	30

Table 5–14. loop_filter_c Bit Settings

LFC[1]	LFC[0]	Decimal Value for Setting
0	0	0
0	1	1
1	1	3

Bypassing PLL

Bypassing a PLL counter results in a multiply (`m` counter) or a divide (`n` and `C0` to `C9` counters) factor of one.

Table 5–15 shows the settings for bypassing the counters in Stratix IV PLLs.

Table 5–15. PLL Counter Settings

PLL Scan Chain Bits [0..8] Settings									
LSB *(2)*								MSB *(1)*	Description
0	X	X	X	X	X	X	X	1 *(3)*	PLL counter bypassed
X	X	X	X	X	X	X	X	0 *(3)*	PLL counter not bypassed because bit 10 (`MSB`) is set to 0

Notes to Table 5–15:

(1) Most significant bit (MSB)

(2) Least significant bit (LSB).

(3) Counter-bypass bit.

☞ To bypass any of the PLL counters, set the bypass bit to 1. The values on the other bits are ignored. To bypass the VCO post-scale counter (`K`), set the corresponding bit to 0.

Dynamic Phase-Shifting

The dynamic phase-shifting feature allows the output phases of individual PLL outputs to be dynamically adjusted relative to each other and to the reference clock, without the need to send serial data through the scan chain of the corresponding PLL. This feature simplifies the interface and allows you to quickly adjust clock-to-out (t_{co}) delays by changing the output clock phase-shift in real time. This adjustment is achieved by incrementing or decrementing the VCO phase-tap selection to a given C counter or to the M counter. The phase is shifted by 1/8 of the VCO frequency at a time. The output clocks are active during this phase-reconfiguration process.

Table 5–16 shows the control signals that are used for dynamic phase-shifting.

Table 5–16. Dynamic Phase-Shifting Control Signals

Signal Name	Description	Source	Destination
PHASECOUNTER SELECT[3:0]	Counter select. Four bits decoded to select either the M or one of the C counters for phase adjustment. One address maps to select all C counters. This signal is registered in the PLL on the rising edge of SCANCLK.	Logic array or I/O pins	PLL reconfiguration circuit
PHASEUPDOWN	Selects dynamic phase shift direction; 1= UP; 0 = DOWN. Signal is registered in the PLL on the rising edge of SCANCLK.	Logic array or I/O pin	PLL reconfiguration circuit
PHASESTEP	Logic high enables dynamic phase shifting.	Logic array or I/O pin	PLL reconfiguration circuit
SCANCLK	Free running clock from core used in combination with PHASESTEP to enable/disable dynamic phase shifting. Shared with SCANCLK for dynamic reconfiguration.	GCLK/RCLK or I/O pin	PLL reconfiguration circuit
PHASEDONE	When asserted, it indicates to core-logic that the phase adjustment is complete and PLL is ready to act on a possible second adjustment pulse. Asserts based on internal PLL timing. De-asserts on rising edge of SCANCLK.	PLL reconfiguration circuit	Logic array or I/O pins

Table 5–17 shows the PLL counter selection based on the corresponding PHASECOUNTERSELECT setting.

Table 5–17. Phase Counter Select Mapping (Part 1 of 2)

PHASECOUNTERSELECT[3]	[2]	[1]	[0]	Selects
0	0	0	0	All Output Counters
0	0	0	1	M Counter
0	0	1	0	C0 Counter
0	0	1	1	C1 Counter
0	1	0	0	C2 Counter
0	1	0	1	C3 Counter
0	1	1	0	C4 Counter
0	1	1	1	C5 Counter
1	0	0	0	C6 Counter

Table 5–17. Phase Counter Select Mapping (Part 2 of 2)

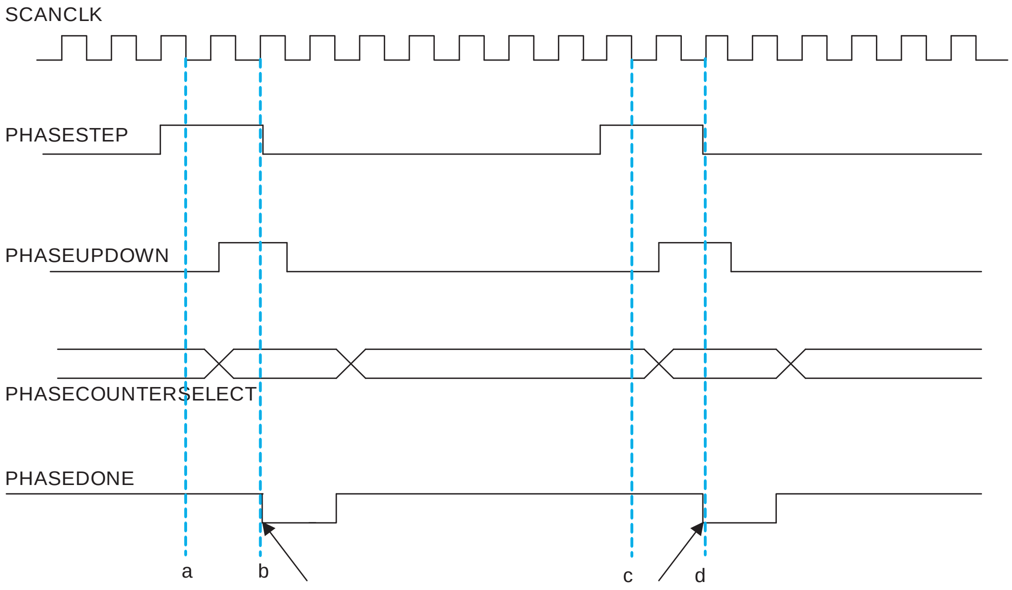
PHASECOUNTERSELECT[3]	[2]	[1]	[0]	Selects
1	0	0	1	C7 Counter
1	0	1	0	C8 Counter
1	0	1	1	C9 Counter

The procedure to perform one dynamic phase-shift step is as follows:

1. Set `phaseupdown` and `phasecounterselect` as required.
2. Assert `phasestep`. Each `phasestep` pulse enables one phase shift. The `phasestep` pulses must be at least one `scanclk` cycle apart.
3. Wait for `phasedone` to go low.
4. Deassert `phasestep.`
5. Wait for `phasedone` to go high.
6. Repeat steps 1-5 as many times as required to perform multiple phase-shifts.

All signals are synchronous to `scanclk` and must meet tsu/th requirements with respect to `scanclk` edges. They are latched on `scanclk` edges and must meet tsu/th requirements with respect to `scanclk` edges.

Figure 5–41. Dynamic Phase Shifting Waveform



PHASEDONE goes low synchronous with SCANCLK

☞ Dynamic phase-shifting can be repeated indefinitely. For example, in a design where the VCO frequency is set to 1000 MHz and the output clock frequency is 100 Mhz, performing 40 dynamic phase shifts (each one yields 125 ps phase shift) results in shifting the output clock by 180 degrees i.e phase shift of 5 ns.

The phasestep signal is latched on the negative edge of scanclk. In Figure 5–41, this is shown by the second scanclk falling edge. phasestep must stay high for at least two scanclk cycles. On the second scanclk rising edge after phasestep is latched (the fourth scanclk rising edge in Figure 5–41), the values of phaseupdown and phasecounterselect are latched and the PLL starts dynamic phase-shifting for the specified counter(s) and in the indicated direction. On the fourth scanclk rising edge, phasedone goes from high to low and remains low until the PLL finishes dynamic phase-shifting. You can perform another dynamic phase shift after the phasedone signal goes from low to high.

Depending on the VCO and scanclk frequencies, phasedone low time may be greater-than-or-less than one scanclk cycle. The maximum time for reconfiguring phase shift dynamically is to be determined based on device characterization.

After phasedone goes from low to high, you can perform another dynamic phase shift.

For details on the ALTPLL_RECONFIG Megawizard Plug-In Manager, refer to the *ALTPLL_RECONFIG Megafunction User Guide*.

PLL Specifications

Refer to the *DC and Switching Characteristics of Stratix IV Devices* chapter in volume 2 of the *Stratix IV Device Handbook* for information on PLL timing specifications.

Conclusion

Stratix IV PLLs provide you with complete control of device clocks and system timing. The ability to reconfigure the PLL counter clock frequency and phase shift in real time can be especially useful in prototyping environments, allowing you to sweep PLL output frequencies and adjust the output-clock phase-shift dynamically. These PLLs are also capable of offering flexible system-level clock management that was previously only available in discrete PLL devices. Stratix IV PLLs meet and exceed the features offered by these high-end discrete devices, reducing the need for other timing devices in the system.

Referenced Documents

This chapter references the following documents:

- *ALTPLL Megafunction User Guide*
- *ALTPLL_RECONFIG Megafunction User Guide*
- *DC and Switching Characteristics of Stratix IV Devices*
- *I/O Features in Stratix IV Devices*

Document Revision History

Table 5–18 shows the revision history for this document.

Table 5–18. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	■ Updated Table 5–7 ■ Updated Note 1 of Figure 5–10 ■ Updated Table 5–15 ■ Updated Figure 5–20 ■ Added Figure 5–21 ■ Made minor editorial changes	—
May 2008 v1.0	Initial Release.	—

Section II. I/O Interfaces

This section provides information on Stratix® IV device I/O features, external memory interfaces, and high-speed differential interfaces with DPA. This section includes the following chapters:

- Chapter 6, I/O Features in Stratix IV Devices
- Chapter 7, External Memory Interfaces in Stratix IV Devices
- Chapter 8, High-Speed Differential I/O Interfaces and DPA in Stratix IV Devices

Revision History

Refer to each chapter for its own specific revision history. For information on when each chapter was updated, refer to the Chapter Revision Dates section, which appears in the full handbook.

6. I/O Features in Stratix IV Devices

SIV51006-2.0

Introduction

Altera® Stratix IV FPGAs deliver a breakthrough level of system bandwidth and power efficiency for high-end applications, allowing you to innovate without compromise. Stratix IV I/Os are specifically designed for ease of use and rapid system integration while simultaneously providing the high bandwidth required to maximize internal logic capabilities and produce system-level performance. The Stratix IV I/O capability far exceeds the I/O bandwidth available from previous generation FPGAs. Independent modular I/O banks with a common bank structure for vertical migration lend efficiency and flexibility to the high-speed I/O. Package and die enhancements with dynamic termination and output control provide best-in-class signal integrity. Numerous I/O features assist high-speed data transfer into and out of the device, including:

- Up to 32 full-duplex clock data recovery (CDR) -based transceivers supporting data rates between 600 Mbps and 8.5 Gbps
- Dedicated circuitry to support physical layer functionality for popular serial protocols, such as PCI Express Gen1 and Gen2, Gigabit Ethernet, Serial RapidIO, SONET/SDH, XAUI/HiGig, (OIF) CEI-6G, SD/HD/3G-SDI, Fibre Channel, SFI-5, and Interlaken
- Complete PCI Express (PIPE) protocol solution with embedded PCI Express hard IP blocks that implement PHY-MAC layer, data-link layer, and transaction layer functionality
- Single-ended, non-voltage-referenced and voltage-referenced I/O standards
- Low-voltage differential signaling (LVDS), reduced swing differential signaling (RSDS), mini-LVDS, high-speed transceiver logic (HSTL), and SSTL
- Single data rate (SDR) and half data rate (HDR—half frequency and twice data width of SDR) input and output options
- Up to 132 full duplex 1.6 Gbps true LVDS channels (132 Tx + 132 Rx) on the row I/O banks
- Hard dynamic phase alignment (DPA) block with serializer/deserializer (SERDES)
- Deskew, read and write leveling, and clock-domain crossing functionality
- Programmable output current strength
- Programmable slew rate
- Programmable delay
- Programmable bus-hold circuit
- Programmable pull-up resistor
- Open-drain output
- Serial, parallel, and dynamic on-chip termination (OCT)

- Differential OCT
- Programmable pre-emphasis
- Programmable equalization
- Programmable differential output voltage (V_{OD})

This chapter contains the following sections:

- "I/O Standards Support"
- "I/O Banks" on page 6–5
- "I/O Structure" on page 6–17
- "On-Chip Termination Support and I/O Termination Schemes" on page 6–24
- "OCT Calibration" on page 6–29
- "Termination Schemes for I/O Standards" on page 6–35
- "Design Considerations" on page 6–42

I/O Standards Support

Stratix IV devices support a wide range of industry I/O standards. Table 6–1 shows the I/O standards Stratix IV devices support as well as the typical applications. These devices support V_{CCIO} voltage levels of 3.0, 2.5, 1.8, 1.5, and 1.2 V.

Table 6–1. Stratix IV I/O Standards and Applications (Part 1 of 2)

I/O Standard	Application
3.3-V LVTTL/LVCMOS *(1)*	General purpose
2.5-V LVTTL/LVCMOS	General purpose
1.8-V LVTTL/LVCMOS	General purpose
1.5-V LVTTL/LVCMOS	General purpose
1.2-V LVTTL/LVCMOS	General purpose
3.0-V PCI	PC and embedded system
3.0-V PCI-X	PC and embedded system
SSTL-2 Class I	DDR SDRAM
SSTL-2 Class II	DDR SDRAM
SSTL-18 Class I	DDR2 SDRAM
SSTL-18 Class II	DDR2 SDRAM
SSTL-15 Class I	DDR3 SDRAM
SSTL-15 Class II	DDR3 SDRAM
HSTL-18 Class I	QDRII/RLDRAM II
HSTL-18 Class II	QDRII/RLDRAM II
HSTL-15 Class I	QDRII/QDRII+/RLDRAM II
HSTL-15 Class II	QDRII/QDRII+/RLDRAM II
HSTL-12 Class I	General purpose
HSTL-12 Class II	General purpose

Table 6–1. Stratix IV I/O Standards and Applications (Part 2 of 2)

I/O Standard	Application
Differential SSTL-2 Class I	DDR SDRAM
Differential SSTL-2 Class II	DDR SDRAM
Differential SSTL-18 Class I	DDR2 SDRAM
Differential SSTL-18 Class II	DDR2 SDRAM
Differential SSTL-15 Class I	DDR3 SDRAM
Differential SSTL-15 Class II	DDR3 SDRAM
Differential HSTL-18 Class I	Clock interfaces
Differential HSTL-18 Class II	Clock interfaces
Differential HSTL-15 Class I	Clock interfaces
Differential HSTL-15 Class II	Clock interfaces
Differential HSTL-12 Class I	Clock interfaces
Differential HSTL-12 Class II	Clock interfaces
LVDS	High-speed communications
RSDS	Flat panel display
mini-LVDS	Flat panel display
LVPECL	Video graphics and clock distribution

Note to Table 6–1:

(1) The 3.3 V LVTTL/LVCMOS standard is supported using V_{CCIO} at 3.0 V.

I/O Standards and Voltage Levels

Stratix IV devices support a wide range of industry I/O standards, including single-ended, voltage-referenced single-ended, and differential I/O standards.

Table 6–2 shows the supported I/O standards and the typical values for input and output V_{CCIO}, V_{CCPD}, V_{REF}, and board V_{TT}.

Table 6–2. Stratix IV I/O Standards and Voltage Levels *(Note 1)*, *(2)* (Part 1 of 3)

I/O Standard	Standard Support	V_{CCIO} (V)				V_{CCPD} (V) (Pre-Driver Voltage)	V_{REF} (V) (Input Ref Voltage)	V_{TT} (V) (Board Termination Voltage)
		Input Operation		Output Operation				
		Column I/O Banks	Row I/O Banks	Column I/O Banks	Row I/O Banks			
3.3-V LVTTL	JESD8-B	3.0/2.5	3.0/2.5	3.0	3.0	3.0	N/A	N/A
3.3-V LVCMOS	JESD8-B	3.0/2.5	3.0/2.5	3.0	3.0	3.0	N/A	N/A
2.5-V LVTTL/LVCMOS	JESD8-5	3.0/2.5	3.0/2.5	2.5	2.5	2.5	N/A	N/A
1.8-V LVTTL/LVCMOS	JESD8-7	1.8/1.5	1.8/1.5	1.8	1.8	2.5	N/A	N/A
1.5-V LVTTL/LVCMOS	JESD8-11	1.8/1.5	1.8/1.5	1.5	1.5	2.5	N/A	N/A
1.2-V LVTTL/LVCMOS	JESD8-12	1.2	1.2	1.2	1.2	2.5	N/A	N/A
3.0-V PCI	PCI Rev 2.1	3.0	3.0	3.0	3.0	3.0	N/A	N/A

Table 6–2. Stratix IV I/O Standards and Voltage Levels *(Note 1)*, *(2)* (Part 2 of 3)

I/O Standard	Standard Support	V_{CCIO} (V)				V_{CCPD} (V) (Pre-Driver Voltage)	V_{REF} (V) (Input Ref Voltage)	V_{TT} (V) (Board Termination Voltage)
		Input Operation		Output Operation				
		Column I/O Banks	Row I/O Banks	Column I/O Banks	Row I/O Banks			
3.0-V PCI-X	PCI-X Rev 1.0	3.0	3.0	3.0	3.0	3.0	N/A	N/A
SSTL-2 Class I	JESD8-9B	*(2)*	*(2)*	2.5	2.5	2.5	1.25	1.25
SSTL-2 Class II	JESD8-9B	*(2)*	*(2)*	2.5	2.5	2.5	1.25	1.25
SSTL-18 Class I	JESD8-15	*(2)*	*(2)*	1.8	1.8	2.5	0.90	0.90
SSTL-18 Class II	JESD8-15	*(2)*	*(2)*	1.8	1.8	2.5	0.90	0.90
SSTL-15 Class I	—	*(2)*	*(2)*	1.5	1.5	2.5	0.75	0.75
SSTL-15 Class II	—	*(2)*	*(2)*	1.5	N/A	2.5	0.75	0.75
HSTL-18 Class I	JESD8-6	*(2)*	*(2)*	1.8	1.8	2.5	0.90	0.90
HSTL-18 Class II	JESD8-6	*(2)*	*(2)*	1.8	1.8	2.5	0.90	0.90
HSTL-15 Class I	JESD8-6	*(2)*	*(2)*	1.5	1.5	2.5	0.75	0.75
HSTL-15 Class II	JESD8-6	*(2)*	*(2)*	1.5	N/A	2.5	0.75	0.75
HSTL-12 Class I	JESD8-16A	*(2)*	*(2)*	1.2	1.2	2.5	0.6	0.6
HSTL-12 Class II	JESD8-16A	*(2)*	*(2)*	1.2	N/A	2.5	0.6	0.6
Differential SSTL-2 Class I	JESD8-9B	*(2)*	*(2)*	2.5	2.5	2.5	N/A	1.25
Differential SSTL-2 Class II	JESD8-9B	*(2)*	*(2)*	2.5	2.5	2.5	N/A	1.25
Differential SSTL-18 Class I	JESD8-15	*(2)*	*(2)*	1.8	1.8	2.5	N/A	0.90
Differential SSTL-18 Class II	JESD8-15	*(2)*	*(2)*	1.8	1.8	2.5	N/A	0.90
Differential SSTL-15 Class I	—	*(2)*	*(2)*	1.5	1.5	2.5	N/A	0.75
Differential SSTL-15 Class II	—	*(2)*	*(2)*	1.5	N/A	2.5	N/A	0.75
Differential HSTL-18 Class I	JESD8-6	*(2)*	*(2)*	1.8	1.8	2.5	N/A	0.90
Differential HSTL-18 Class II	JESD8-6	*(2)*	*(2)*	1.8	1.8	2.5	N/A	0.90
Differential HSTL-15 Class I	JESD8-6	*(2)*	*(2)*	1.5	1.5	2.5	N/A	0.75
Differential HSTL-15 Class II	JESD8-6	*(2)*	*(2)*	1.5	N/A	2.5	N/A	0.75
Differential HSTL-12 Class I	JESD8-16A	*(2)*	*(2)*	1.2	1.2	2.5	N/A	0.60
Differential HSTL-12 Class II	JESD8-16A	*(2)*	*(2)*	1.2	N/A	2.5	N/A	0.60
LVDS *(3)*, *(4)*	ANSI/TIA/ EIA-644	*(2)*	*(2)*	2.5	2.5	2.5	N/A	N/A
RSDS *(5)*, *(6)*	—	*(2)*	*(2)*	2.5	2.5	2.5	N/A	N/A
mini-LVDS *(5)*, *(6)*	—	*(2)*	*(2)*	2.5	2.5	2.5	N/A	N/A

Table 6–2. Stratix IV I/O Standards and Voltage Levels *(Note 1)*, *(2)* (Part 3 of 3)

I/O Standard	Standard Support	V_{CCIO} (V)				V_{CCPD} (V) (Pre-Driver Voltage)	V_{REF} (V) (Input Ref Voltage)	V_{TT} (V) (Board Termination Voltage)
		Input Operation		Output Operation				
		Column I/O Banks	Row I/O Banks	Column I/O Banks	Row I/O Banks			
LVPECL	—	*(3)*	2.5	N/A	N/A	2.5	N/A	N/A

Notes to Table 6–2:

(1) V_{CCPD} is either 2.5 or 3.0 V. For V_{CCIO} = 3.0 V, V_{CCPD} = 3.0 V. For V_{CCIO} = 2.5 V or less, V_{CCPD} = 2.5 v.

(2) Single-ended HSTL/SSTL, differential SSTL/HSTL, and LVDS input buffers are powered by V_{CCPD}. Row I/O banks support both true differential input buffers and true differential output buffers. Column I/O banks support true differential input buffers, but not true differential output buffers. I/O pins are organized in pairs to support differential standards. Column I/O differential HSTL and SSTL inputs use LVDS differential input buffers without on-chip R_D support.

(3) Column I/O banks support LVPECL I/O standards for input clock operation. Clock inputs on column I/O are powered by $V_{CCCLKIN}$ when configured as differential clock input. They are powered by V_{CCIO} when configured as single-ended clock input. Differential clock inputs in Row I/O are powered by V_{CCPD}.

(4) Column and Row I/O banks support LVDS outputs using two single-ended output buffers, an external one-resistor (LVDS_E_1R), and a three-resistor (LVDS_E_3R) network.

(5) Row I/O banks support RSDS and mini-LVDS I/O standards using a dedicated LVDS output buffer without a resistor network.

(6) Column and Row I/O banks support RSDS and mini-LVDS I/O standards using two single-ended output buffers with one-resistor (RSDS_E_1R and mini-LVDS_E_1R) and three-resistor (RSDS_E_3R and mini-LVDS_E_3R) networks.

For detailed electrical characteristics of each I/O standard, refer to the *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook*.

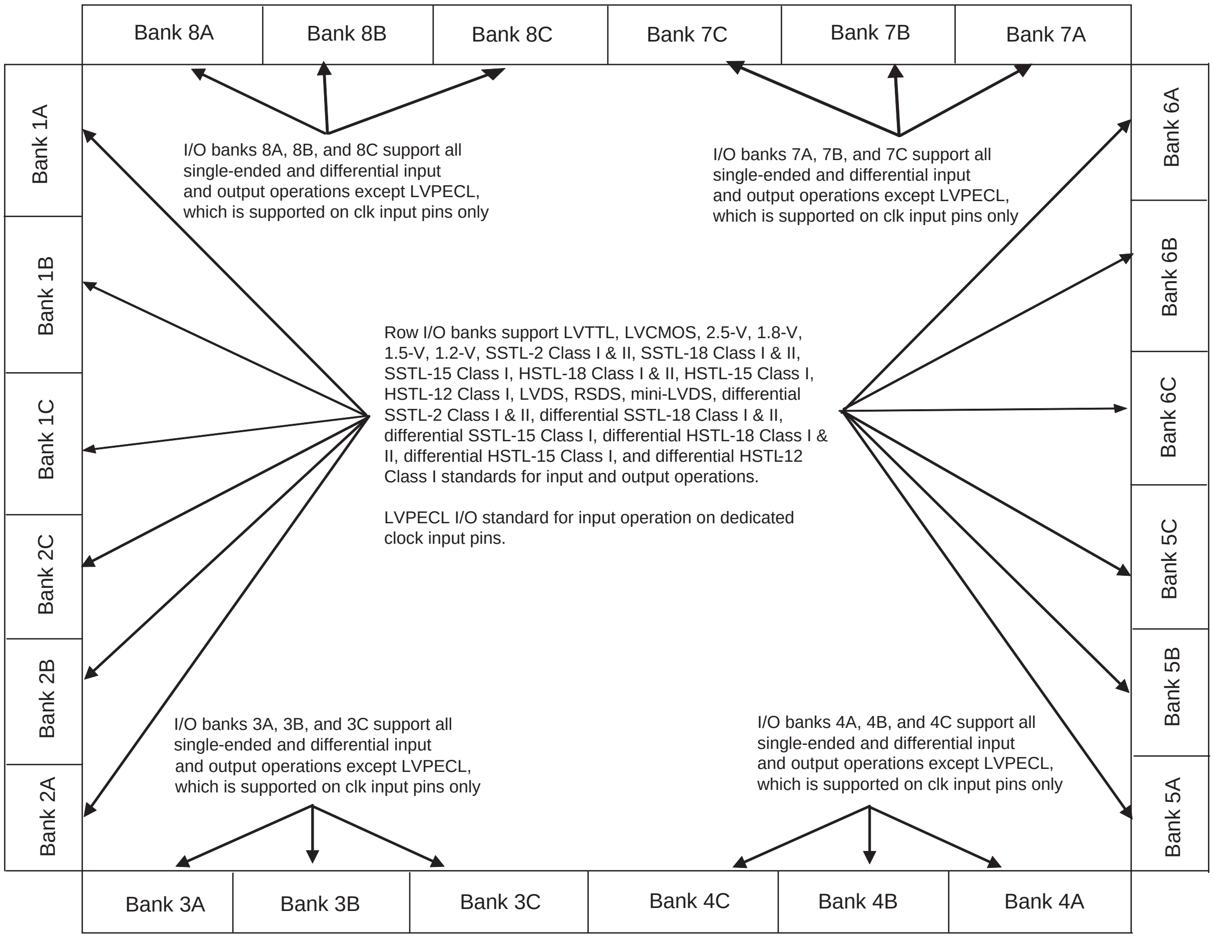
I/O Banks

Stratix IV devices contain up to 24 I/O banks, as shown in Figure 6–1. The row I/O banks contain true differential input and output buffers and dedicated circuitry to support differential standards at speeds up to 1.6 Gbps.

Each I/O bank of Stratix IV devices can support high-performance external memory interfaces with dedicated circuitry. The I/O pins are organized in pairs to support differential standards. Each I/O pin pair can support both differential input and output buffers. The only exceptions are the `clk[1,3,8,10]`, `PLL_L[1,4]_clk`, and `PLL_R[1,4]_clk` pins, which support differential input operations only.

For the number of channels available for the LVDS I/O standard, refer to the *High-Speed Differential I/O Interface with DPA* chapter in volume 1 of the *Stratix IV Device Handbook*. For more information on transceiver-bank-related features, refer to the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

Figure 6–1. Stratix IV E Devices I/O Banks *(Note 1)*, *(2)*,*(3)*,*(4)*,*(5)*,*(6)*,*(7)*, *(8)*


Bank 8A
Bank 8B
Bank 8C
Bank 7C
Bank 7B
Bank 7A
Bank 1A
Bank 1B
Bank 1C
Bank 2C
Bank 2B
Bank 2A
Bank 6A
Bank 6B
Bank 6C
Bank 5C
Bank 5B
Bank 5A
Bank 3A
Bank 3B
Bank 3C
Bank 4C
Bank 4B
Bank 4A
I/O banks 8A, 8B, and 8C support all single-ended and differential input and output operations except LVPECL, which is supported on clk input pins only
I/O banks 7A, 7B, and 7C support all single-ended and differential input and output operations except LVPECL, which is supported on clk input pins only
Row I/O banks support LVTTL, LVCMOS, 2.5-V, 1.8-V, 1.5-V, 1.2-V, SSTL-2 Class I & II, SSTL-18 Class I & II, SSTL-15 Class I, HSTL-18 Class I & II, HSTL-15 Class I, HSTL-12 Class I, LVDS, RSDS, mini-LVDS, differential SSTL-2 Class I & II, differential SSTL-18 Class I & II, differential SSTL-15 Class I, differential HSTL-18 Class I & II, differential HSTL-15 Class I, and differential HSTL-12 Class I standards for input and output operations.
LVPECL I/O standard for input operation on dedicated clock input pins.
I/O banks 3A, 3B, and 3C support all single-ended and differential input and output operations except LVPECL, which is supported on clk input pins only
I/O banks 4A, 4B, and 4C support all single-ended and differential input and output operations except LVPECL, which is supported on clk input pins only

Notes to Figure 6–1:

(1) Differential HSTL and SSTL outputs are not true differential outputs. They use two single-ended outputs with the second output programmed as inverted.

(2) Column I/O differential HSTL and SSTL inputs use LVDS differential input buffers without differential OCT support.

(3) Column I/O supports LVDS outputs using single-ended buffers and external resistor networks.

(4) Column I/O supports PCI/PCI-X with on-chip clamp diode. Row I/O supports PCI/PCI-X with external clamp diode.

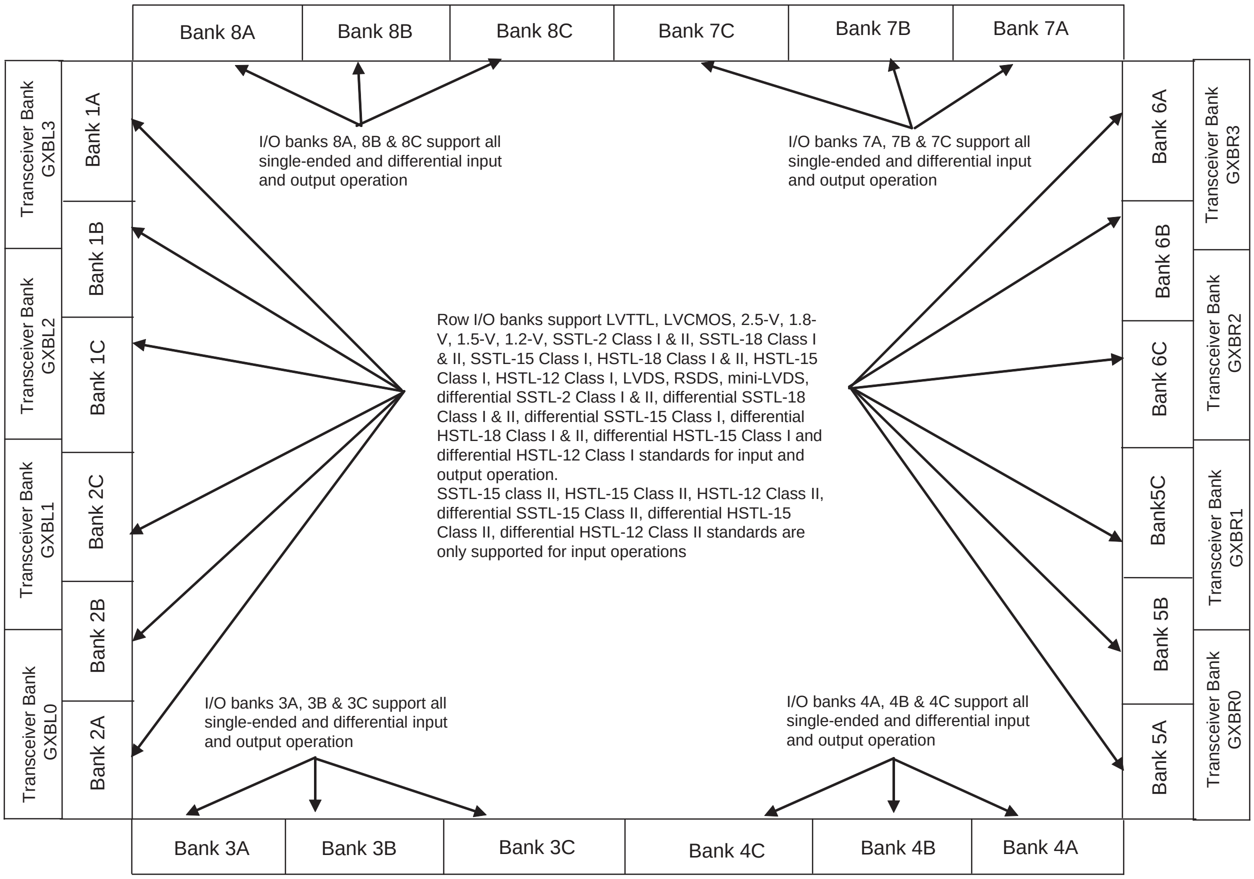
(5) Clock inputs on column I/O are powered by $V_{CCCLKIN}$ when configured as differential clock input. They are powered by V_{CCIO} when configured as single-ended clock inputs. All outputs use the corresponding bank V_{CCIO}.

(6) Row I/O supports the dedicated LVDS output buffer.

(7) Column and Row I/O banks support LVPECL standards for input clock operation.

(8) Figure 6–1 is a top view of the silicon die that corresponds to a reverse view for flip chip packages. It is a graphical representation only.

Figure 6–2. Stratix IV GX Devices I/O Banks *(Note 1)*, *(2)*, *(3)*, *(4)*, *(5)*, *(6)*, *(7)*, *(8)*


Bank 8A
Bank 8B
Bank 8C
Bank 7C
Bank 7B
Bank 7A
Transceiver Bank GXBL3
Transceiver Bank GXBL2
Transceiver Bank GXBL1
Transceiver Bank GXBL0
Bank 1A
Bank 1B
Bank 1C
Bank 2C
Bank 2B
Bank 2A
Bank 6A
Bank 6B
Bank 6C
Bank5C
Bank 5B
Bank 5A
Transceiver Bank GXBR3
Transceiver Bank GXBR2
Transceiver Bank GXBR1
Transceiver Bank GXBR0
I/O banks 8A, 8B & 8C support all single-ended and differential input and output operation
I/O banks 7A, 7B & 7C support all single-ended and differential input and output operation
Row I/O banks support LVTTL, LVCMOS, 2.5-V, 1.8-V, 1.5-V, 1.2-V, SSTL-2 Class I & II, SSTL-18 Class I & II, SSTL-15 Class I, HSTL-18 Class I & II, HSTL-15 Class I, HSTL-12 Class I, LVDS, RSDS, mini-LVDS, differential SSTL-2 Class I & II, differential SSTL-18 Class I & II, differential SSTL-15 Class I, differential HSTL-18 Class I & II, differential HSTL-15 Class I and differential HSTL-12 Class I standards for input and output operation.
SSTL-15 class II, HSTL-15 Class II, HSTL-12 Class II, differential SSTL-15 Class II, differential HSTL-15 Class II, differential HSTL-12 Class II standards are only supported for input operations
I/O banks 3A, 3B & 3C support all single-ended and differential input and output operation
I/O banks 4A, 4B & 4C support all single-ended and differential input and output operation
Bank 3A
Bank 3B
Bank 3C
Bank 4C
Bank 4B
Bank 4A

Notes to Figure 6–2:

(1) Differential HSTL and SSTL outputs are not true differential outputs. They use two single-ended outputs with the second output programmed as inverted.

(2) Column I/O differential HSTL and SSTL inputs use LVDS differential input buffers without differential OCT support.

(3) Column I/O supports LVDS outputs using single-ended buffers and external resistor networks.

(4) Column I/O supports PCI/PCI-X with on-chip clamp diode. Row I/O supports PCI/PCI-X with external clamp diode.

(5) Clock inputs on column I/O are powered by $V_{CCCLKIN}$ when configured as differential clock input. They are powered by V_{CCIO} when configured as single-ended clock inputs.All outputs use the corresponding bank V_{CCIO}.

(6) Row I/O banks support the dedicated LVDS output buffer.

(7) Column and Row I/O banks support LVPECL standards for input clock operation.

(8) Figure 6–2 is a top view of the silicon die that corresponds to a reverse view for flip chip packages. It is a graphical representation only.

Modular I/O Banks

The I/O pins in Stratix IV devices are arranged in groups called modular I/O banks. Depending on device densities, the number of Stratix IV device I/O banks ranges from 16 to 24, and the number of I/O pins on each bank is 24, 32, 36, 40, or 48. Figure 6–4 through Figure 6–15 show the number of I/O pins available in each I/O bank.

In Stratix IV devices, the maximum number of I/O banks per side is either four or six, depending on the device density. When migrating between devices with a different number of I/O banks per side, it is the middle or "B" bank which is removed or inserted. For example, when moving from a 24-bank device to a 16-bank device, the banks that are dropped are "B" banks, namely: 1B, 2B, 3B, 4B, 5B, 6B, 7B, and 8B. Similarly, when moving from a 16-bank device to a 24-bank device, the banks that are added are the same "B" banks.

Upon migration from a smaller device to a larger device, the bank size increases or remains the same, but never decreases. For example, the number of I/O pins to a bank may increase from 24 to 26, 32, 36, 40, 42, or 48, but will never decrease. This is shown in Figure 6–3.

Figure 6–3. Bank Migration Path with Increasing Device Size

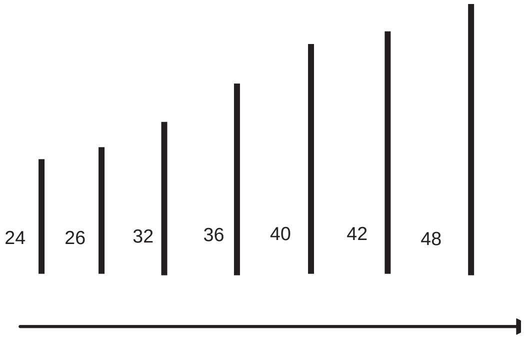
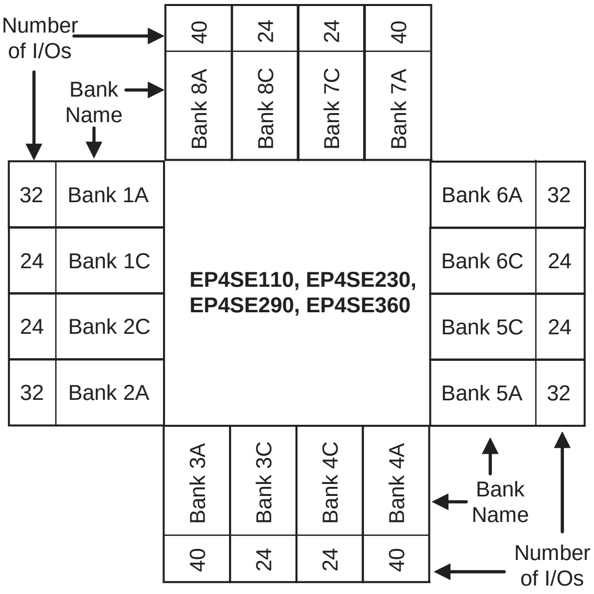


Figure 6–4 through Figure 6–15 show the number of I/O pins and packaging information for different sets of available devices.

Figure 6–4. Number of I/Os in Each Bank in EP4SE110, EP4SE230, EP4SE290, and EP4SE360 in the 780-Pin FineLine BGA Package *(Note 1)*, *(2)*

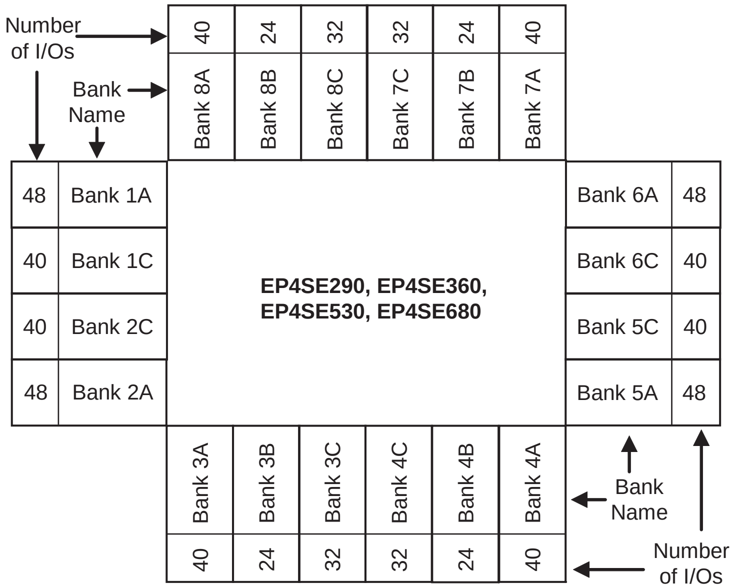


Notes to Figure 6–4:

(1) I/O counts do not include dedicated clock inputs that can be used as data inputs.

(2) Figure 6–4 is a top view of the silicon die that corresponds to a reverse view for flip chip packages. It is a graphical representation only.

Figure 6–5. Number of I/Os in Each Bank in EP4SE290, EP4SE360, EP4SE530, and EP4SE680 in the 1152-Pin FineLine BGA Package *(Note 1)*, *(2)*

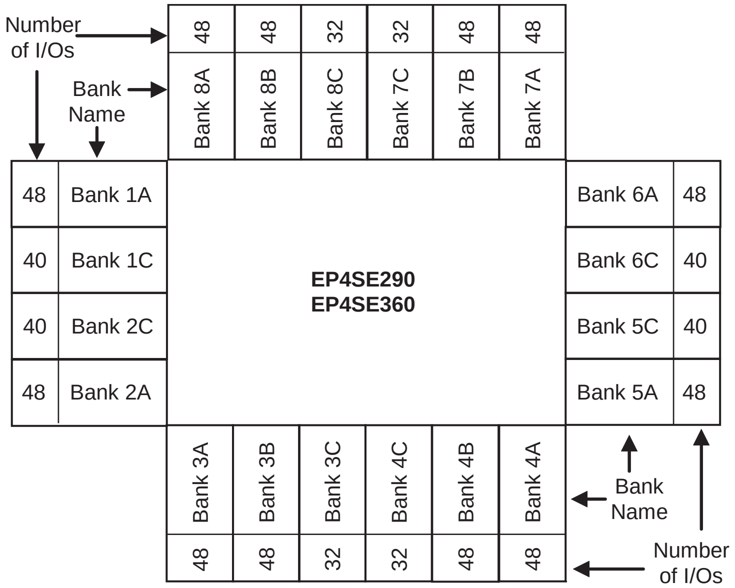


Notes to Figure 6–5:

(1) I/O counts do not include dedicated clock inputs that can be used as data inputs.

(2) Figure 6–5 is a top view of the silicon die that corresponds to a reverse view for flip chip packages. It is a graphical representation only.

Figure 6–6. Number of I/Os in Each Bank in EP4SE290 and EP4SE360 in 1517-Pin FineLine BGA Package *(Note 1)*, *(2)*

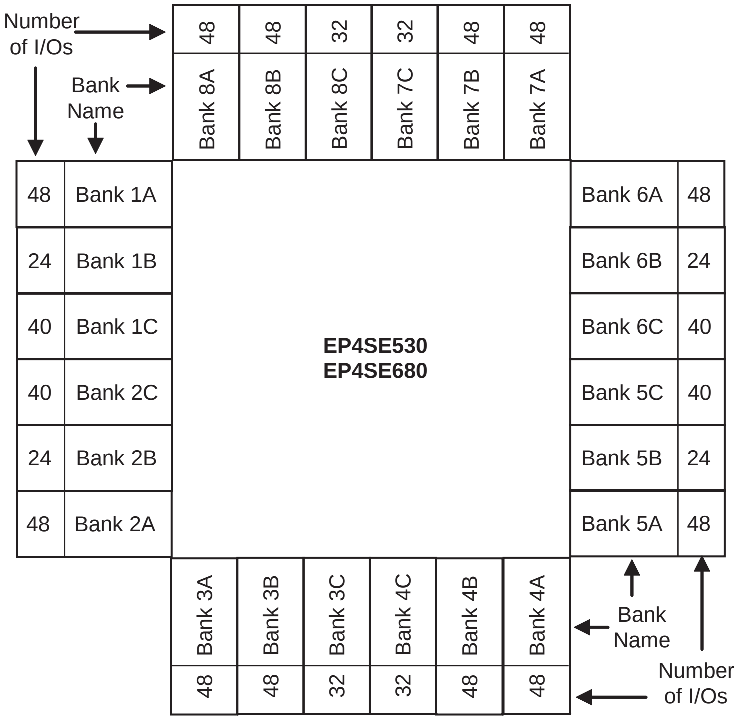


Notes to Figure 6–6:

(1) I/O counts do not include dedicated clock inputs that can be used as data inputs.

(2) Figure 6–6 is a top view of the silicon die that corresponds to a reverse view for flip chip packages. It is a graphical representation only.

Figure 6–7. Number of I/Os in Each Bank in EP4SE530 and EP4SE680 in 1517-Pin and EP4SE530 in 1932-Pin FineLine BGA Package *(Note 1)*, *(2)*

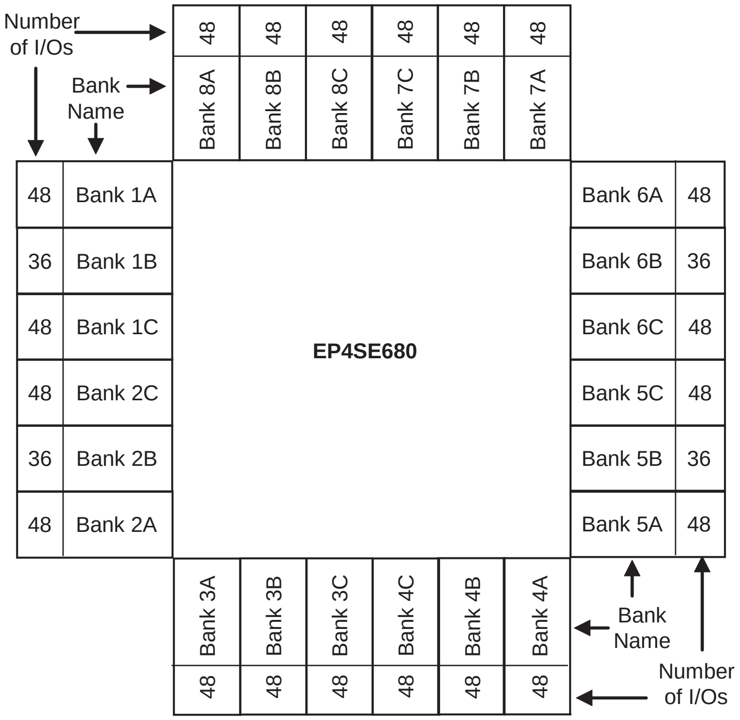


Notes to Figure 6–7:

(1) I/O counts do not include dedicated clock inputs that can be used as data inputs.

(2) Figure 6–7 is a top view of the silicon die that corresponds to a reverse view for flip chip packages. It is a graphical representation only.

Figure 6–8. Number of I/Os in Each Bank in EP4SE680 in 1932-Pin FineLine BGA Package *(Note 1)*, *(2)*

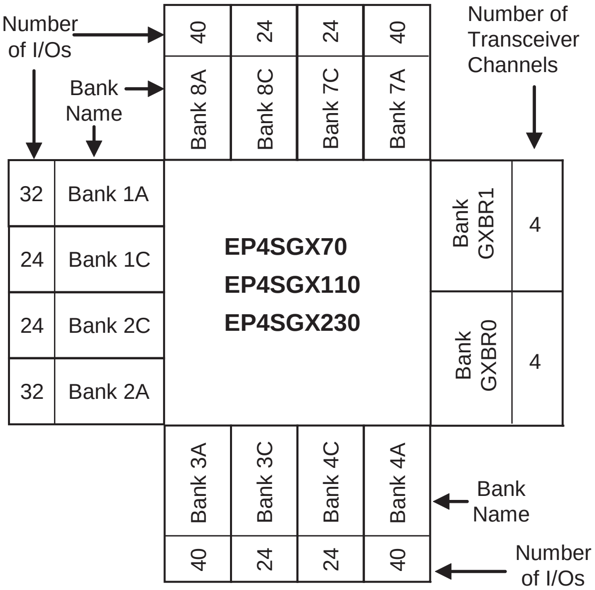


Notes to Figure 6–8:

(1) I/O counts do not include dedicated clock inputs that can be used as data inputs.

(2) Figure 6–8 is a top view of the silicon die that corresponds to a reverse view for flip chip packages. It is a graphical representation only.

Figure 6–9. Number of I/Os in Each Bank in EP4SGX70, EP4SGX110, and EP4SGX230 in 780-Pin FineLine BGA Package *(Note 1)*, *(2)*

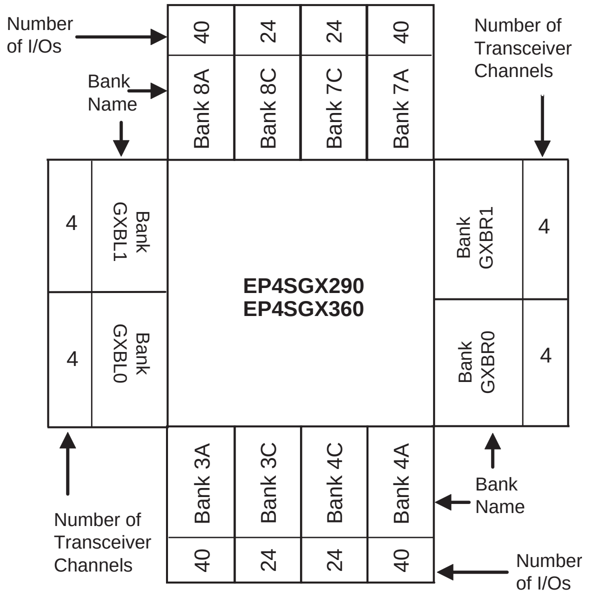


Notes to Figure 6–9:

(1) I/O counts do not include dedicated clock inputs that can be used as data inputs.

(2) Figure 6–9 is a top view of the silicon die that corresponds to a reverse view for flip chip packages. It is a graphical representation only.

Figure 6–10. Number of I/Os in Each Bank in EP4SGX290 and EP4SGX360 in 780-Pin FineLine BGA Package *(Note 1)*, *(2)*

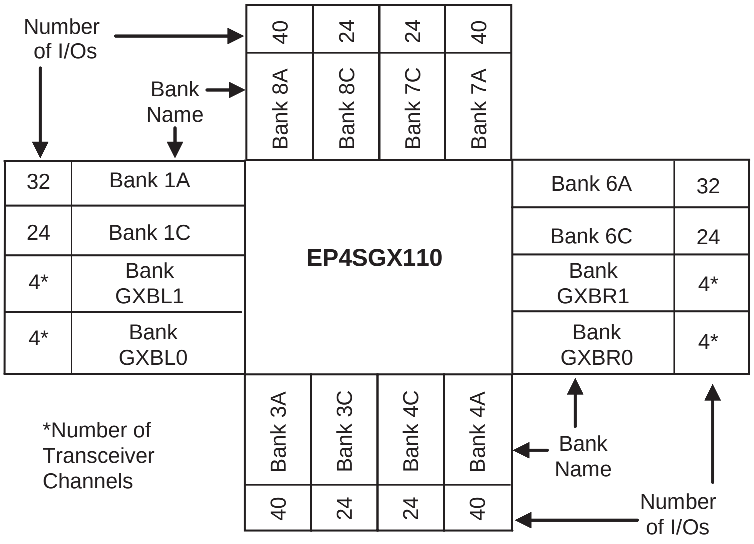


Notes to Figure 6–10:

(1) I/O counts do not include dedicated clock inputs that can be used as data inputs.

(2) Figure 6–10 is a top view of the silicon die that corresponds to a reverse view for flip chip packages. It is a graphical representation only.

Figure 6–11. Number of I/Os in Each Bank in EP4SGX110 in 1152-Pin FineLine BGA Package *(Note 1)*, *(2)*

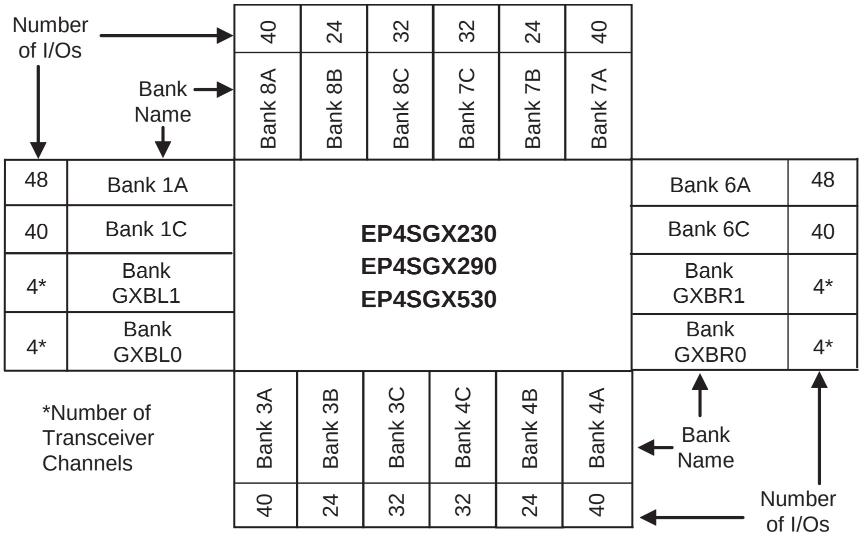


Notes to Figure 6–11:

(1) I/O counts do not include dedicated clock inputs that can be used as data inputs.

(2) Figure 6–11 is a top view of the silicon die that corresponds to a reverse view for flip chip packages. It is a graphical representation only.

Figure 6–12. Number of I/Os in Each Bank in EP4SGX230, EP4SGX290, EP4SGX360, and EP4SGX530 in 1152-Pin FineLine BGA Package *(Note 1)*, *(2)*

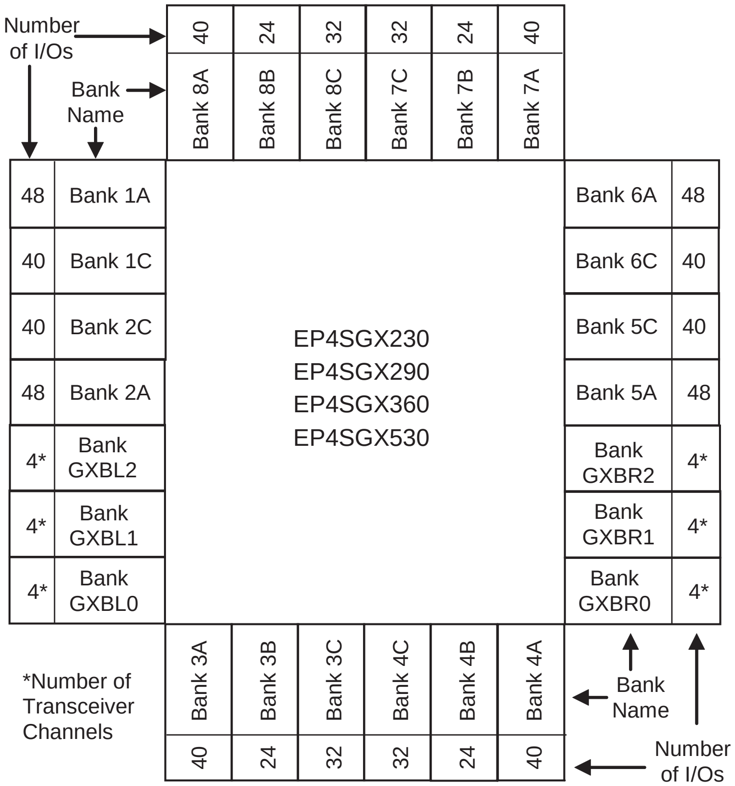


Notes to Figure 6–12:

(1) I/O counts do not include dedicated clock inputs that can be used as data inputs.

(2) Figure 6–12 is a top view of the silicon die that corresponds to a reverse view for flip chip packages. It is a graphical representation only.

Figure 6–13. Number of I/Os in Each Bank in EP4SGX230, EP4SGX290, EP4SGX360, and EP4SGX530 in 1517-Pin FineLine BGA Package *(Note 1)*, *(2)*

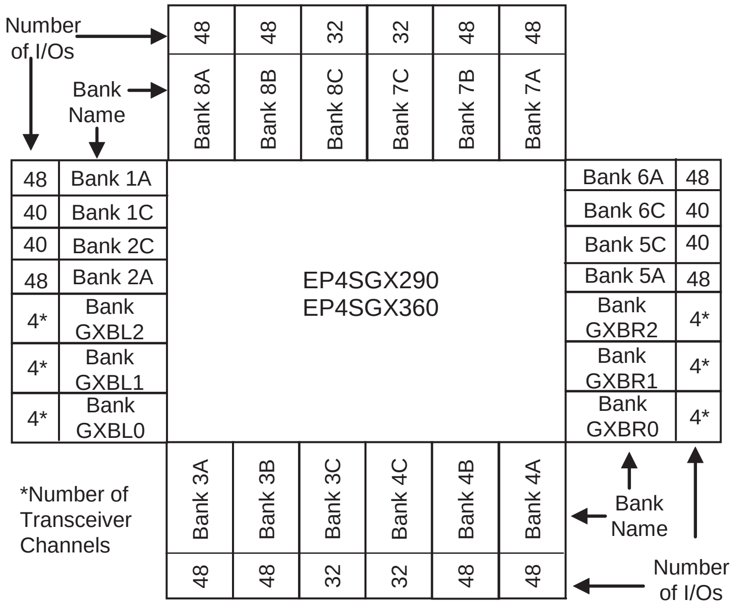


Notes to Figure 6–13:

(1) I/O counts do not include dedicated clock inputs that can be used as data inputs.

(2) Figure 6–13 is a top view of the silicon die that corresponds to a reverse view for flip chip packages. It is a graphical representation only.

Figure 6–14. Number of I/Os in Each Bank in EP4SGX290 and EP4SGX360 in 1932-Pin FineLine BGA Package *(Note 1)*, *(2)*

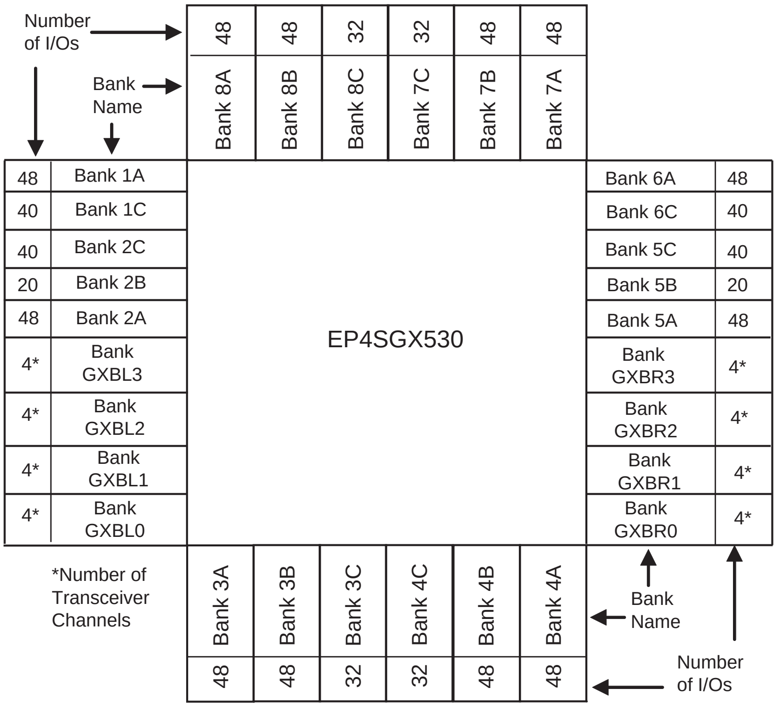


Notes to Figure 6–14:

(1) I/O counts do not include dedicated clock inputs that can be used as data inputs.

(2) Figure 6–14 is a top view of the silicon die that corresponds to a reverse view for flip chip packages. It is a graphical representation only.

Figure 6–15. Number of I/Os in Each Bank in EP4SGX530 in 1932-Pin FineLine BGA Package *(Note 1)*, *(2)*



Notes to Figure 6–15:

(1) I/O counts do not include dedicated clock inputs that can be used as data inputs.

(2) Figure 6–15 is a top view of the silicon die that corresponds to a reverse view for flip chip packages. It is a graphical representation only.

I/O Structure

The I/O element (IOE) in Stratix IV devices contains a bidirectional I/O buffer and I/O registers to support a complete embedded bidirectional single data rate or DDR transfer. The IOEs are located in I/O blocks around the periphery of the Stratix IV device. There are up to four IOEs per row I/O block and four IOEs per column I/O block. The row IOEs drive row, column, or direct link interconnects. The column IOEs drive column interconnects.

The Stratix IV bidirectional IOE also supports the following features:

- Programmable input delay
- Programmable output-current strength
- Programmable slew rate
- Programmable output delay
- Programmable bus-hold
- Programmable pull-up resistor
- Open-drain output
- On-chip series termination with calibration
- On-chip series termination without calibration

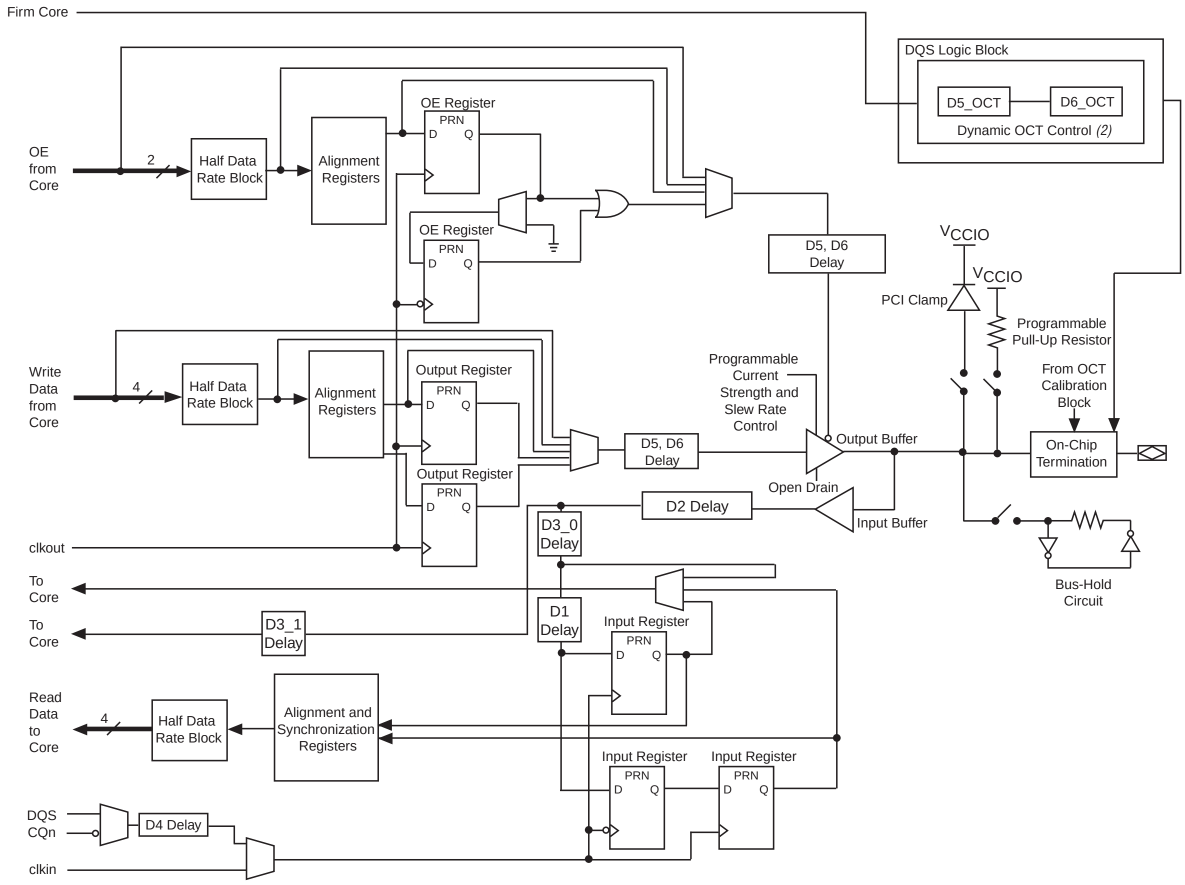
- On-chip parallel termination with calibration
- On-chip differential termination
- PCI clamping diode

The I/O registers are composed of the input path for handling data from the pin to the core, the output path for handling data from the core to the pin, and the output-enable (OE) path for handling the OE signal to the output buffer. These registers allow faster source-synchronous register-to-register transfers and resynchronization. The input path consists of the DDR input registers, alignment and synchronization registers, and HDR. You can bypass each block of the input path.

The output and OE paths are divided into output or OE registers, alignment registers, and HDR blocks. You can bypass each block of the output and OE paths.

Figure 6–16 shows the Stratix IV IOE structure.

Figure 6–16. Stratix IV IOE Structure *(Note 1)*, *(2)*



Notes to Figure 6–16:

(1) D3_0 and D3_1 delays have the same available settings in the Quartus II software.

(2) One dynamic OCT control is available per DQ/DQS group.

For more information about I/O registers and how they are used for memory applications, refer to the *External Memory Interfaces* chapter in volume 1 of the *Stratix IV Device Handbook*.

3.3-V I/O Interface

Stratix IV I/O buffers support 3.3-V I/O standards. You can use them as transmitters or receivers in your system. The output high voltage (VOH), output low voltage (VOL), input high voltage (VIH), and input low voltage (VIL) levels meet the 3.3-V I/O standards specifications defined by EIA/JEDEC Standard JESD8-B with margin when the Stratix IV V_{CCIO} voltage is powered by 3.0 V.

To ensure device reliability and proper operation, when interfacing with a 3.3 V I/O system using Stratix IV devices, it is important to make sure that the absolute maximum ratings of the devices are not violated. Altera recommends performing IBIS simulation to determine that the overshoot and undershoot voltages are within the guidelines.

When using the Stratix IV device as a transmitter, you can use some techniques to limit overshoot and undershoot at the I/O pins, such as using slow slew rate and series termination, but they are not required. Transmission line effects that cause large voltage deviations at the receiver are associated with an impedance mismatch between the driver and transmission lines. By matching the impedance of the driver to the characteristic impedance of the transmission line, you can significantly reduce overshoot voltage. You can use a series termination resistor placed physically close to the driver to match total driver impedance to transmission line impedance. Stratix IV devices support series on-chip termination (OCT) for all LVTTL and LVCMOS I/O standards in all I/O banks.

When using the Stratix IV device as a receiver, a technique you can use to limit overshoot, though not required, is using a clamping diode (on-chip or off-chip). Stratix IV devices provide an optional on-chip PCI-clamping diode for column I/O pins. You can use this diode to protect I/O pins against overshoot voltage.

The 3.3 V I/O standard is supported using bank supply voltage (V_{CCIO}) at 3.0 V. In this method, the clamping diode (on-chip or off-chip), when enabled, can sufficiently clamp overshoot voltage to within the DC and AC input voltage specifications. The clamped voltage can be expressed as the sum of the supply voltage (V_{CCIO}) and the diode forward voltage.

For more details about absolute maximum rating and maximum allowed overshoot during transitions, refer to the *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook*.

External Memory Interfaces

In addition to the I/O registers in each IOE, Stratix IV devices also have dedicated registers and phase-shift circuitry on all I/O banks for interfacing with external memory interfaces.

For more information about external memory interfaces, refer to the *External Memory Interfaces* chapter in volume 1 of the *Stratix IV Device Handbook.*

High-Speed Differential I/O with DPA Support

Stratix IV devices have the following dedicated circuitry for high-speed differential I/O support:

- Differential I/O buffer

- Transmitter serializer
- Receiver deserializer
- Data realignment
- Dynamic phase aligner (DPA)
- Synchronizer (FIFO buffer)
- Phase-locked loops (PLLs)

For more information about DPA support, refer to the *High-Speed Differential I/O Interfaces with DPA* chapter in volume 1 of the *Stratix IV Device Handbook*.

Current Strength

The output buffer for each Stratix IV device I/O pin has a programmable current strength control for certain I/O standards. You can use programmable current strength to mitigate the effects of high signal attenuation due to a long transmission line or a legacy backplane. The LVTTL, LVCMOS, SSTL, and HSTL standards have several levels of current strength that you can control. Information about programmable current strength is shown in Table 6–3.

Table 6–3. Programmable Current Strength (Part 1 of 2) *(Note 1)*, *(2)*

I/O Standard	I_{OH} / I_{OL} Current Strength Setting (mA) for Column I/O Pins	I_{OH} / I_{OL} Current Strength Setting (mA) for Row I/O Pins
3.3-V LVTTL	16, 12, 8, 4	12, 8, 4
3.3-V LVCMOS	16, 12, 8, 4	8, 4
2.5-V LVTTL/LVCMOS	16, 12, 8, 4	12, 8, 4
1.8-V LVTTL/LVCMOS	12, 10, 8, 6, 4, 2	8, 6, 4, 2
1.5-V LVTTL/LVCMOS	12, 10, 8, 6, 4, 2	8, 6, 4, 2
1.2-V LVTTL/LVCMOS	8, 6, 4, 2	4, 2
SSTL-2 Class I	12, 10, 8	12, 8
SSTL-2 Class II	16	16
SSTL-18 Class I	12, 10, 8, 6, 4	12, 10, 8, 6, 4
SSTL-18 Class II	16, 8	16, 8
SSTL-15 Class I	12, 10, 8, 6, 4	8, 6, 4
SSTL-15 Class II	16, 8	—
HSTL-18 Class I	12, 10, 8, 6, 4	12, 10, 8, 6, 4
HSTL-18 Class II	16	16
HSTL-15 Class I	12, 10, 8, 6, 4	8, 6, 4
HSTL-15 Class II	16	—
HSTL-12 Class I	12, 10, 8, 6, 4	8, 6, 4

Table 6–3. Programmable Current Strength (Part 2 of 2) *(Note 1)*, *(2)*

I/O Standard	I_{OH} / I_{OL} Current Strength Setting (mA) for Column I/O Pins	I_{OH} / I_{OL} Current Strength Setting (mA) for Row I/O Pins
HSTL-12 Class II	**16**	—

Note to Table 6–3:

(1) The default setting in the Quartus II software is 50-Ω OCT Rs without calibration for all non-voltage reference and HSTL and SSTL class I I/O standards. The default setting is 25-Ω OCT Rs without calibration for HSTL and SSTL class II I/O standards.

(2) The 3.3-V LVTTL and 3.3-V LVCMOS are supported using V_{CCIO} and V_{CCPD} at 3.0 V.

☞ Altera recommends performing IBIS or SPICE simulations to determine the best current strength setting for your specific application.

Slew Rate Control

The output buffer for each Stratix IV device regular- and dual-function I/O pin has a programmable output slew-rate control that you can configure for low-noise or high-speed performance. A faster slew rate provides high-speed transitions for high-performance systems. A slower slew rate can help reduce system noise, but adds a nominal delay to the rising and falling edges. Each I/O pin has an individual slew-rate control, allowing you to specify the slew rate on a pin-by-pin basis.

☞ You cannot use the programmable slew rate feature when using OCT R_S.

The Quartus II software allows four settings for programmable slew rate control—0, 1, 2, and 3—where 0 is slow slew rate and 3 is fast slew rate. Figure 6–4 lists the default slew rate settings from the Quartus II software.

Table 6–4. Default Slew Rate Settings

I/O Standard	Slew Rate Option	Default Slew Rate
1.2-V, 1.5-V, 1.8-V, 2.5-V, 3.3-V LVTTL/LVCMOS	0, 1, 2, 3	1
SSTL-2, SSTL-18, SSTL-15, HSTL-18, HSTL-15, HSTL-12	0, 1, 2, 3	3
3.0-V PCI/PCI-X	0, 1, 2, 3	3
LVDS_E_1R, mini-LVDS_E_1R, RSDS_E_1R	0, 1, 2, 3	3
LVDS_E_3R, mini-LVDS_E_3R, RSDS_E_3R	2	2

You can use faster slew rates to improve the available timing margin in memory-interface applications or when the output pin has a high-capacitive loading. Altera recommends performing IBIS or SPICE simulations to determine the best slew rate setting for your specific application.

I/O Delay

The following sections discuss programmable IOE delay and programmable output buffer delay.

Programmable IOE Delay

The Stratix IV device IOE includes programmable delays shown in Figure 6–16 that you can activate to ensure zero hold times, minimize setup times, or increase clock-to-output times. Each pin can have a different input delay from pin to input register or a delay from output register to output pin values to ensure that the bus has the same delay going into or out of the device. This feature helps read and time margins because it minimizes the uncertainties between signals in the bus.

For programmable IOE delay specifications, refer to the *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook*.

Programmable Output Buffer Delay

Stratix IV devices support delay chains built inside the single-ended output buffer, as shown in Figure 6–16. The delay chains can independently control the rising and falling edge delays of the output buffer, providing the ability to adjust the output-buffer duty cycle, compensate channel-to-channel skew, reduce simultaneous switching output (SSO) noise by deliberately introducing channel-to-channel skew, and improve high-speed memory-interface timing margins. Stratix IV devices support four levels of output buffer delay settings. The default setting is **No Delay**.

For the programmable output buffer delay specifications, refer to the *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook*.

Open-Drain Output

Stratix IV devices provide an optional open-drain output (equivalent to an open collector output) for each I/O pin. When configured as open drain, the logic value of the output is either high-Z or 0. Typically, an external pull-up resistor is needed to provide logic high.

Bus Hold

Each Stratix IV device I/O pin provides an optional bus-hold feature. The bus-hold circuitry can weakly hold the signal on an I/O pin at its last-driven state. Because the bus-hold feature holds the last-driven state of the pin until the next input signal is present, you do not need an external pull-up or pull-down resistor to hold a signal level when the bus is tri-stated.

The bus-hold circuitry also pulls non-driven pins away from the input threshold voltage where noise can cause unintended high-frequency switching. You can select this feature individually for each I/O pin. The bus-hold output drives no higher than V_{CCIO} to prevent over-driving signals. If you enable the bus-hold feature, you cannot use the programmable pull-up option. Disable the bus-hold feature if the I/O pin is configured for differential signals.

The bus-hold circuitry uses a resistor with a nominal resistance (R_{BH}) of approximately 7k Ω to weakly pull the signal level to the last-driven state.

For the specific sustaining current driven through this resistor and the overdrive current used to identify the next-driven input level, refer to the *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook*. This information is provided for each V_{CCIO} voltage level.

The bus-hold circuitry is active only after configuration. When going into user mode, the bus-hold circuit captures the value on the pin present at the end of configuration.

Pull-Up Resistor

Each Stratix IV device I/O pin provides an optional programmable pull-up resistor during user mode. If you enable this feature for an I/O pin, the pull-up resistor (typically 25K Ω) weakly holds the I/O to the V_{CCIO} level.

Programmable pull-up resistors are only supported on user I/O pins and are not supported on dedicated configuration pins, JTAG pins, or dedicated clock pins. If you enable the programmable pull-up option, you cannot use the bus-hold feature.

Pre-Emphasis

Stratix IV LVDS transmitters support programmable pre-emphasis to compensate for the frequency dependent attenuation of the transmission line. The Quartus II software allows four settings for programmable pre-emphasis.

For more information about programmable pre-emphasis, refer to the *High-Speed Differential I/O Interfaces* chapter in volume 1 of the *Stratix IV Device Handbook*.

Differential Output Voltage

Stratix IV LVDS transmitters support programmable VOD. The programmable VOD settings allow you to adjust output eye height to optimize trace length and power consumption. A higher VOD swing improves voltage margins at the receiver end while a smaller VOD swing reduces power consumption. The Quartus II software allows four settings for programmable VOD.

For more information about programmable VOD, refer to the *High-Speed Differential I/O Interfaces with DPA* chapter in volume 1 of the *Stratix IV Device Handbook*.

MultiVolt I/O Interface

The Stratix IV architecture supports the MultiVolt I/O interface feature that allows the Stratix IV devices in all packages to interface with systems of different supply voltages.

You can connect the VCCIO pins to a 1.2-, 1.5-, 1.8-, 2.5-, or 3.0-V power supply, depending on the output requirements. The output levels are compatible with systems of the same voltage as the power supply. (For example, when VCCIO pins are connected to a 1.5-V power supply, the output levels are compatible with 1.5-V systems).

The Stratix IV VCCPD power pins must be connected to a 2.5- or 3.0-V power supply. Using these power pins to supply the pre-driver power to the output buffers increases the performance of the output pins. Table 6–5 summarizes Stratix IV MultiVolt I/O support.

Table 6–5. Stratix IV MultiVolt I/O Support *(Note 1)*, *(2)*

VCCIO (V)	Input Signal (V)						Output Signal (V)					
	1.2	1.5	1.8	2.5	3.0	3.3	1.2	1.5	1.8	2.5	3.0	3.3
1.2	✓	—	—	—	—	—	✓	—	—	—	—	—
1.5	—	✓	✓	—	—	—	—	✓	—	—	—	—
1.8	—	✓	✓	—	—	—	—	—	✓	—	—	—
2.5	—	—	—	✓	✓	✓	—	—	—	✓	—	—
3.0	—	—	—	✓	✓	✓	—	—	—	—	✓	—

Notes to Table 6–5:

(1) The pin current may be slightly higher than the default value. You must verify that the driving device's VOL maximum and VOH minimum voltages do not violate the applicable Stratix IV VIL maximum and VIH minimum voltage specifications.

(2) Altera recommends that you use an external clamping diode on the column I/O pins when the input signal is 3.0 V or 3.3 V.

On-Chip Termination Support and I/O Termination Schemes

Stratix IV devices feature dynamic series and parallel on-chip termination (OCT) to provide I/O impedance matching and termination capabilities. OCT maintains signal quality, saves board space, and reduces external component costs.

Stratix IV devices support:

- On-chip series (R_S) with calibration
- On-chip series (R_S) without calibration
- Parallel (R_T) with calibration
- Dynamic series termination for single-ended I/O standards
- Parallel termination for single-ended I/O standards
- On-chip differential termination (R_D) for differential LVDS I/O standards

Stratix IV devices support OCT in all I/O banks by selecting one of the OCT I/O standards.

These devices also support OCT R_S and R_T in the same I/O bank for different I/O standards if they use the same V_{CCIO} supply voltage. You can independently configure each I/O in an I/O bank to support OCT R_S, programmable current strength, or OCT R_T.

You cannot configure both OCT R_S and programmable current strength for the same I/O buffer.

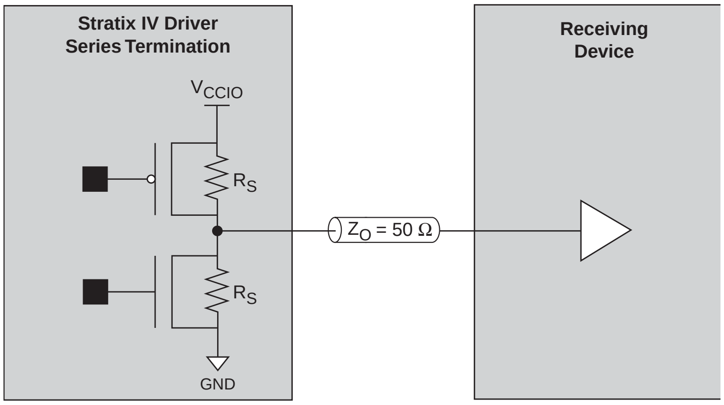
A pair of RUP and RDN pins are available in a given I/O bank, and are shared for series- and parallel-calibrated termination. The RUP and RDN pins share the same V_{CCIO} and GND, respectively, with the I/O bank where they are located. The RUP and RDN pins are dual-purpose I/Os and function as regular I/Os if you do not use the calibration circuit. When used for calibration, the RUP pin is connected to V_{CCIO} through an external 25-Ω ± 1% or 50-Ω ± 1% resistor for an on-chip series termination value of 25-Ω or 50-Ω, respectively; the RDN pin is connected to GND through an external 25-Ω ±1% or 50-Ω ±1% resistor for an on-chip series termination value of 25-Ω or 50-Ω, respectively. For on-chip parallel termination, the RUP pin is connected to V_{CCIO} through an external 50-Ω ±1% resistor; the RDN pin is connected to GND through an external 50-Ω ±1% resistor.

On-Chip Series (R_S) Termination without Calibration

Stratix IV devices support driver-impedance matching to provide the I/O driver with controlled output impedance that closely matches the impedance of the transmission line. As a result, you can significantly reduce reflections. Stratix IV devices support on-chip series termination for single-ended I/O standards (see Figure 6–17).

The R_S shown in Figure 6–17 is the intrinsic impedance of the output transistors. The typical R_S values are 25 Ω and 50 Ω. When you select matching impedance, current strength is no longer selectable.

Figure 6–17. Stratix IV On-Chip Series Termination without Calibration



To use on-chip termination for the SSTL Class I standard, you should select the 50-Ω on-chip series termination setting, thus eliminating the external 25-Ω R_S (to match the 50-Ω transmission line). For the SSTL Class II standard, you should select the 25-Ω on-chip series termination setting (to match the 50-Ω transmission line and the near-end external 50-Ω pull-up to V_{TT}).

On-Chip Series Termination with Calibration

Stratix IV devices support on-chip series termination with calibration in all banks. The on-chip series termination calibration circuit compares the total impedance of the I/O buffer to the external 25-Ω ±1% or 50-Ω ±1% resistors connected to the RUP and RDN pins, and dynamically enables or disables the transistors until they match.

The R_S shown in Figure 6–18 is the intrinsic impedance of the transistors. Calibration occurs at the end of device configuration. When the calibration circuit finds the correct impedance, it powers down and stops changing the characteristics of the drivers.

Figure 6–18. Stratix IV On-Chip Series Termination with Calibration

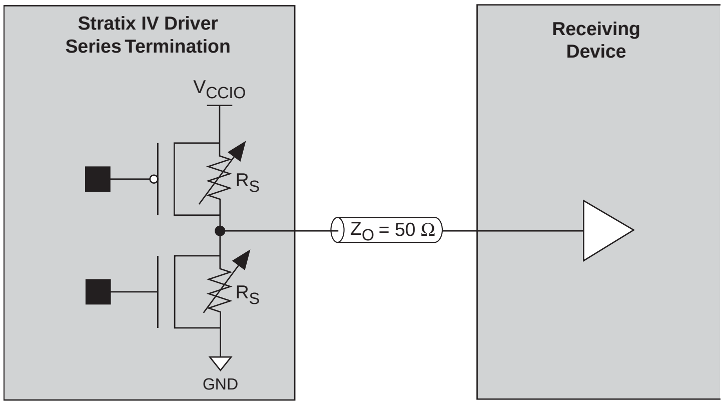


Table 6–6 shows the list of I/O standards that support on-chip series termination with calibration.

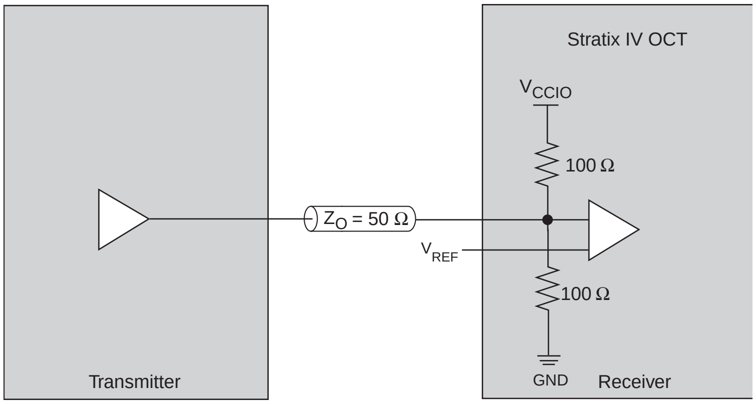
Table 6–6. Selectable I/O Standards for On-Chip Series Termination with Calibration

I/O Standard	On-Chip Series Termination Setting		
	Row I/O	Column I/O	Unit
3.3-V LVTTL/LVCMOS	50	50	Ω
	25	25	Ω
2.5-V LVTTL/LVCMOS	50	50	Ω
	25	25	Ω
1.8-V LVTTL/LVCMOS	50	50	Ω
	25	25	Ω
1.5-V LVTTL/LVCMOS	50	50	Ω
		25	Ω
1.2-V LVTTL/LVCMOS	50	50	Ω
		25	Ω
SSTL-2 Class I	50	50	Ω
SSTL-2 Class II	25	25	Ω
SSTL-18 Class I	50	50	Ω
SSTL-18 Class II	25	25	Ω
SSTL-15 Class I	50	50	Ω
SSTL-15 Class II	N/A	25	Ω
HSTL-18 Class I	50	50	Ω
HSTL-18 Class II	25	25	Ω
HSTL-15 Class I	50	50	Ω
HSTL-15 Class II	N/A	25	Ω
HSTL-12 Class I	50	50	Ω
HSTL-12 Class II	N/A	25	Ω

On-Chip Parallel Termination with Calibration

Stratix IV devices support on-chip parallel termination with calibration in all banks. On-chip parallel termination with calibration is only supported for input configuration of input and bidirectional pins. Output pin configurations do not support on-chip parallel termination with calibration. Figure 6–19 shows on-chip parallel termination with calibration. When parallel OCT is used, the V_{CCIO} of the bank must match the I/O standard of the pin at which the parallel OCT is enabled.

Figure 6–19. Stratix IV On-Chip Parallel Termination with Calibration



The on-chip parallel termination calibration circuit compares the total impedance of the I/O buffer to the external 50-Ω ±1% resistors connected to the RUP and RDN pins and dynamically enables or disables the transistors until they match. Calibration occurs at the end of the device configuration. When the calibration circuit finds the correct impedance, it powers down and stops changing the characteristics of the drivers. Table 6–7 shows the list of I/O standards that support on-chip parallel termination with calibration.

Table 6–7. ***Selectable I/O Standards with On-Chip Parallel Termination with Calibration***

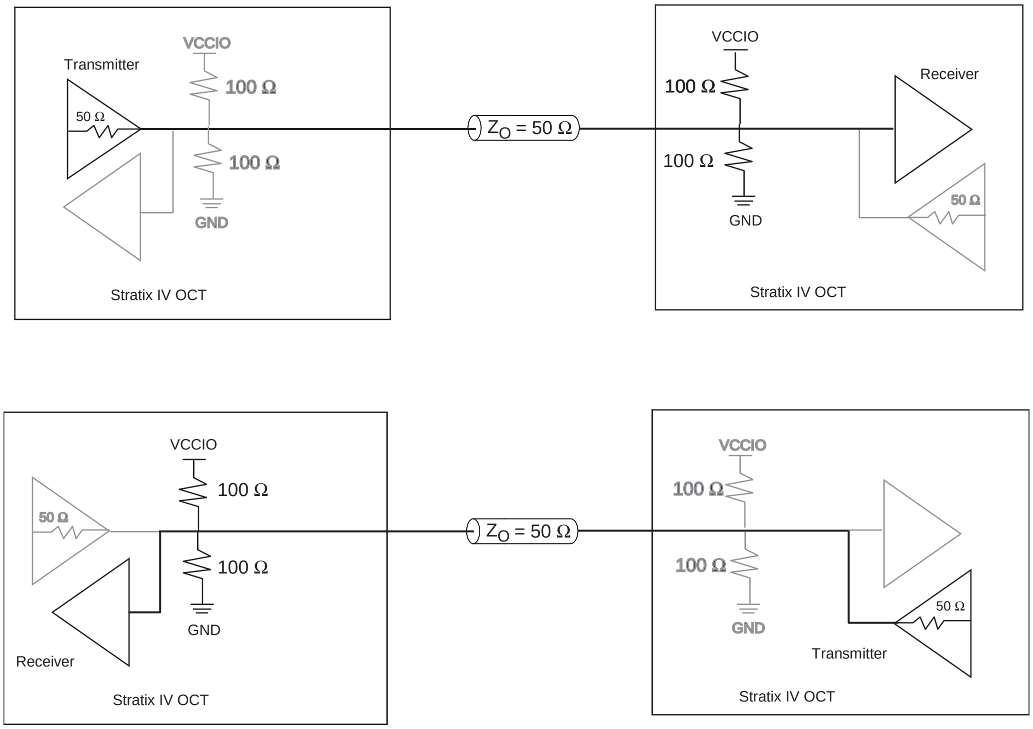
I/O Standard	On-Chip Parallel Termination Setting (Column I/O)	On-Chip Parallel Termination Setting (Row I/O)	Unit
SSTL-2 Class I, II	50	50	Ω
SSTL-18 Class I, II	50	50	Ω
SSTL-15 Class I, II	50	50	Ω
HSTL-18 Class I, II	50	50	Ω
HSTL-15 Class I, II	50	50	Ω
HSTL-12 Class I, II	50	50	Ω
Differential SSTL-2 Class I, II	50	50	Ω
Differential SSTL-18 Class I, II	50	50	Ω
Differential SSTL-15 Class I, II	50	50	Ω
Differential HSTL-18 Class I, II	50	50	Ω
Differential HSTL-15 Class I, II	50	50	Ω
Differential HSTL-12 Class I, II	50	50	Ω

Dynamic On-Chip Termination

Stratix IV devices support on-off dynamic termination, both series and parallel, for a bidirectional I/O in all I/O banks. Figure 6–20 shows the termination schemes supported in these Stratix IV devices. Dynamic parallel termination is enabled only when the bidirectional I/O acts as a receiver and is disabled when it acts as a driver. Similarly, dynamic series termination is enabled only when the bidirectional I/O acts as a driver and is disabled when it acts as a receiver. This feature is useful for terminating any high-performance bidirectional path because the signal integrity is optimized depending on the direction of the data.

Using dynamic OCT helps save power because the device termination is internal instead of external. The termination only switches on during input operation, thus drawing less static power.

Figure 6–20. Dynamic Parallel OCT in Stratix IV Devices

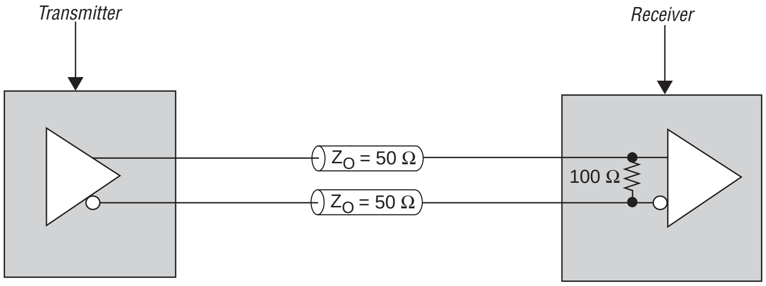


For more information about tolerance specifications for on-chip termination with calibration, refer to the *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook*.

LVDS Input On-Chip Termination (R_D)

Stratix IV devices support on-chip termination for differential LVDS input buffers with a nominal resistance value of 100 Ω, as shown in Figure 6–21. Differential OCT R_D is only available in row I/O banks; column I/O banks do not support OCT $R_{D.}$ The dedicated clock input pairs `CLK[1,3,8,10][p,n]`, `PLL_L[1,4]_CLK[p,n]`, and `PLL_R[1,4]_CLK[p,n]` on the row I/O banks of Stratix IV devices do not support R_D termination.

Figure 6–21. Differential Input On-Chip Termination



For more information about differential on-chip termination, refer to the *High Speed Differential I/O Interfaces with DPA* chapter in volume 1 of the *Stratix IV Device Handbook.*

OCT Calibration

Stratix IV devices support calibrated on-chip series termination (R_S) and calibrated on-chip parallel termination (R_T) on all I/O pins. You can calibrate the devices' I/O bank with any of the OCT calibration blocks (CB) available in the device.

OCT Calibration Block Location

Table 6–8 and Table 6–9 show the location of OCT calibration blocks in Stratix IV devices. Table 6–8 shows the OCT calibration blocks in Banks 1A through 4C. Table 6–9 shows the OCT calibration blocks in Banks 5A through 8C. For both tables, the following legend applies:

- "✓" indicates I/O banks with OCT calibration block
- "X" indicates I/O banks without OCT calibration block
- "—" indicates I/O banks that are not available in the device

Table 6–8. OCT Calibration Block Counts and Placement in Stratix IV Devices (1A through 4C) (Part 1 of 2) *(Note 1)*

Device	Pin	Number of OCT Blocks	Bank											
			1A	1B	1C	2A	2B	2C	3A	3B	3C	4A	4B	4C
EP4SE110	780	8	✓	—	X	✓	—	X	✓	—	X	✓	—	X
EP4SE230	780	8	✓	—	X	✓	—	X	✓	—	X	✓	—	X
EP4SE290	780	8	✓	—	X	✓	—	X	✓	—	X	✓	—	X
	1152	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
	1517	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
EP4SE360	780	8	✓	—	X	✓	—	X	✓	—	X	✓	—	X
	1152	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
	1517	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
EP4SE530	1152	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
	1517	10	✓	X	X	✓	X	X	✓	X	✓	✓	X	X
	1932	10	✓	X	X	✓	X	X	✓	X	✓	✓	X	X

Table 6–8. OCT Calibration Block Counts and Placement in Stratix IV Devices (1A through 4C) (Part 2 of 2) *(Note 1)*

Device	Pin	Number of OCT Blocks	Bank											
			1A	1B	1C	2A	2B	2C	3A	3B	3C	4A	4B	4C
EP4SE680	1152	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
	1517	10	✓	X	X	✓	X	X	✓	X	✓	✓	X	X
	1932	10	✓	X	X	✓	X	X	✓	X	✓	✓	X	X
EP4SGX70	780	8	✓	—	X	✓	—	X	✓	—	X	✓	—	X
EP4SGX110	780	8	✓	—	X	✓	—	X	✓	—	X	✓	—	X
	1152	8	✓	—	X	—	—	—	✓	—	X	✓	—	X
EP4SGX230	780	8	✓	—	X	✓	—	X	✓	—	X	✓	—	X
	1152	8	✓	—	X	—	—	—	✓	X	X	✓	X	X
	1517	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
EP4SGX290	780	8	—	—	—	—	—	—	✓	—	X	✓	—	X
	1152	8	✓	—	X	—	—	—	✓	X	X	✓	X	X
	1517	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
	1932	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
EP4SGX360	780	8	—	—	—	—	—	—	✓	—	X	✓	—	X
	1152	8	✓	—	X	—	—	—	✓	X	X	✓	X	X
	1517	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
	1932	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
EP4SGX530	1152	8	✓	—	X	—	—	—	✓	X	X	✓	X	X
	1517	10	✓	—	X	✓	—	✓	✓	X	X	✓	X	X
	1932	10	✓	X	X	✓	—	✓	✓	X	X	✓	X	X

Note to Table 6–8:

(1) This table does not show transceiver banks and transceiver calibration blocks.

Table 6–9. OCT Calibration Block Counts and Placement in Stratix IV Devices (5A through 8C) (Part 1 of 2) *(Note 1)*

Device	Pin	Number of OCT Blocks	Bank											
			5A	5B	5C	6A	6B	6C	7A	7B	7C	8A	8B	8C
EP4SE110	780	8	✓	—	X	✓	—	X	✓	—	X	✓	—	X
EP4SE230	780	8	✓	—	X	✓	—	X	✓	—	X	✓	—	X
EP4SE290	780	8	✓	—	X	✓	—	X	✓	—	X	✓	—	X
	1152	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
	1517	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
EP4SE360	780	8	✓	—	X	✓	—	X	✓	—	X	✓	—	X
	1152	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
	1517	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
EP4SE530	1152	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
	1517	10	✓	X	X	✓	X	X	✓	X	X	✓	X	✓
	1932	10	✓	X	X	✓	X	X	✓	X	X	✓	X	✓

Table 6–9. OCT Calibration Block Counts and Placement in Stratix IV Devices (5A through 8C) (Part 2 of 2) *(Note 1)*

			Bank											
Device	**Pin**	**Number of OCT Blocks**	**5A**	**5B**	**5C**	**6A**	**6B**	**6C**	**7A**	**7B**	**7C**	**8A**	**8B**	**8C**
EP4SE680	1152	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
	1517	10	✓	X	X	✓	X	X	✓	X	X	✓	X	✓
	1932	10	✓	X	X	✓	X	X	✓	X	X	✓	X	✓
EP4SGX70	780	8	—	—	—	—	—	—	✓	—	X	✓	—	X
EP4SGX110	780	8	—	—	—	—	—	—	✓	—	X	✓	—	X
	1152	8	—	—	—	✓	—	X	✓	—	X	✓	—	X
EP4SGX230	780	8	—	—	—	—	—	—	✓	—	X	✓	—	X
	1152	8	—	—	—	✓	—	X	✓	X	X	✓	✓	X
	1517	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
EP4SGX290	780	8	—	—	—	—	—	—	✓	—	X	✓	—	X
	1152	8	—	—	—	✓	—	X	✓	X	X	✓	X	X
	1517	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
	1932	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
EP4SGX360	780	8	—	—	—	—	—	—	✓	—	X	✓	—	X
	1152	8	—	—	—	✓	—	X	✓	X	X	✓	X	X
	1517	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
	1932	8	✓	—	X	✓	—	X	✓	X	X	✓	X	X
EP4SGX530	1152	8	—	—	—	✓	—	X	✓	X	X	✓	X	X
	1517	10	✓	—	X	✓	—	X	✓	X	X	✓	X	✓
	1932	10	✓	—	X	✓	X	X	✓	X	X	✓	X	✓

Note to Table 6–9:

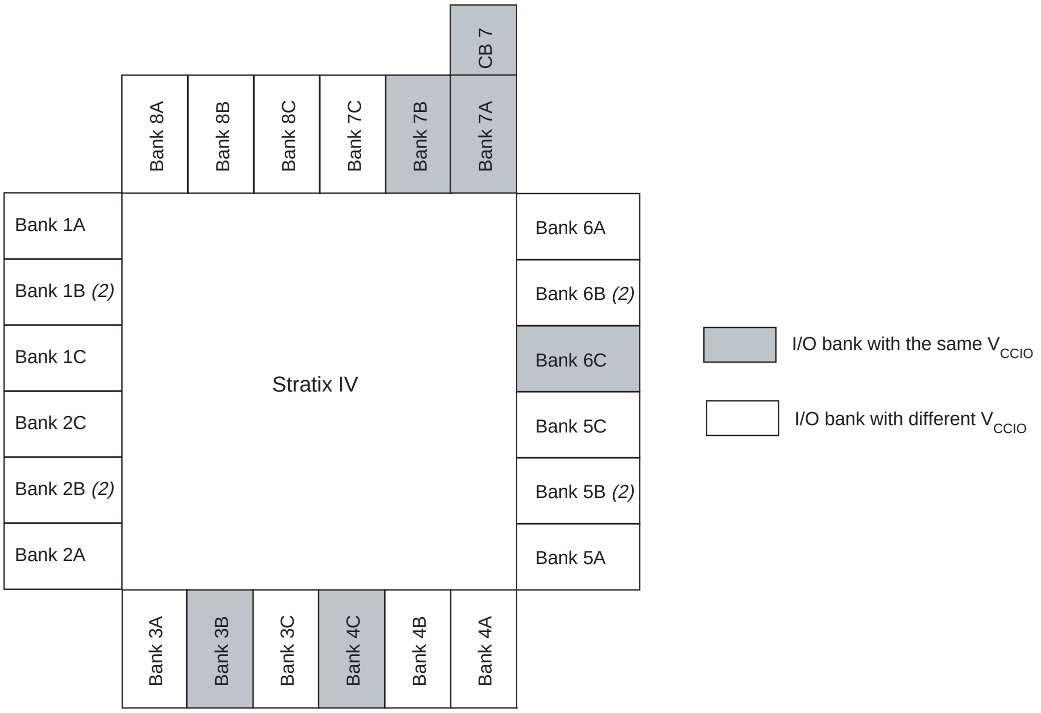
(1) This table does not show transceiver banks and transceiver calibration blocks.

Sharing OCT Calibration Block on Multiple I/O Banks

An OCT calibration block has the same V_{CCIO} as the I/O bank that contains the block. OCT R_S calibration is supported on all I/O banks with different V_{CCIO} voltage standards, up to the number of available OCT calibration blocks. You can configure I/O banks to receive calibration codes from any OCT calibration block with the same V_{CCIO}. All I/O banks with the same V_{CCIO} can share one OCT calibration block, even if that particular I/O bank has an OCT calibration block.

For example, Figure 6–22 shows a group of I/O banks that has the same V_{CCIO} voltage. If a group of I/O banks has the same V_{CCIO} voltage, you can use one OCT calibration block to calibrate the group of I/O banks placed around the periphery. Because 3B, 4C, 6C, and 7B have the same V_{CCIO} as bank 7A, you can calibrate all four I/O banks (3B, 4C, 6C, and 7B) with the OCT calibration block located in bank 7A. You can enable this by serially shifting out OCT R_S calibration codes from the OCT calibration block located in bank 7A to the I/O banks located around the periphery.

Figure 6–22. Example of Calibrating Multiple I/O Banks with One Shared OCT Calibration Block *(Note 1)*, *(2)*



Note to Figure 6–22:

(1) Figure 6–22 is a top view of the silicon die that corresponds to a reverse view for flip chip packages. It is a graphical representation only. This figure does not show transceiver banks and transceiver calibration blocks.

(2) Banks 1B, 2B, 5B and 6B are available in Stratix IV devices only.

OCT Calibration Block Modes of Operation

Stratix IV devices support OCT R_S and OCT R_T on all I/O banks. The calibration can occur in either power-up mode or user mode.

Power-Up Mode

In power-up mode, OCT calibration is automatically performed at power up. Calibration codes are shifted to selected I/O buffers before transitioning to user mode.

User Mode

In user mode, the `OCTUSRCLK`, `ENAOCT`, `nCLRUSR`, and `ENASER[9..0]` signals are used to calibrate and serially transfer calibration codes from each OCT calibration block to any I/O. Table 6–10 shows the user-controlled calibration block signal names and their descriptions.

Table 6–10. OCT Calibration Block Ports for User Control (Part 1 of 2)

Signal Name	Description
OCTUSRCLK	Clock for OCT block.
ENAOCT	Enable OCT Termination (Generated by user IP).

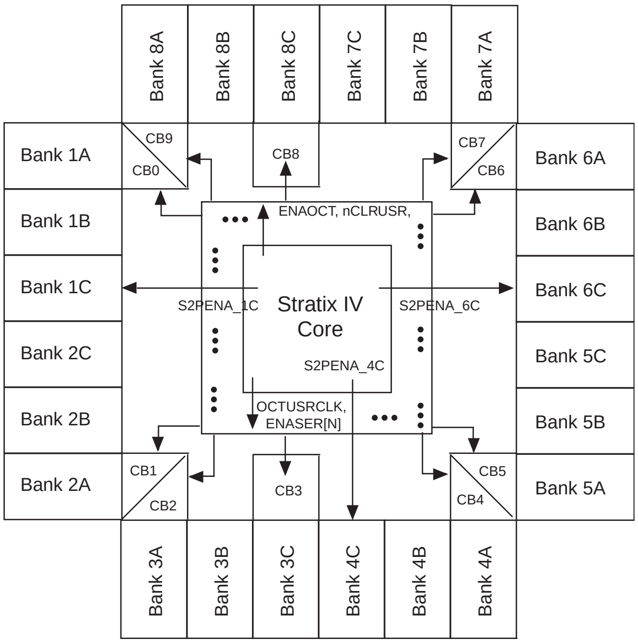
Table 6–10. OCT Calibration Block Ports for User Control (Part 2 of 2)

Signal Name	Description
`ENASER[9..0]`	When `ENOCT` = 0, each signal enables the OCT serializer for the corresponding OCT calibration block. When `ENAOCT` = 1, each signal enables OCT calibration for the corresponding OCT calibration block.
`S2PENA_<bank#>`	Serial-to-parallel load enable per I/O bank.
`nCLRUSR`	Clear user.

Figure 6–23 shows the flow of the user signal. When `ENAOCT` is 1, all OCT calibration blocks are in calibration mode; when `ENAOCT` is 0, all OCT calibration blocks are in serial data transfer mode. The `OCTUSRCLK` clock frequency must be 20 MHz or less.

You must generate all user signals on the rising edge of `OCTUSRCLK`.

Figure 6–23. Signals Used for User Mode Calibration *(Note 1)*



Note to Figure 6–23:

(1) This figure does not show transceiver banks and transceiver calibration blocks.

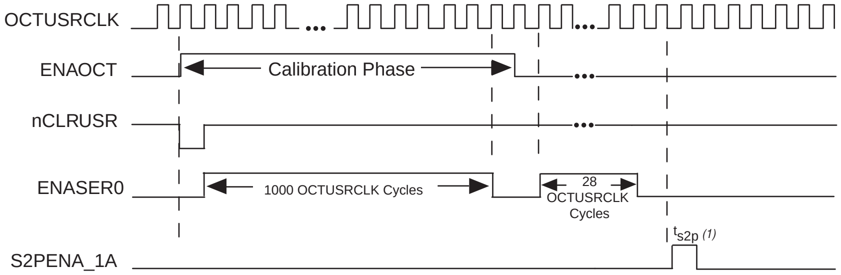
OCT Calibration

Figure 6–24 shows user mode signal-timing waveforms. To calibrate OCT block[N] (where N is a calibration block number), you must assert `ENAOCT` one cycle before asserting `ENASER[N]`. Also, `nCLRUSR` must be set to low for one `OCTUSRCLK` cycle before the `ENASER[N]` signal is asserted. Assert the `ENASER[N]` signals for 1000 `OCTUSRCLK` cycles to perform OCTRS and OCTRT calibration. `ENAOCT` can be deasserted one clock cycle after the last `ENASER` is deasserted.

Serial Data Transfer

After you complete calibration, you must serially shift out the 28-bit OCT calibration codes (14-bit OCT RS code and 14-bit OCT RT) from each OCT calibration block to the corresponding I/O buffers. Only one OCT calibration block can send out the codes at any given time by asserting only one ENASER[N] signal at a time. After ENAOCT is deasserted, you must wait at least one OCTUSRCLK cycle to enable any ENASER[N] signal to begin serial transfer. To shift the 28-bit code from the OCT calibration block[N], ENASER[N] must be asserted for exactly 28 OCTUSRCLK cycles. Between two consecutive asserted ENASER signals, there must be at least one OCTUSRCLK cycle gap. Refer to Figure 6–24 for these requirements.

Figure 6–24. OCT User Mode Signal-Timing Waveform for One OCT Block *(Note 1)*



Note to Figure 6–24:

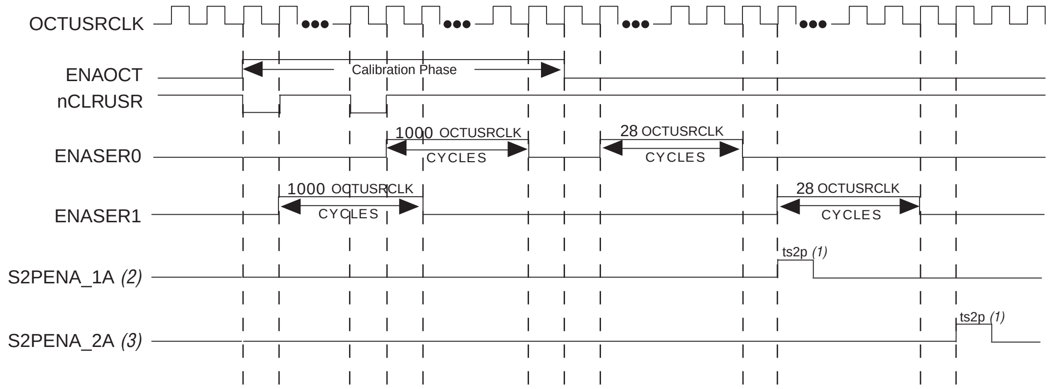
(1) $t_{s2p} \geq 25$ ns.

After calibrated codes are shifted in serially to each I/O bank, the calibrated codes must be converted from serial format to parallel format before being used in the I/O buffers. Figure 6–24 shows S2PENA signals that can be asserted at any time to update the calibration codes in each I/O bank. All I/O banks that received the codes from the same OCT calibration block can have S2PENA asserted at the same time, or at a different time, even while another OCT calibration block is calibrating and serially shifting codes. The S2PENA signal is asserted one OCTUSRCLK cycle after ENASER is deasserted for at least 25 ns. You cannot use I/Os for transmitting or receiving data when their S2PENA is asserted for parallel codes transfer.

Example of Using Multiple OCT Calibration Blocks

Figure 6–25 shows a signal timing waveform for two OCT calibration blocks doing RS and RT calibration. Calibration blocks can start calibrating at different times by asserting ENASER signals at different times. ENAOCT must stay asserted while any calibration is ongoing. nCLRUSR must be set to low for one OCTUSRCLK cycle before each ENASER[N] signal is asserted. In Figure 6–25, when nCLRUSR is set to 0 for the second time to initialize OCT calibration block 0, this does not affect OCT calibration block 1, whose calibration is already in progress.

Figure 6–25. OCT User-Mode Signal Timing Waveform for Two OCT Blocks *(Note 1), (2), (3)*



Notes to Figure 6–25:

(1) `ts2p` ≥ 25 ns.

(2) `S2PENA_1A` is asserted in Bank 1A for calibration block 0.

(3) `S2PENA_2A` is asserted in Bank 2A for calibration block 1.

RS Calibration

If only R_S calibration is used for an OCT calibration block, its corresponding `ENASER` signal only needs to be asserted for 240 `OCTUSRCLK` cycles for calibration.

You still have to assert the `ENASER` signal for 28 `OCTUSRCLK` cycles for serial transfer.

Termination Schemes for I/O Standards

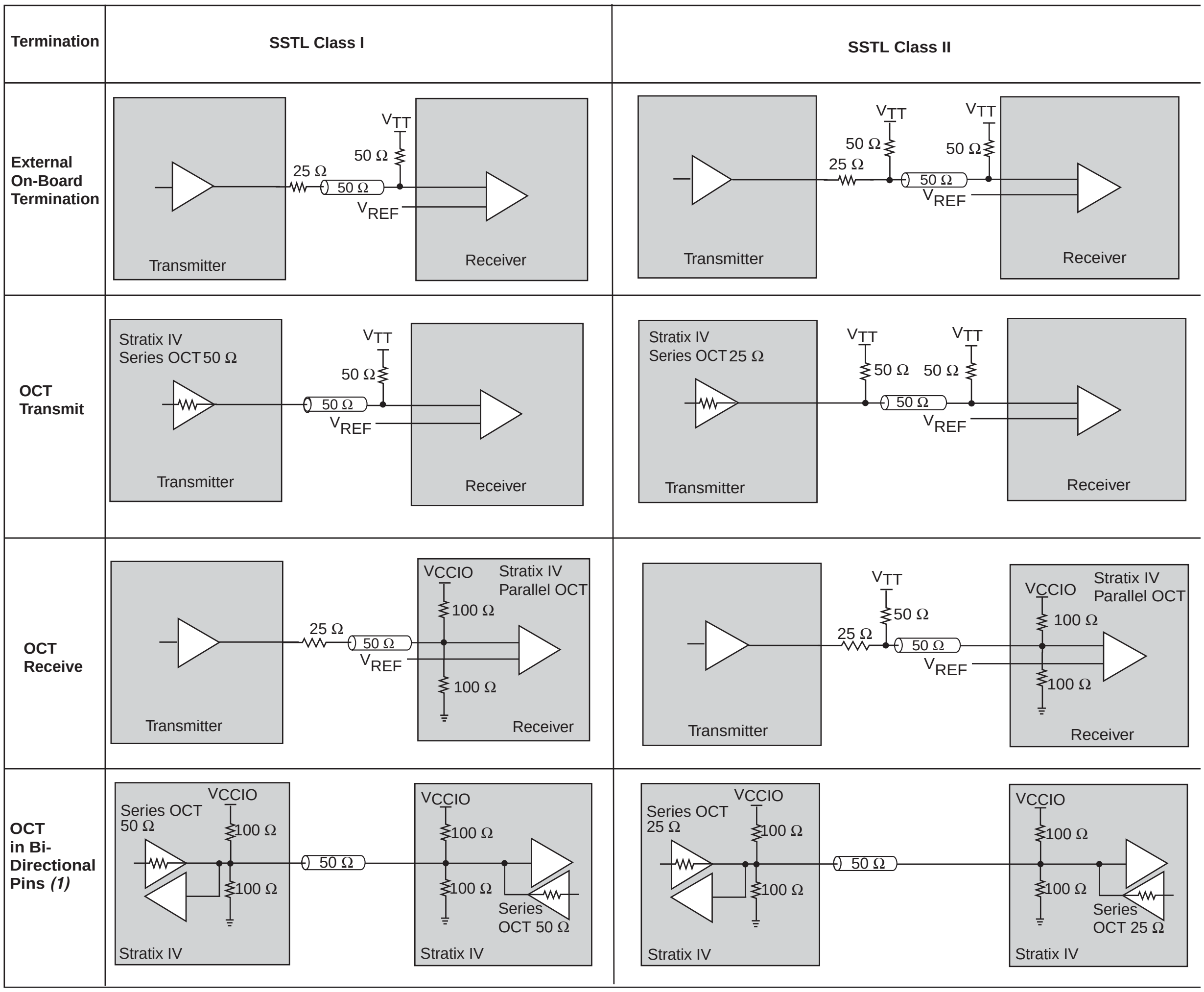
The following section describes the different termination schemes for the I/O standards used in Stratix IV devices.

Single-Ended I/O Standards Termination

Voltage-referenced I/O standards require both an input reference voltage, V_{REF}, and a termination voltage, V_{TT}. The reference voltage of the receiving device tracks the termination voltage of the transmitting device.

Figure 6–26 and Figure 6–27 show the details of SSTL and HSTL I/O termination on Stratix IV devices.

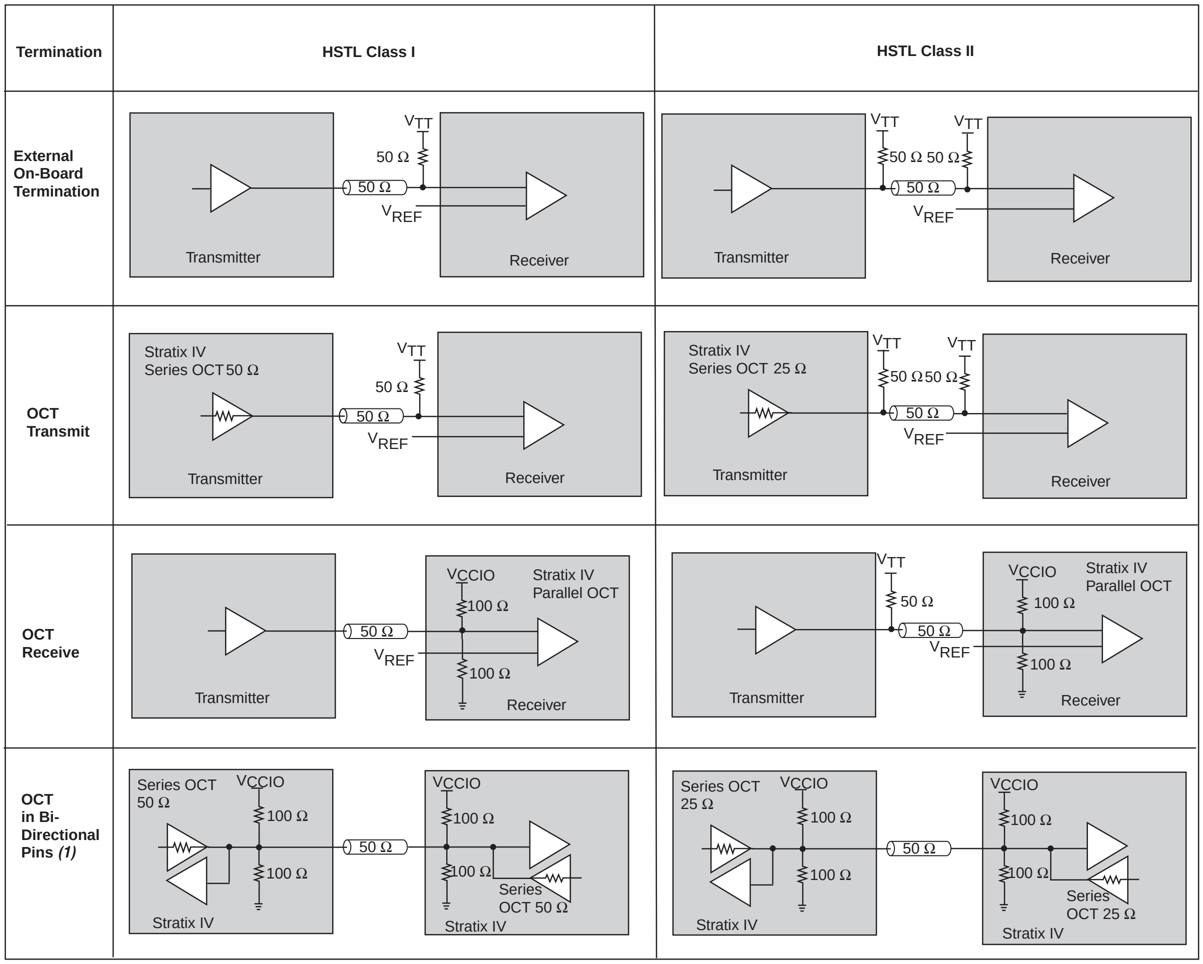
Figure 6–26. SSTL I/O Standard Termination *(Note 1)*



Note to Figure 6–26:

(1) In Stratix IV devices, series and parallel OCT cannot be used simultaneously. For more information, refer to "Dynamic On-Chip Termination" on page 6–28.

Figure 6–27. HSTL I/O Standard Termination



Note to Figure 6–27:

(1) In Stratix IV devices, series and parallel OCT cannot be used simultaneously. For more information, refer to "Dynamic On-Chip Termination" on page 6–28.

Differential I/O Standards Termination

Stratix IV devices support differential SSTL-18 and SSTL-2, differential HSTL-18, HSTL-15, HSTL-12, LVDS, LVPECL, RSDS, and mini-LVDS. Figure 6–28 through Figure 6–34 show the details of various differential I/O terminations on these devices.

☞ Differential HSTL and SSTL outputs are not true differential outputs. They use two single-ended outputs with the second output programmed as inverted.

Figure 6–28. Differential SSTL I/O Standard Termination

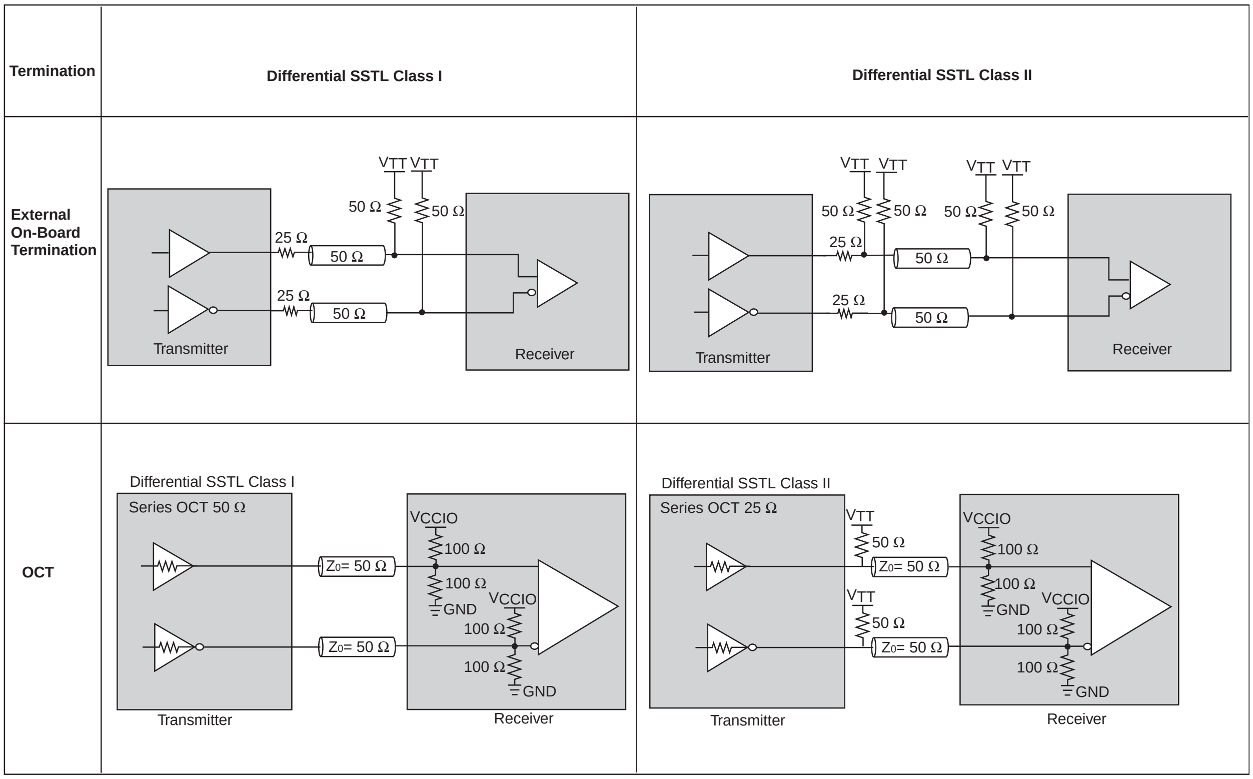
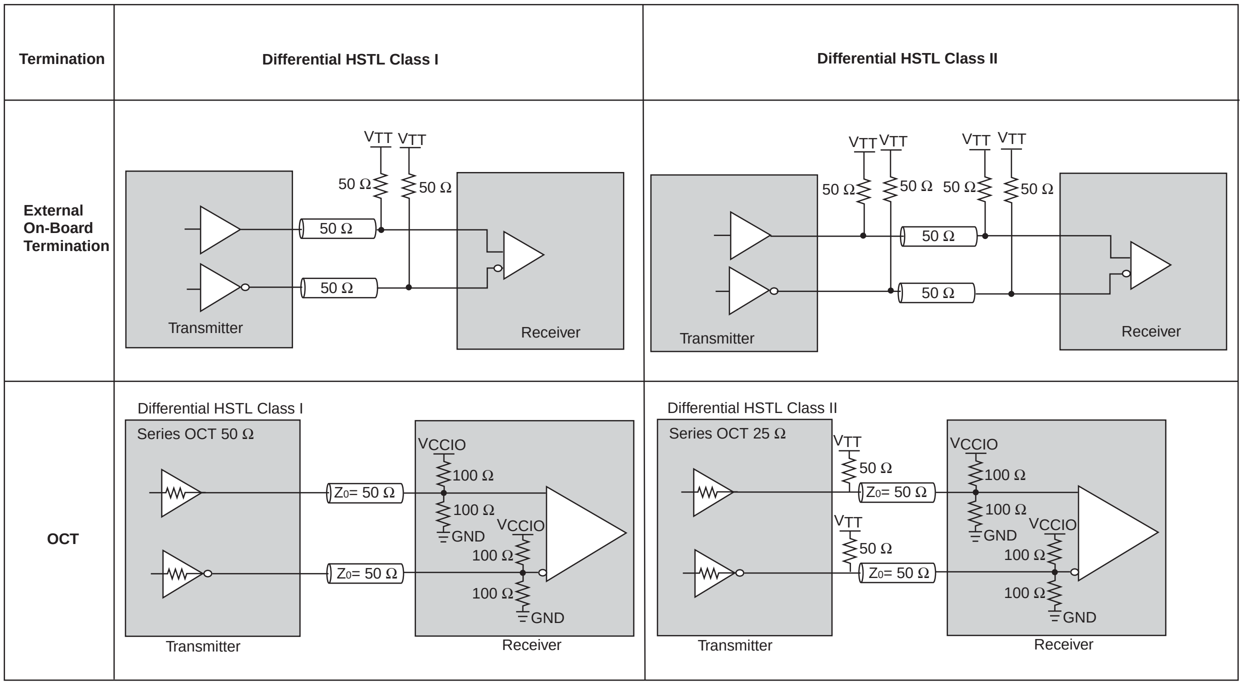


Figure 6–29. Differential HSTL I/O Standard Termination

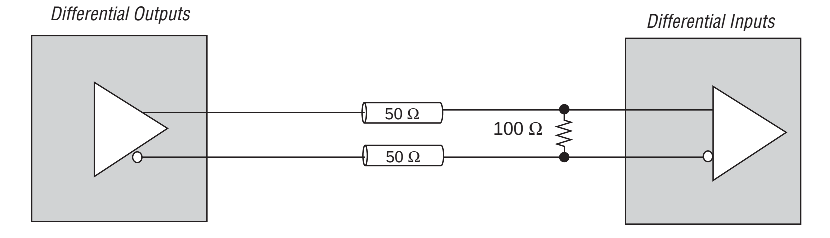
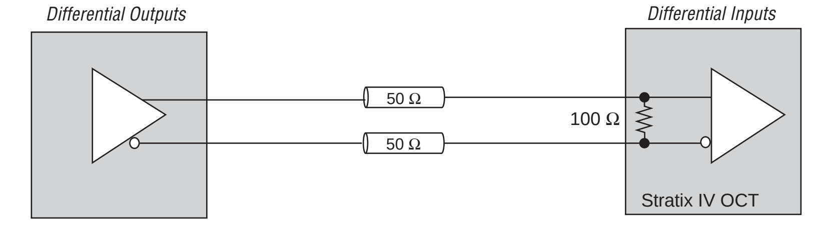
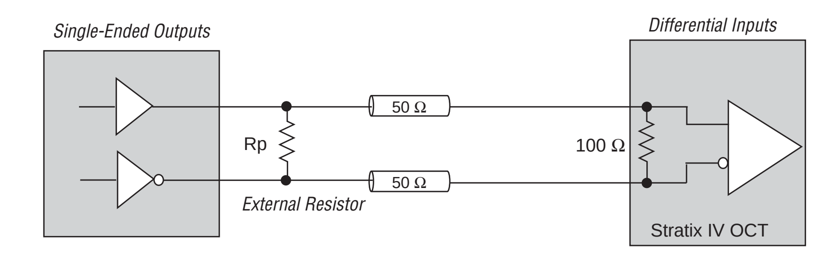
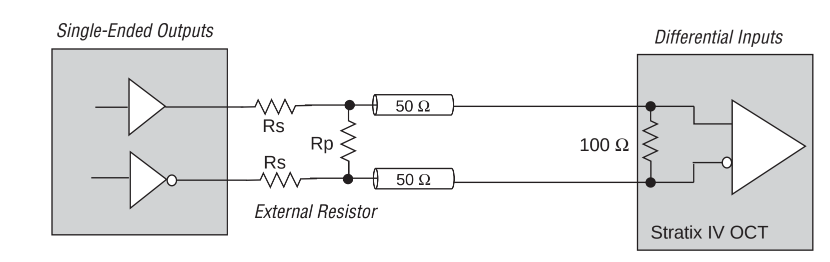


LVDS

The LVDS I/O standard is a differential high-speed, low-voltage swing, low-power, general-purpose I/O interface standard. In Stratix IV devices, the LVDS I/O standard requires a 2.5-V V_{CCIO} level. The LVDS input buffer requires 2.5-V V_{CCPD}. Use this standard in applications requiring high-bandwidth data transfer, such as backplane drivers, and clock distribution. LVDS requires a 100-Ω termination resistor between the two signals at the input buffer. Stratix IV devices provide an optional 100-Ω differential termination resistor in the device using on-chip differential termination.

Figure 6–30 shows the details of LVDS termination. The on-chip differential resistor is only available in row I/O banks. The one-resistor topology is for data rates up to 200 Mbps. The three-resistor topology is for data rates above 200 Mbps.

Figure 6–30. LVDS I/O Standard Termination *(Note 1)*, *(2)*, *(3)*

Termination	LVDS
External On-Board Termination	 Differential Outputs Differential Inputs 50 Ω 50 Ω 100 Ω
OCT Receive (Dedicated LVDS Output) *(2)*	 Differential Outputs Differential Inputs 50 Ω 50 Ω 100 Ω Stratix IV OCT
OCT Receive (Single-Ended LVDS Output with One Resistor Network, LVDS_E_1R) *(3)*	 Single-Ended Outputs Differential Inputs Rp 50 Ω 50 Ω 100 Ω External Resistor Stratix IV OCT
OCT Receive (Single-Ended LVDS Output with Three Resistor Network, LVDS_E_3R) *(3)*	 Single-Ended Outputs Differential Inputs Rs Rs Rp 50 Ω 50 Ω 100 Ω External Resistor Stratix IV OCT

Notes to Figure 6–30:

(1) The R_S and R_P values are pending characterization.

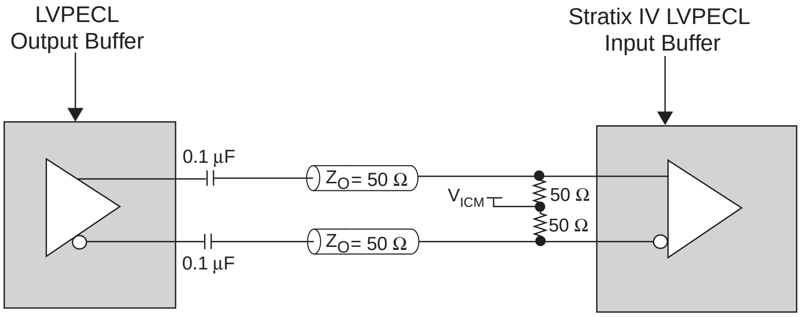
(2) Side I/O banks support dedicated LVDS output buffers.

(3) Column I/O banks support LVDS_E_1R and LVDS_E_3R I/O standards using two single-ended output buffers.

Differential LVPECL

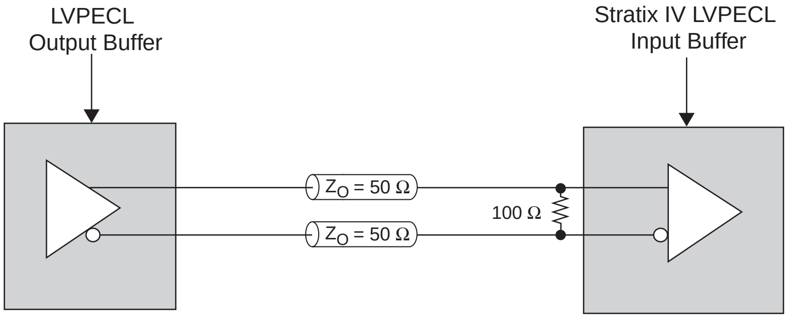
In Stratix IV devices, the LVPECL I/O standard is supported on input clock pins on column and row I/O banks. LVPECL output operation is not supported by Stratix IV devices. LVDS input buffers are used to support LVPECL input operation. AC coupling is required when the LVPECL common-mode voltage of the output buffer is higher than the LVPECL input common-mode voltage. Figure 6–31 shows the AC-coupled termination scheme. The 50-Ω resistors used at the receiver end are external to the device.

Figure 6–31. LVPECL AC-Coupled Termination



DC-coupled LVPECL is supported if the LVPECL output common mode voltage is within the Stratix IV LVPECL input buffer specification (Figure 6–32).

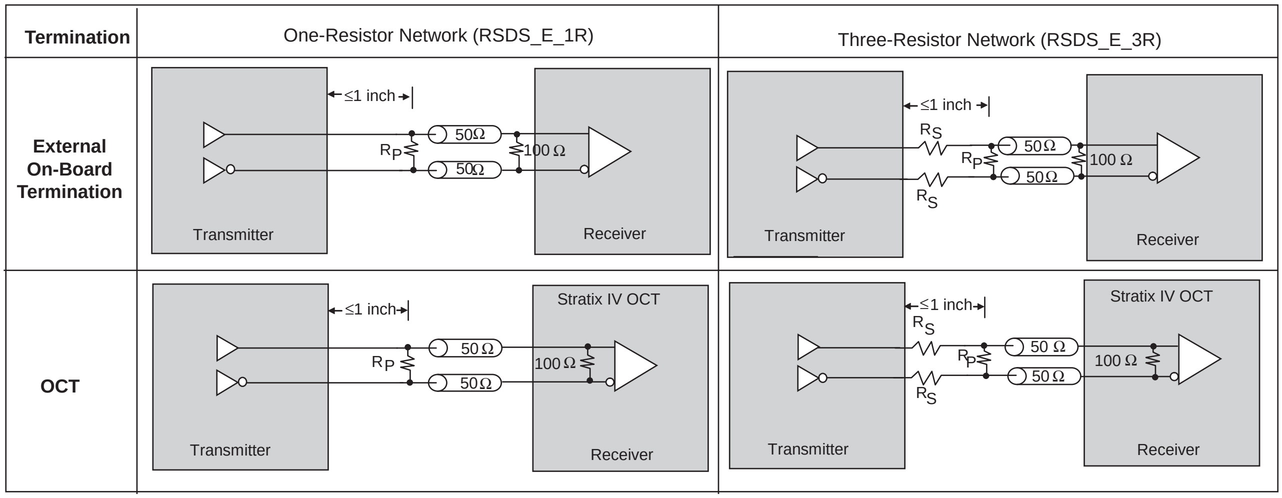
Figure 6–32. LVPECL DC-Coupled Termination



RSDS

Stratix IV devices support the RSDS output standard with data rates up to 230 Mbps using LVDS output buffer types. For transmitters, use two single-ended output buffers with the external one- or three-resistor networks in the column I/O bank, as shown in Figure 6–33. The one-resistor topology is for data rates up to 200 Mbps. The three-resistor topology is for data rates above 200 Mbps. The row I/O banks support RSDS output using dedicated LVDS output buffers without an external resistor network.

Figure 6–33. RSDS I/O Standard Termination *(Note 1)*


Termination
One-Resistor Network (RSDS_E_1R)
Three-Resistor Network (RSDS_E_3R)
External On-Board Termination
OCT
≤1 inch
Transmitter
Receiver
Stratix IV OCT
R_P
R_S
50Ω
50 Ω
100 Ω

Note to Figure 6–33:

(1) The R_S and R_P values are pending characterization.

A resistor network is required to attenuate the LVDS output-voltage swing to meet the RSDS specifications. You can modify the three-resistor network values to reduce power or improve the noise margin. The resistor values chosen should satisfy Equation 6–1:

Equation 6–1.

$$\frac{R_S \times \frac{R_P}{2}}{R_S + \frac{R_P}{2}} = 50\Omega$$

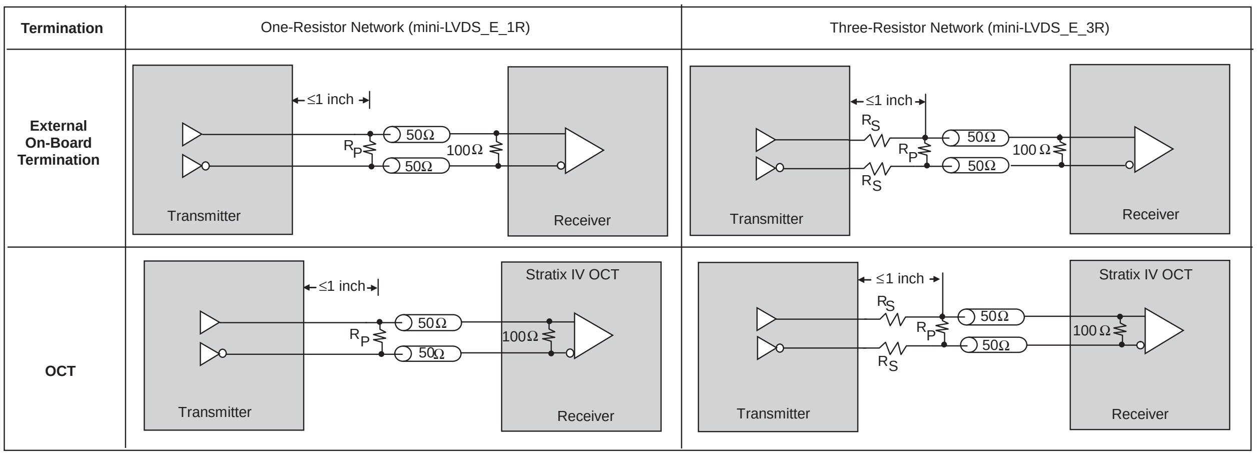
Altera recommends that you perform additional simulations using IBIS models to validate that custom resistor values meet the RSDS requirements.

For more information about the RSDS I/O standard, refer to the *RSDS Specification* from the National Semiconductor website at www.national.com.

Mini-LVDS

Stratix IV devices support the mini-LVDS output standard with data rates up to 340 Mbps using LVDS output buffer types. For transmitters, use two single-ended output buffers with external one- or three-resistor networks, as shown in Figure 6–34. The one-resistor topology is for data rates up to 200 Mbps. The three-resistor topology is for data rates above 200 Mbps. The row I/O banks support mini-LVDS output using dedicated LVDS output buffers without an external resistor network.

Figure 6–34. Mini-LVDS I/O Standard Termination *(Note 1)*


Termination
One-Resistor Network (mini-LVDS_E_1R)
Three-Resistor Network (mini-LVDS_E_3R)
External On-Board Termination
OCT
≤1 inch
50Ω
100Ω
R_P
R_S
Transmitter
Receiver
Stratix IV OCT

Note to Figure 6–34:

(1) The R_S and R_P values are pending characterization.

A resistor network is required to attenuate the LVDS output voltage swing to meet the mini-LVDS specifications. You can modify the three-resistor network values to reduce power or improve the noise margin. The resistor values chosen should satisfy Equation 6–2:

Equation 6–2.

$$\frac{R_S \times \frac{R_P}{2}}{R_S + \frac{R_P}{2}} = 50\Omega$$

Altera recommends that you perform additional simulations using IBIS models to validate that custom resistor values meet the RSDS requirements.

For more information about the mini-LVDS I/O standard, see the *mini-LVDS Specification* from the Texas Instruments website at www.ti.com.

Design Considerations

While Stratix IV devices feature various I/O capabilities for high-performance and high-speed system designs, there are several other design considerations that require your attention to ensure the success of your designs.

I/O Bank Restrictions

Each I/O bank can simultaneously support multiple I/O standards. The following sections provide guidelines for mixing non-voltage-referenced and voltage-referenced I/O standards in Stratix IV devices.

Non-Voltage-Referenced Standards

Each I/O bank of a Stratix IV device has its own VCCIO pins and supports only one V_{CCIO}, either 1.2, 1.5, 1.8, 2.5, or 3.0 V. An I/O bank can simultaneously support any number of input signals with different I/O standard assignments, as shown in Table 6–2.

For output signals, a single I/O bank supports non-voltage-referenced output signals that are driving at the same voltage as V_{CCIO}. Because an I/O bank can only have one V_{CCIO} value, it can only drive out that one value for non-voltage-referenced signals. For example, an I/O bank with a 2.5-V V_{CCIO} setting can support 2.5-V standard inputs and outputs as well as 3.0-V LVCMOS inputs (but not output or bidirectional pins).

Voltage-Referenced Standards

To accommodate voltage-referenced I/O standards, each Stratix IV device's I/O bank supports multiple VREF pins feeding a common V_{REF} bus. The number of available VREF pins increases as device density increases. If these pins are not used as VREF pins, they cannot be used as generic I/O pins and should be tied to V_{CCIO} or GND. Each bank can only have a single V_{CCIO} voltage level and a single V_{REF} voltage level at a given time.

An I/O bank featuring single-ended or differential standards can support voltage-referenced standards if all voltage-referenced standards use the same V_{REF} setting.

For performance reasons, voltage-referenced input standards use their own V_{CCPD} level as the power source. This feature allows you to place voltage-referenced input signals in an I/O bank with a V_{CCIO} of 2.5 V or below. For example, you can place HSTL-15 input pins in an I/O bank with 2.5-V V_{CCIO}. However, the voltage-referenced input with parallel OCT enabled requires the V_{CCIO} of the I/O bank to match the voltage of the input standard.

Voltage-referenced bidirectional and output signals must be the same as the I/O bank's V_{CCIO} voltage. For example, you can only place SSTL-2 output pins in an I/O bank with a 2.5-V V_{CCIO}.

Mixing Voltage-Referenced and Non-Voltage-Referenced Standards

An I/O bank can support both voltage-referenced and non-voltage-referenced pins by applying each of the rule sets individually. For example, an I/O bank can support SSTL-18 inputs and 1.8-V inputs and outputs with a 1.8-V V_{CCIO} and a 0.9-V V_{REF}. Similarly, an I/O bank can support 1.5-V standards, 1.8-V inputs (but not outputs), and HSTL and HSTL-15 I/O standards with a 1.5-V V_{CCIO} and 0.75-V V_{REF}.

Conclusion

Stratix IV devices provide I/O capabilities that allow you to work in compliance with current and emerging I/O standards and requirements. With these device features, you can reduce board design interface costs and increase development flexibility.

Referenced Documents

This chapter references the following documents:

- *AN 224: High-Speed Board Layout Guidelines*
- *AN 315: Guidelines for Designing High-Speed FPGA PCBs*
- *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook*
- *External Memory Interfaces in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook*
- *High-Speed Differential I/O Interface with DPA* chapter in volume 1 of the *Stratix IV Device Handbook*
- *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*
- *Stratix IV Device I/O Features* chapter in volume 1 of the of the *Stratix IV Device Handbook*

Document Revision History

Table 6–11 shows the revision history for this document.

Table 6–11. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	■ Updated "Modular I/O Banks" on page 6–7 ■ Updated Figure 6–3 ■ Updated Figure 6–22 ■ Made minor editorial changes.	—
May 2008 v1.0	Initial release.	—

7. External Memory Interfaces in Stratix IV Devices

SIV51007-2.0

Introduction

To support the level of system bandwidth achievable with Altera Stratix IV FPGAs, the devices provide an efficient architecture to quickly and easily fit wide external memory interfaces with their small modular I/O bank structure. The I/Os are designed to provide flexible and high-performance support for existing and emerging external double data rate (DDR) memory standards, such as DDR3, DDR2, DDR SDRAM, QDRII+, QDRII SRAM, and RLDRAM II.

Stratix IV I/O elements provide easy-to-use built-in functionality required for a rapid and robust implementation with features such as dynamic calibrated on-chip termination (OCT), trace mismatch compensation, read- and write-leveling circuit for DDR3 SDRAM interfaces, half data rate (HDR) blocks, and 4- to 36-bit programmable DQ group widths.

The high-performance memory interface solution is backed-up by a self-calibrating megafunction (ALTMEMPHY), optimized to take advantage of the Stratix IV I/O structure and the TimeQuest Timing Analyzer, which completes the picture by providing the total solution for the highest reliable frequency of operation across process, voltage, and temperature (PVT) variations.

This chapter contains the following sections:

- "Memory Interfaces Pin Support" on page 7–4
- "Stratix IV External Memory Interface Features" on page 7–27

Table 7–1 and Table 7–2 summarize the maximum clock rate Stratix IV devices can support with external memory devices. The difference in package design in the Stratix IV GX devices lead to performance support difference in memory interfaces.

Table 7–1. Stratix IV Maximum Clock Rate Support for External Memory Interfaces with Half-Rate Controller *(Note 1)*, *(2)*

Memory Standards	Stratix IV GX Devices with 1152-pin (with 24 Transceivers), 1517-Pin, and 1932-Pin Packages Stratix IV E Devices with 1152-Pin, 1517-Pin, and 1760-Pin Packages						Stratix IV GX Devices with 780-Pin and 1152-Pin (with 16 Transceivers) Packages Stratix IV E Devices with 780-Pin Packages					
	-2 Speed Grade (MHz)		-3 Speed Grade (MHz)		-4 Speed Grade (MHz)		-2X Speed Grade (MHz)		-3 Speed Grade (MHz)		-4 Speed Grade (MHz)	
	Column I/O Banks	Row I/O Banks *(3)*	Column I/O Banks	Row I/O Banks *(3)*	Column I/O Banks	Row I/O Banks *(3)*	Column I/O Banks	Row I/O Banks *(3)*	Column I/O Banks	Row I/O Banks *(3)*	Column I/O Banks	Row I/O Banks *(3)*
DDR3 SDRAM *(4)*	533	333	400	333	333	333	333	333	333	333	333	333
DDR2 SDRAM *(4)*	400	333	333	333	333	333	333	333	333	333	333	333
DDR SDRAM *(4)*	200	200	200	200	200	200	200	200	200	200	200	200
QDRII+ SRAM (2.5 clock-cycle latency only) *(5)*, *(6)*	400	300	350	300	300	300	300	300	300	300	300	300
QDRII SRAM (1.5-V & 1.8-V HSTL) *(6)*	350	300	300	300	300	300	300	300	300	300	300	300
RLDRAM II (1.5-V & 1.8-V HSTL)	400	333	333	333	333	333	333	333	333	333	333	333

Notes to Table 7–1:

(1) Numbers are preliminary until characterization is final. The supported operating frequencies listed here are memory interface maximums for the FPGA device family. Your design's actual achievable performance is based on design and system-specific factors, as well as static timing analysis of the completed design.

(2) Column I/Os refer to top and bottom I/Os. Row I/Os refer to left and right I/Os.

(3) The row I/O banks do not support 1.5-V HSTL Class II and 1.5-V SSTL Class II I/O standards.

(4) This applies to interfaces with both modules and components.

(5) The QDRII+ SRAM devices with 2.0 clock-cycle latency are not supported due to hardware limitations.

(6) Stratix IV devices support the combining of ×18 DQS/DQ groups for ×36 QDRII+/QDRII SRAM interfaces at a lower maximum frequencies as detailed in "Combining ×16/×18 DQS/DQ groups for ×36 QDRII+/QDRII SRAM Interface" on page 7–25.

Table 7–2. Stratix IV Maximum Clock Rate Support for External Memory Interfaces with Full-Rate Controller *(Note 1)*, *(2)*, *(3)*

Memory Standards	-2/2X Speed Grade (MHz)		-3 Speed Grade (MHz)		-4 Speed Grade (MHz)	
	Column I/O Banks	Row I/O Banks *(4)*	Column I/O Banks	Row I/O Banks *(4)*	Column I/O Banks	Row I/O Banks *(4)*
DDR2 SDRAM	267	267	233	233	200	200
DDR SDRAM	200	200	200	200	200	200

Notes to Table 7–2:

(1) Numbers are preliminary until characterization is final. The supported operating frequencies listed here are memory interface maximums for the FPGA device family. Your design's actual achievable performance is based on design- and system-specific factors, as well as static timing analysis of the completed design.

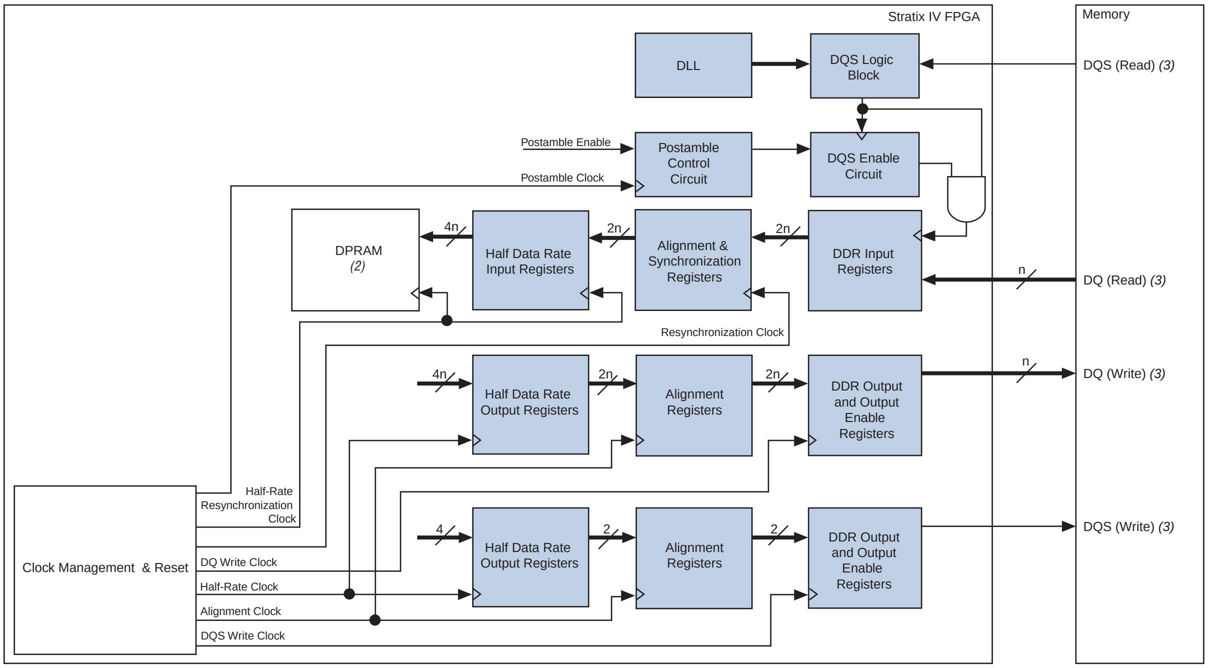
(2) Column I/Os refer to top and bottom I/Os. Row I/Os refer to left and right I/Os.

(3) This applies to interfaces with both modules and components.

(4) The row I/O banks do not support 1.5-V HSTL Class II and 1.5-V SSTL Class II I/O standards.

Figure 7–1 shows the overview of the memory interface data path that uses all the Stratix IV I/O elements (IOE) features.

Figure 7–1. External Memory Interface Data Path Overview *(Note 1)*, *(2)*



Notes to Figure 7–1:

(1) You can bypass each register block.

(2) The blocks used for each memory interface may differ slightly. The shaded blocks are part of the Stratix IV IOE.

(3) These signals may be bidirectional or unidirectional, depending on the memory standard. When bidirectional, the signal is active during both read and write operations.

This chapter describes the hardware features in Stratix IV devices that facilitate high-speed memory interfacing for the DDR memory standard. These features include delay-locked loops (DLLs), dynamic OCT control, and read- and write-leveling circuitry.

Memory interfaces also use I/O features such as OCT, programmable input delay chains, programmable output delay, slew rate adjustment, and programmable drive strength.

For more information about any of the above-mentioned features, refer to the *I/O Features in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook*.

The ALTMEMPHY megafunction instantiates a phase-locked loop (PLL) and PLL reconfiguration logic to adjust the phase shift based on VT variation.

For more information about the Stratix IV PLL, refer to the *Clock Networks and PLLs in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook*. For more information about the ALTMEMPHY megafunction, refer to the *ALTMEMPHY Megafunction User Guide*.

Memory Interfaces Pin Support

A typical memory interface requires data (D, Q, or DQ), data strobe (DQS/CQ and DQSn/CQn), address, command, and clock pins. Some memory interfaces use data mask (DM, BWSn, or NWSn) pins to enable write masking and QVLD pins to indicate that the read data is ready to be captured. This section describes how Stratix IV devices support all these different pins.

Table 7–3 summarizes the pin connections between a Stratix IV device and an external memory device.

Table 7–3. Stratix IV Memory Interfaces Pin Utilization (Part 1 of 2)

Pin Description	Memory Standard	Stratix IV Pin Utilization
Read Data	All	DQ
Write Data	All	DQ *(1)*
Parity, DM, BWSn, NWSn, QVLD, ECC	All	DQ *(1)*, *(2)*
Read Data Strobes/Clocks	DDR3 SDRAM DDR2 SDRAM (with differential DQS signaling) *(3)* RLDRAM II	Differential DQS/DQSn (also used as write data clock)
	DDR2 SDRAM (with single-ended DQS signaling) *(3)* DDR SDRAM	Single-ended DQS (also used as write data clock)
	QDRII+ SRAM QDRII SRAM	Complementary CQ/CQn
Write Data Clocks	QDRII+ SRAM *(4)* QDRII SRAM *(4)* RLDRAM II Separate I/O (SIO)	Any DQS and DQSn pin pairs associated with the DQ groups used for the write data pins *(1)*

Table 7–3. Stratix IV Memory Interfaces Pin Utilization (Part 2 of 2)

Pin Description	Memory Standard	Stratix IV Pin Utilization
Memory Clocks (for Address and Command) *(5)*	DDR3 SDRAM DDR2 SDRAM (with differential DQS signaling) *(3)*	Any unused DQ or DQS pins with `DIFFIO_RX` capability for the `mem_clk[0]` and `mem_clk_n[0]` signals.
		Any unused DQ or DQS pins with `DIFFOUT` capability for the `mem_clk[n:1]` and `mem_clk_n[n:1]` signals (where n is greater than or equal to 1).
	DDR2 SDRAM (with single-ended DQS signaling) *(3)* DDR SDRAM RLDRAM II	Any `DIFFOUT` pins

Notes to Table 7–3:

(1) If the write data signals are unidirectional, connect them, including the data mask pins, to a separate DQS/DQ group other than the read DQS/DQ group. Connect the write clock to the DQS and DQSn pin-pair associated with that DQS/DQ group. Do not use the CQ and CQn pin-pair as write clocks.

(2) The BWSn, NWSn, and DM pins need to be part of the write DQS/DQ group, while parity, QVLD, and ECC pins need to be part of the read DQS/DQ group. The ALTMEMPHY megafunction does not support the QVLD pin. However, if your design supports the QVLD pin, the QVLD pin needs to be part of the read DQS/DQ group as well.

(3) DDR2 SDRAM supports either single-ended or differential DQS signaling.

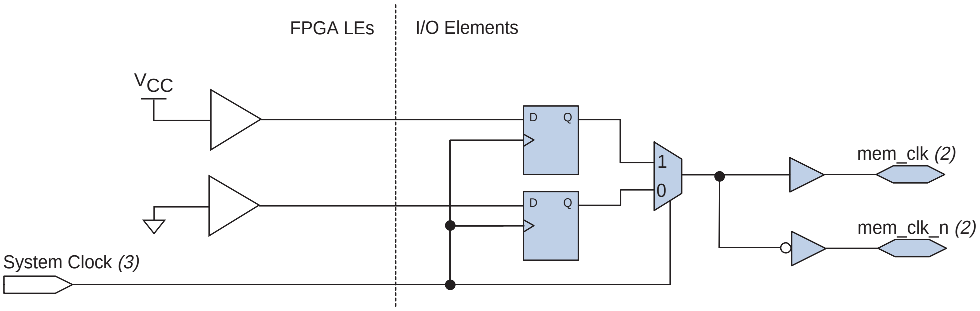
(4) QDRII+/QDRII SRAM devices uses the K/K# clock pin-pair to latch write data, address, and command signals. The clocks must be part of the DQS/DQ group and follow the write data clock rules in this case.

(5) ALTMEMPHY megafunction implementation for a DDR3, DDR2, or DDR SDRAM interface requires that you place all memory clock pin-pairs in a single DQ group of adequate width to minimize skew. For example, DIMMs requiring three memory clock pin-pairs need to use a ×4 DQS/DQ group.

DDR3, DDR2, DDR SDRAM, and RLDRAM II devices use the CK and CK# signals to capture the address and command signals. Generate these signals to mimic the write-data strobe using Stratix IV DDR I/O registers (DDIOs) to ensure that timing relationships between CK/CK# and DQS signals (t_{DQSS}, t_{DSS}, and t_{DSH} in DDR3, DDR2, and DDR SDRAM devices or t_{CKDK} in RLDRAM II devices) are met. QDRII+ and QDRII SRAM devices use the same clock (K/K#) to capture write data, address, and command signals.

The memory clock pins in Stratix IV devices are generated using a DDIO register going to differential output pins (see Figure 7–2), marked in the pin table with `DIFFOUT`, `DIFFIO_TX`, or `DIFFIO_RX` prefixes. For more information about which pins to use for memory clock pins, refer to Table 7–3.

Figure 7–2. Memory Clock Generation *(Note 1)*, *(2)*, *(3)*



Notes to Figure 7–2:

(1) Refer to Table 7–3 for pin location requirements for these pins.

(2) The `mem_clk[0]` and `mem_clk_n[0]` pins for DDR3, DDR2, and DDR SDRAM interfaces use the I/O input buffer for feedback required by the ALTMEMPHY megafunction for tracking, hence bidirectional I/O buffers are used for these pins. For memory interfaces using a differential DQS input, the input feedback buffer is configured as differential input; for memory interfaces using a single-ended DQS input, the input buffer is configured as a single-ended input. Using a single-ended input feedback buffer requires that I/O standard's VREF voltage is provided to that I/O bank's VREF pins.

(3) Regional clock networks are required for memory output clock generation to minimize jitter.

Stratix IV devices offer differential input buffers for differential read-data strobe and clock operations. In addition, Stratix IV devices also provide an independent DQS logic block for each CQn pin for complementary read-data strobe and clock operations. In the Stratix IV pin tables, the differential DQS pin pairs are denoted as DQS and DQSn pins, while the complementary CQ signals are denoted as CQ and CQn pins. DQSn and CQn pins are marked separately in the pin table. Each CQn pin connects to a DQS logic block and the shifted CQn signals go to the negative-edge input registers in the DQ IOE registers.

Use differential DQS signaling for DDR2 SDRAM interfaces running at or above 333 MHz.

The DQ pins can be bidirectional signals, as in DDR3, DDR2, and DDR SDRAM, and RLDRAM II common I/O (CIO) interfaces, or unidirectional signals, as in QDRII+, QDRII SRAM, and RLDRAM II separate I/O (SIO) devices. Connect the unidirectional read-data signals to Stratix IV DQ pins and the unidirectional write-data signals to a different DQS/DQ group than the read DQS/DQ group. Furthermore, the write clocks must be assigned to the DQS/DQSn pins associated to this write DQS/DQ group. Do not use the CQ/CQn pin-pair for write clocks.

Using a DQS/DQ group for the write-data signals minimizes output skew, allows access to the write-leveling circuitry (for DDR3 SDRAM interfaces), and allows vertical migration. These pins also have access to deskewing circuitry (using programmable delay chains) that can compensate for delay mismatch between signals on the bus.

The DQS and DQ pin locations are fixed in the pin table. Memory interface circuitry is available in every Stratix IV I/O bank that does not support transceivers. All the memory interface pins support the I/O standards required to support DDR3, DDR2, DDR SDRAM, QDRII+, QDRII SRAM, and RLDRAM II devices.

The Stratix IV device supports DQS and DQ signals with DQ bus modes of ×4, ×8/×9, ×16/×18, or ×32/×36, although not all devices support DQS bus mode ×32/×36. When any of these pins are not used for memory interfacing, you can use them as user I/Os. In addition, you can use any DQSn or CQn pins not used for clocking as DQ (data) pins. Table 7–4 lists pin support per DQS/DQ bus mode, including the DQS/CQ and DQSn/CQn pin pair.

Table 7–4. Stratix IV DQS/DQ Bus Mode Pins

Mode	DQSn Support	CQn Support	Parity or DM (Optional)	QVLD (Optional) *(1)*	Typical Number of Data Pins per Group	Maximum Number of Data Pins per Group *(2)*
×4	Yes	No	No *(6)*	No	4	5
×8/×9 *(3)*	Yes	Yes	Yes	Yes	8 or 9	11
×16/×18 *(4)*	Yes	Yes	Yes	Yes	16 or 18	23
×32/×36 *(5)*	Yes	Yes	Yes	Yes	32 or 36	47

Notes to Table 7–4:

(1) The QVLD pin is not used in the ALTMEMPHY megafunction.

(2) This represents the maximum number of DQ pins (including parity, data mask, and QVLD pins) connected to the DQS bus network with single-ended DQS signaling. When you use differential or complementary DQS signaling, the maximum number of data per group decreases by one. This number may vary per DQS/DQ group in a particular device. Check with the pin table for the accurate number per group. For DDR3, DDR2, and DDR interfaces, the number of pins is further reduced for an interface larger than ×8 due to the need of one DQS pin for each ×8/×9 group that is used to form the x16/x18 and ×32/×36 groups.

(3) Two ×4 DQS/DQ groups are stitched to make a ×8/×9 group, so there are a total of 12 pins in this group.

(4) Four ×4 DQS/DQ groups are stitched to make a ×16/×18 group.

(5) Eight ×4 DQS/DQ groups are stitched to make a ×32/×36 group.

(6) The DM pin can be supported if differential DQS is not used and the group does not have additional signals.

Table 7–5 shows the maximum number of DQS/DQ groups per side of the Stratix IV device. For a more detailed listing of the number of DQS/DQ groups available per bank in each Stratix IV device, see Figure 7–3 through Figure 7–6. These figures represent the die-top view of the Stratix IV device.

Table 7–5. *Number of DQS/DQ Groups in Stratix IV Devices per Side (Part 1 of 3) (Note 1)*

Device	Package	Side	×4 *(2)*	×8/×9	×16/×18	×32/×36 *(3)*
EP4SGX70 EP4SGX110 EP4SGX230	780-pin FineLine BGA	Left	14	6	2	0
		Bottom	17	8	2	0
		Right	0	0	0	0
		Top	17	8	2	0
EP4SGX110	1152-pin FineLine BGA	Left	7	3	1	0
		Bottom	17	8	2	0
		Right	7	3	1	0
		Top	17	8	2	0
EP4SE110 EP4SE230 EP4SE290 EP4SE360	780-pin FineLine BGA	Left	14	6	2	0
		Bottom	17	8	2	0
		Right	14	6	2	0
		Top	17	8	2	0

Table 7–5. ***Number of DQS/DQ Groups in Stratix IV Devices per Side (Part 2 of 3) (Note 1)***

Device	Package	Side	×4 (2)	×8/×9	×16/×18	×32/×36 (3)
EP4SGX230 EP4SGX290 EP4SGX360 EP4SGX530	1152-pin FineLine BGA	Left	13	6	2	0
		Bottom	26	12	4	0
		Right	13	6	2	0
		Top	26	12	4	0
EP4SGX230 EP4SGX290 EP4SGX360 EP4SGX530	1517-pin FineLine BGA	Left	26	12	4	0
		Bottom	26	12	4	0
		Right	26	12	4	0
		Top	26	12	4	0
EP4SGX290 EP4SGX360	780-pin FineLine BGA	Left	0	0	0	0
		Bottom	18	8	4	0
		Right	0	0	0	0
		Top	18	8	4	0
EP4SE290 EP4SE360 EP4SE530 EP4SE680	1152-pin FineLine BGA	Left	26	12	4	0
		Bottom	26	12	4	0
		Right	26	12	4	0
		Top	26	12	4	0
EP4SE290 EP4SE360	1517-pin FineLine BGA	Left	26	12	4	0
		Bottom	38	18	8	4
		Right	26	12	4	0
		Top	38	18	8	4
EP4SGX290 EP4SGX360	1932-pin FineLine BGA	Left	26	12	4	0
		Bottom	38	18	8	4
		Right	26	12	4	0
		Top	38	18	8	4
EP4SE530 EP4SE680	1517-pin FineLine BGA	Left	34	16	6	0
		Bottom	38	18	8	4
		Right	34	16	6	0
		Top	38	18	8	4
EP4SGX530	1932-pin FineLine BGA	Left	29	13	4	0
		Bottom	38	18	8	4
		Right	29	13	4	0
		Top	38	18	8	4
EP4SE530	1760-pin FineLine BGA	Left	34	16	6	0
		Bottom	38	18	8	4
		Right	34	16	6	0
		Top	38	18	8	4

Table 7–5. ***Number of DQS/DQ Groups in Stratix IV Devices per Side (Part 3 of 3) (Note 1)***

Device	Package	Side	×4 *(2)*	×8/×9	×16/×18	×32/×36 *(3)*
EP4SE680	1760-pin FineLine BGA	Left	40	18	6	0
		Bottom	44	22	10	4
		Right	40	18	6	0
		Top	44	22	10	4

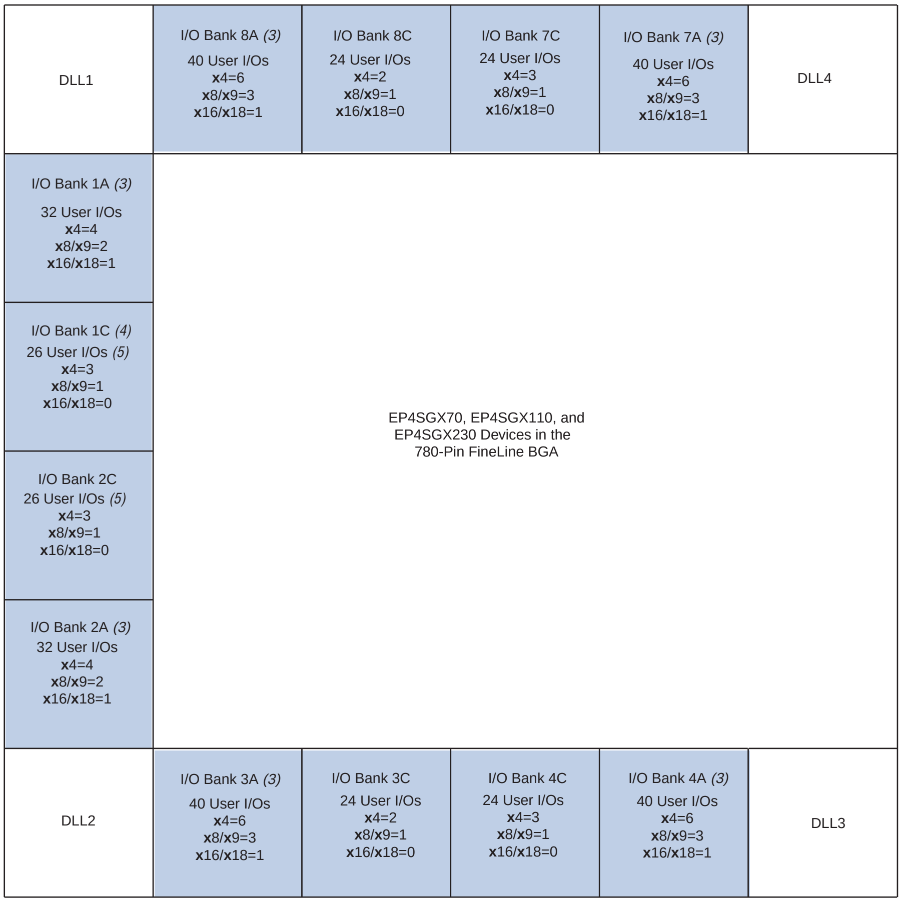
Notes to Table 7–5:

(1) These numbers are preliminary until the devices are available.

(2) Some of the ×4 groups may use R_{UP} and R_{DN} pins. You cannot use these groups if you use the Stratix IV calibrated OCT feature.

(3) To interface with a ×36 QDRII+/QDRII SRAM device in a Stratix IV FPGA that does not support the ×32/×36 DQS/DQ group, refer to "Combining ×16/×18 DQS/DQ groups for ×36 QDRII+/QDRII SRAM Interface" on page 7–25.

Figure 7–3. Number of DQS/DQ Groups per Bank in EP4SGX70, EP4SGX110, and EP4SGX230 Devices in the 780-Pin FineLine BGA Package *(Note 1)*, *(2)*



Notes to Figure 7–3:

(1) These numbers are preliminary until the devices are available.

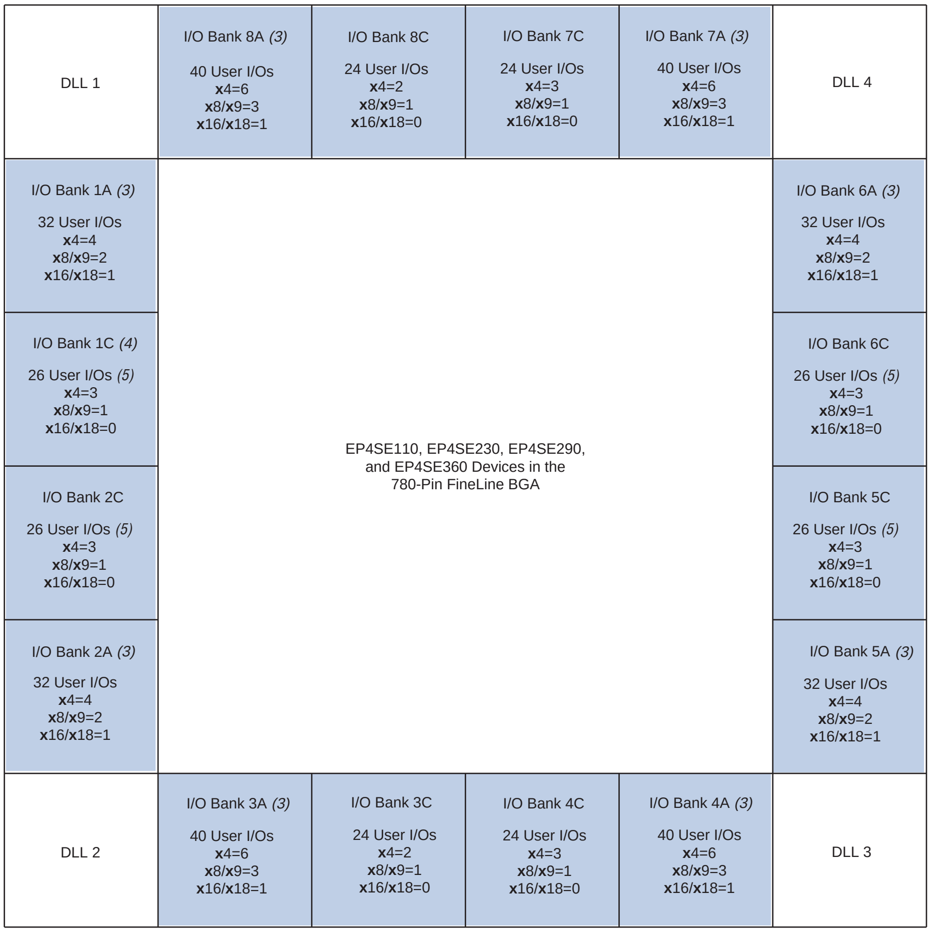
(2) This device does not support ×32/×36 mode. To interface with a ×36 QDRII+/QDRII SRAM device, refer to "Combining ×16/×18 DQS/DQ groups for ×36 QDRII+/QDRII SRAM Interface" on page 7–25.

(3) You can also use DQS/DQSn pins in some of the ×4 groups as R_{UP}/R_{DN} pins. You cannot use a ×4 group for memory interfaces if two pins of the group are being used as R_{UP} and R_{DN} pins for OCT calibration. You can still use the ×16/×18 or ×32/×36 groups that include these ×4 groups. However, there are restrictions on using ×8/×9 groups that include these ×4 groups.

(4) You can also use some of the DQS/DQ pins in I/O Bank 1C as configuration pins. You cannot use a ×4 DQS/DQ group with any of its pin members used for configuration purposes. Make sure that the DQS/DQ groups that you have chosen are not also used for configuration as you may lose up to four ×4 DQS/DQ groups, depending on your configuration scheme.

(5) All I/O pin counts include four dedicated clock inputs (`CLK1p`, `CLK1n`, `CLK3p`, and `CLK3n`) that you can use for data inputs.

Figure 7–4. Number of DQS/DQ Groups per Bank in EP4SE110, EP4SE230, EP4SE290, and EP4SE360 Devices in the 780-Pin FineLine BGA Package *(Note 1)*, *(2)*



DLL 1	I/O Bank 8A *(3)* 40 User I/Os x4=6 x8/x9=3 x16/x18=1	I/O Bank 8C 24 User I/Os x4=2 x8/x9=1 x16/x18=0	I/O Bank 7C 24 User I/Os x4=3 x8/x9=1 x16/x18=0	I/O Bank 7A *(3)* 40 User I/Os x4=6 x8/x9=3 x16/x18=1	DLL 4
I/O Bank 1A *(3)* 32 User I/Os x4=4 x8/x9=2 x16/x18=1					I/O Bank 6A *(3)* 32 User I/Os x4=4 x8/x9=2 x16/x18=1
I/O Bank 1C *(4)* 26 User I/Os *(5)* x4=3 x8/x9=1 x16/x18=0					I/O Bank 6C 26 User I/Os *(5)* x4=3 x8/x9=1 x16/x18=0
		EP4SE110, EP4SE230, EP4SE290, and EP4SE360 Devices in the 780-Pin FineLine BGA			
I/O Bank 2C 26 User I/Os *(5)* x4=3 x8/x9=1 x16/x18=0					I/O Bank 5C 26 User I/Os *(5)* x4=3 x8/x9=1 x16/x18=0
I/O Bank 2A *(3)* 32 User I/Os x4=4 x8/x9=2 x16/x18=1					I/O Bank 5A *(3)* 32 User I/Os x4=4 x8/x9=2 x16/x18=1
DLL 2	I/O Bank 3A *(3)* 40 User I/Os x4=6 x8/x9=3 x16/x18=1	I/O Bank 3C 24 User I/Os x4=2 x8/x9=1 x16/x18=0	I/O Bank 4C 24 User I/Os x4=3 x8/x9=1 x16/x18=0	I/O Bank 4A *(3)* 40 User I/Os x4=6 x8/x9=3 x16/x18=1	DLL 3

Notes to Figure 7–4:

(1) These numbers are preliminary until the devices are available.

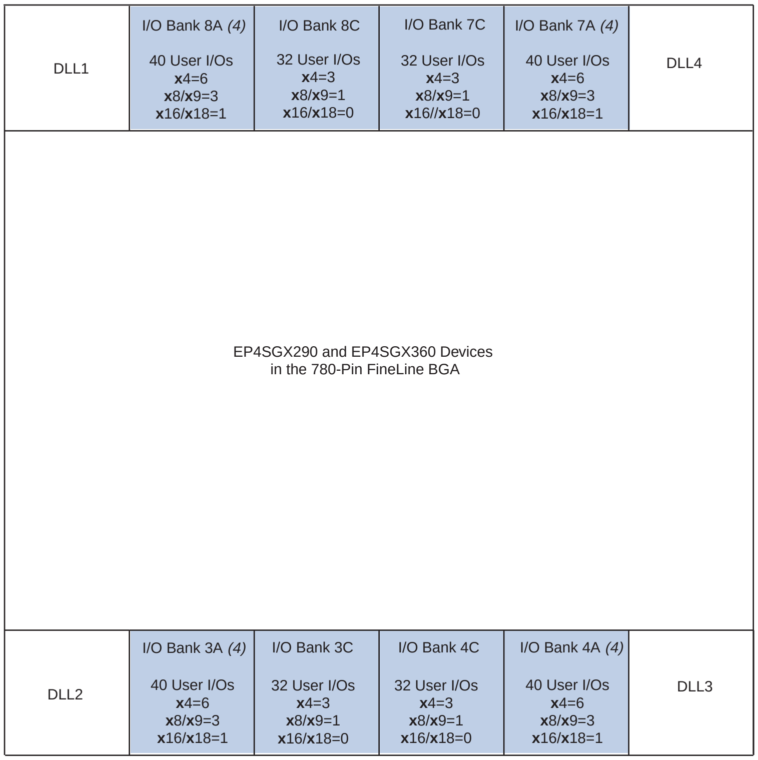
(2) This device does not support ×32/×36 mode. To interface with a ×36 QDRII+/QDRII SRAM device, refer to "Combining ×16/×18 DQS/DQ groups for ×36 QDRII+/QDRII SRAM Interface" on page 7–25.

(3) You can also use DQS/DQSn pins in some of the ×4 groups as R_{UP}/R_{DN} pins. You cannot use a ×4 group for memory interfaces if two pins of the group are being used as R_{UP} and R_{DN} pins for OCT calibration. You can still use the ×16/×18 or ×32/×36 groups that include these ×4 groups. However, there are restrictions on using ×8/×9 groups that include these ×4 groups.

(4) You can also use some of the DQS/DQ pins in I/O Bank 1C as configuration pins. You cannot use a ×4 DQS/DQ group with any of its pin members used for configuration purposes. Make sure that the DQS/DQ groups that you have chosen are not also used for configuration as you may lose up to four ×4 DQS/DQ groups, depending on your configuration scheme.

(5) All I/O pin counts include eight dedicated clock inputs (`CLK1p`, `CLK1n`, `CLK3p`, `CLK3n`, `CLK8p`, `CLK8n`, `CLK10p`, and `CLK10n`) that you can use for data inputs.

Figure 7–5. Number of DQS/DQ Groups per Bank in EP4SGX290 and EP4SGX360 Devices in the 780-Pin FineLine BGA Package *(Note 1)*, *(2)*, *(3)*



Notes to Figure 7–5:

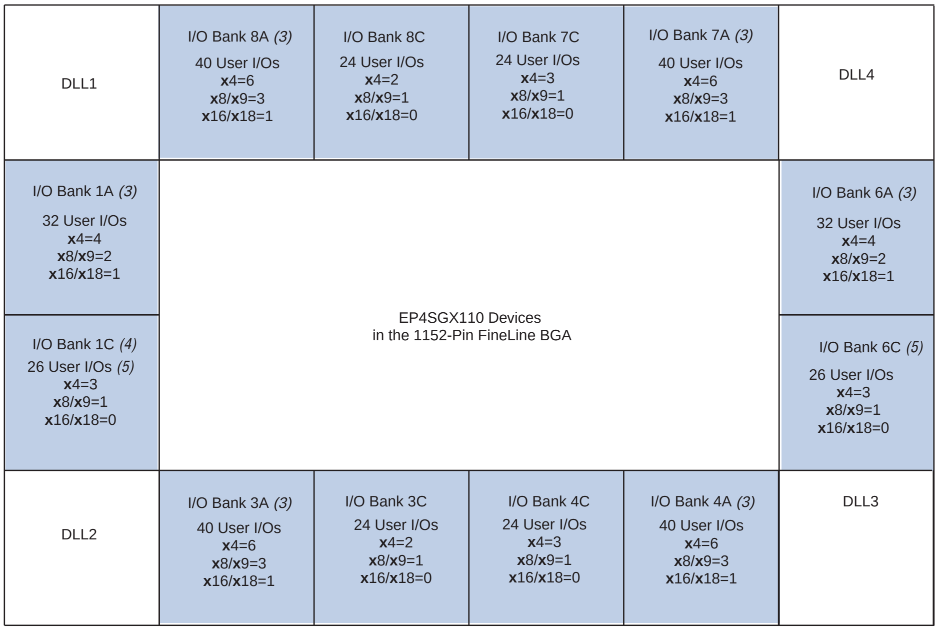
(1) These numbers are preliminary until the devices are available.

(2) This device does not support ×32/×36 mode. To interface with a ×36 QDRII+/QDRII SRAM device, refer to "Combining ×16/×18 DQS/DQ groups for ×36 QDRII+/QDRII SRAM Interface" on page 7–25.

(3) There are 13 pins available in I/O bank 1C that you can use only for configuration purposes. These pins are not available as user I/Os, even if you do not use them for configuration.

(4) You can also use some of the DQS/DQ pins in I/O Bank 1C as configuration pins.You cannot use a ×4 DQS/DQ group with any of its pin members used for configuration purposes. Make sure that the DQS/DQ groups that you have chosen are not also used for configuration as you may lose up to four ×4 DQS/DQ groups, depending on your configuration scheme.

Figure 7–6. Number of DQS/DQ Groups per Bank in EP4SGX110 Devices in the 1152-Pin FineLine BGA Package *(Note 1)*,*(2)*



Notes to Figure 7–6:

(1) These numbers are preliminary until the devices are available.

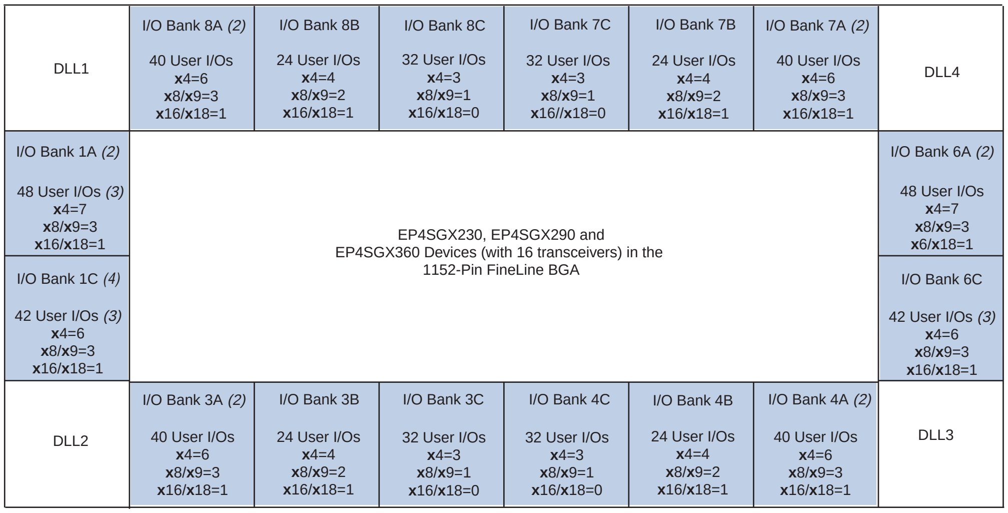
(2) This device does not support ×32/×36 mode. To interface with a ×36 QDRII+/QDRII SRAM device, refer to "Combining ×16/×18 DQS/DQ groups for ×36 QDRII+/QDRII SRAM Interface" on page 7–25.

(3) You can also use DQS/DQSn pins in some of the ×4 groups as R_{UP}/R_{DN} pins. You cannot use a ×4 group for memory interfaces if two pins of the group are being used as R_{UP} and R_{DN} pins for OCT calibration. You can still use the ×16/×18 or ×32/×36 groups that include these ×4 groups. However, there are restrictions on using ×8/×9 groups that include these ×4 groups.

(4) You can also use some of the DQS/DQ pins in I/O Bank 1C as configuration pins. You cannot use a ×4 DQS/DQ group with any of its pin members used for configuration purposes. Make sure that the DQS/DQ groups that you have chosen are not also used for configuration as you may lose up to four ×4 DQS/DQ groups, depending on your configuration scheme.

(5) All I/O pin counts include four dedicated clock inputs (`CLK1p`, `CLK1n`, `CLK10p`, and `CLK10n`) that you can use for data inputs.

Figure 7–7. Number of DQS/DQ Groups per Bank in EP4SGX230, EP4SGX290, and EP4SGX360 Devices with Sixteen Transceivers in the 1152-Pin FineLine BGA Package *(Note 1)*,*(2)*



DLL1	I/O Bank 8A *(2)* 40 User I/Os x4=6 x8/x9=3 x16/x18=1	I/O Bank 8B 24 User I/Os x4=4 x8/x9=2 x16/x18=1	I/O Bank 8C 32 User I/Os x4=3 x8/x9=1 x16/x18=0	I/O Bank 7C 32 User I/Os x4=3 x8/x9=1 x16//x18=0	I/O Bank 7B 24 User I/Os x4=4 x8/x9=2 x16/x18=1	I/O Bank 7A *(2)* 40 User I/Os x4=6 x8/x9=3 x16/x18=1	DLL4
I/O Bank 1A *(2)* 48 User I/Os *(3)* x4=7 x8/x9=3 x16/x18=1	EP4SGX230, EP4SGX290 and EP4SGX360 Devices (with 16 transceivers) in the 1152-Pin FineLine BGA						I/O Bank 6A *(2)* 48 User I/Os x4=7 x8/x9=3 x6/x18=1
I/O Bank 1C *(4)* 42 User I/Os *(3)* x4=6 x8/x9=3 x16/x18=1							I/O Bank 6C 42 User I/Os *(3)* x4=6 x8/x9=3 x16/x18=1
DLL2	I/O Bank 3A *(2)* 40 User I/Os x4=6 x8/x9=3 x16/x18=1	I/O Bank 3B 24 User I/Os x4=4 x8/x9=2 x16/x18=1	I/O Bank 3C 32 User I/Os x4=3 x8/x9=1 x16/x18=0	I/O Bank 4C 32 User I/Os x4=3 x8/x9=1 x16/x18=0	I/O Bank 4B 24 User I/Os x4=4 x8/x9=2 x16/x18=1	I/O Bank 4A *(2)* 40 User I/Os x4=6 x8/x9=3 x16/x18=1	DLL3

Notes to Figure 7–7:

(1) These numbers are preliminary until the devices are available.

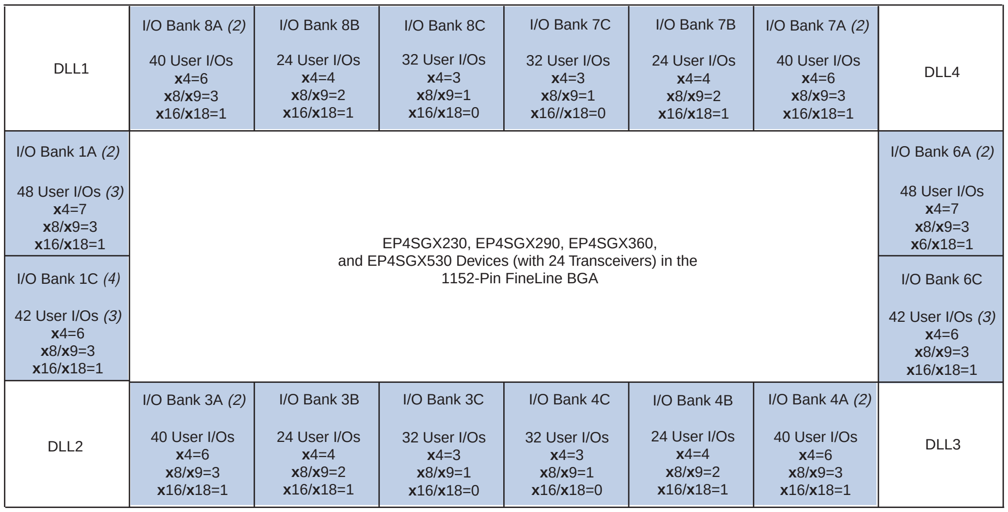
(2) This device does not support ×32/×36 mode. To interface with a ×36 QDRII+/QDRII SRAM device, refer to "Combining ×16/×18 DQS/DQ groups for ×36 QDRII+/QDRII SRAM Interface" on page 7–25.

(3) You can also use DQS/DQSn pins in some of the ×4 groups as R_{UP}/R_{DN} pins. You cannot use a ×4 group for memory interfaces if two pins of the group are being used as R_{UP} and R_{DN} pins for OCT calibration. You can still use the ×16/×18 or ×32/×36 groups that include these ×4 groups. However, there are restrictions on using ×8/×9 groups that include these ×4 groups.

(4) All I/O pin counts include four dedicated clock inputs (`CLK1p`, `CLK1n`, `CLK10p`, and `CLK10n`) that you can use for data inputs.

(5) You can also use some of the DQS/DQ pins in I/O Bank 1C as configuration pins. You cannot use a ×4 DQS/DQ group with any of its pin members used for configuration purposes. Make sure that the DQS/DQ groups that you have chosen are not also used for configuration as you may lose up to four ×4 DQS/DQ groups, depending on your configuration scheme.

Figure 7–8. Number of DQS/DQ Groups per Bank in EP4SGX230, EP4SGX290, EP4SGX360, and EP4SGX530 Devices with Twenty-Four Transceivers in the 1152-Pin FineLine BGA Package *(Note 1)*,*(2)*



Notes to Figure 7–8:

(1) These numbers are preliminary until the devices are available.

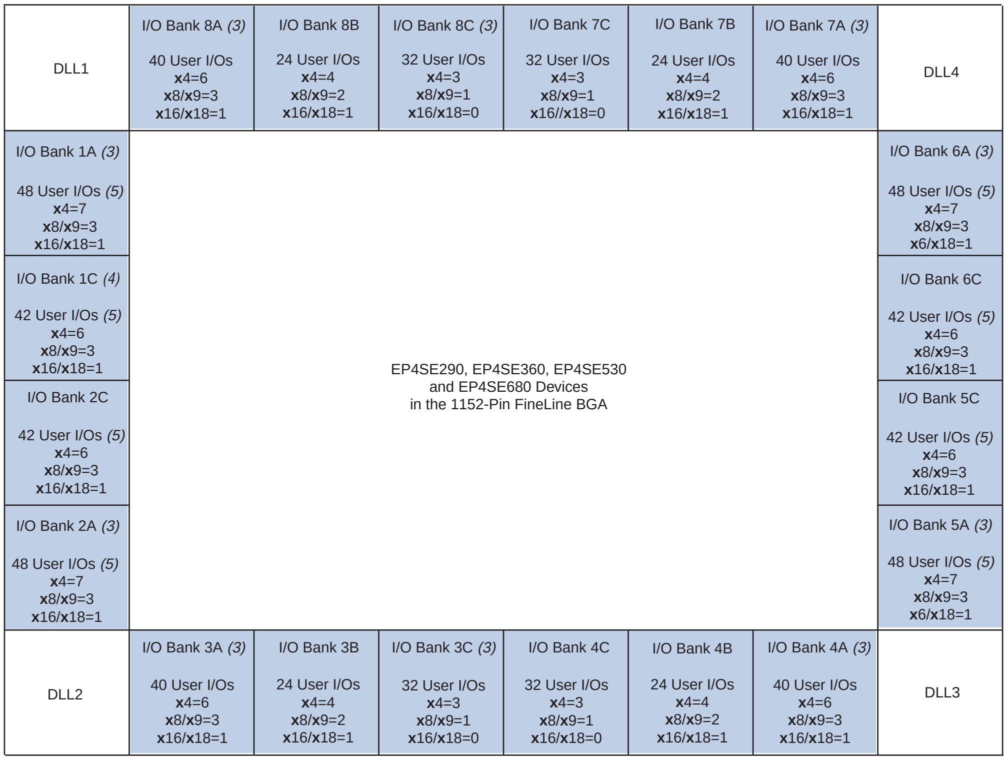
(2) This device does not support ×32/×36 mode. To interface with a ×36 QDRII+/QDRII SRAM device, refer to "Combining ×16/×18 DQS/DQ groups for ×36 QDRII+/QDRII SRAM Interface" on page 7–25.

(3) You can also use DQS/DQSn pins in some of the ×4 groups as R_{UP}/R_{DN} pins. You cannot use a ×4 group for memory interfaces if two pins of the group are being used as R_{UP} and R_{DN} pins for OCT calibration. You can still use the ×16/×18 or ×32/×36 groups that include these ×4 groups. However, there are restrictions on using ×8/×9 groups that include these ×4 groups.

(4) All I/O pin counts include four dedicated clock inputs (`CLK1p`, `CLK1n`, `CLK10p`, and `CLK10n`) that you can use for data inputs.

(5) You can also use some of the DQS/DQ pins in I/O Bank 1C as configuration pins. You cannot use a 4× DQS/DQ group with any of its pin members used for configuration purposes. Make sure that the DQS/DQ groups that you have chosen are not also used for configuration as you may lose up to four ×4 DQS/DQ groups, depending on your configuration scheme.

Figure 7–9. Number of DQS/DQ Groups per Bank in EP4SE290, EP4SE360, EP4SE530, and EPSE680 Devices in the 1152-Pin FineLine BGA Package *(Note 1)*,*(2)*



DLL1	I/O Bank 8A *(3)* 40 User I/Os x4=6 x8/x9=3 x16/x18=1	I/O Bank 8B 24 User I/Os x4=4 x8/x9=2 x16/x18=1	I/O Bank 8C *(3)* 32 User I/Os x4=3 x8/x9=1 x16/x18=0	I/O Bank 7C 32 User I/Os x4=3 x8/x9=1 x16//x18=0	I/O Bank 7B 24 User I/Os x4=4 x8/x9=2 x16/x18=1	I/O Bank 7A *(3)* 40 User I/Os x4=6 x8/x9=3 x16/x18=1	DLL4
I/O Bank 1A *(3)* 48 User I/Os *(5)* x4=7 x8/x9=3 x16/x18=1							I/O Bank 6A *(3)* 48 User I/Os *(5)* x4=7 x8/x9=3 x6/x18=1
I/O Bank 1C *(4)* 42 User I/Os *(5)* x4=6 x8/x9=3 x16/x18=1			EP4SE290, EP4SE360, EP4SE530 and EP4SE680 Devices in the 1152-Pin FineLine BGA				I/O Bank 6C 42 User I/Os *(5)* x4=6 x8/x9=3 x16/x18=1
I/O Bank 2C 42 User I/Os *(5)* x4=6 x8/x9=3 x16/x18=1							I/O Bank 5C 42 User I/Os *(5)* x4=6 x8/x9=3 x16/x18=1
I/O Bank 2A *(3)* 48 User I/Os *(5)* x4=7 x8/x9=3 x16/x18=1							I/O Bank 5A *(3)* 48 User I/Os *(5)* x4=7 x8/x9=3 x6/x18=1
DLL2	I/O Bank 3A *(3)* 40 User I/Os x4=6 x8/x9=3 x16/x18=1	I/O Bank 3B 24 User I/Os x4=4 x8/x9=2 x16/x18=1	I/O Bank 3C *(3)* 32 User I/Os x4=3 x8/x9=1 x16/x18=0	I/O Bank 4C 32 User I/Os x4=3 x8/x9=1 x16/x18=0	I/O Bank 4B 24 User I/Os x4=4 x8/x9=2 x16/x18=1	I/O Bank 4A *(3)* 40 User I/Os x4=6 x8/x9=3 x16/x18=1	DLL3

Notes to Figure 7–9:

(1) These numbers are preliminary until the devices are available.

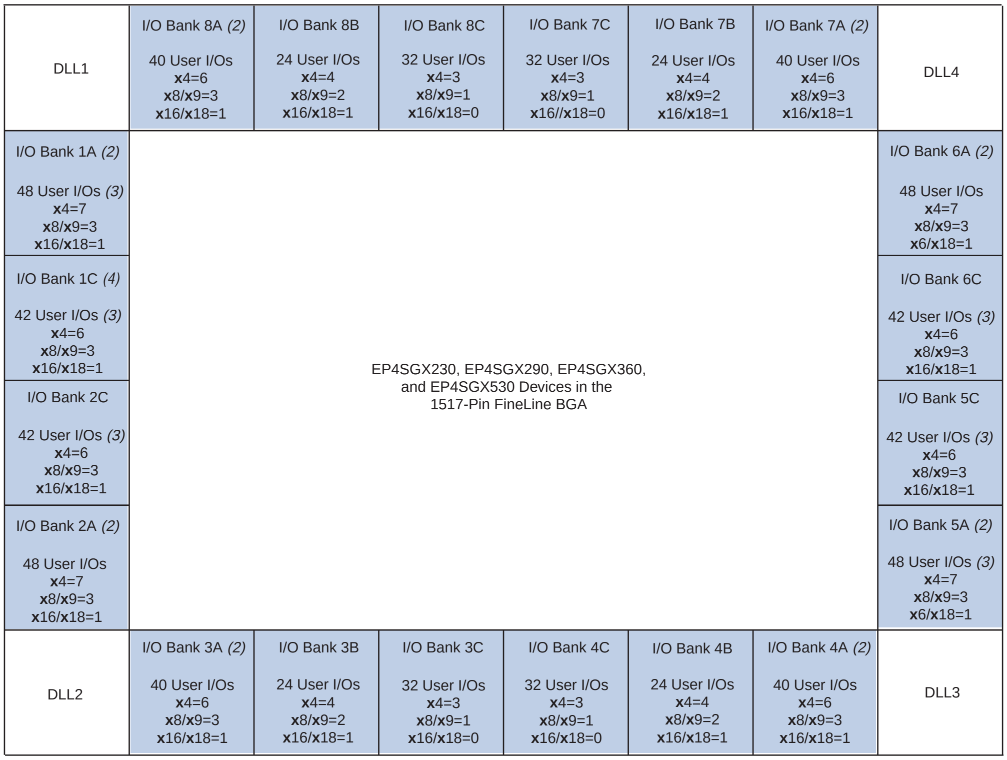
(2) This device does not support ×32/×36 mode. To interface with a ×36 QDRII+/QDRII SRAM device, refer to "Combining ×16/×18 DQS/DQ groups for ×36 QDRII+/QDRII SRAM Interface" on page 7–25.

(3) You can also use DQS/DQSn pins in some of the ×4 groups as R_{UP}/R_{DN} pins. You cannot use a ×4 group for memory interfaces if two pins of the group are being used as R_{UP} and R_{DN} pins for OCT calibration. You can still use the ×16/×18 or ×32/×36 groups that include these ×4 groups. However, there are restrictions on using ×8/×9 groups that include these ×4 groups. Only EP4SE530 has OCT calibration blocks in I/O banks 3C and 8C.

(4) You can also use some of the DQS/DQ pins in I/O Bank 1C as configuration pins. You cannot use a ×4 DQS/DQ group with any of its pin members used for configuration purposes. Make sure that the DQS/DQ groups that you have chosen are not also used for configuration as you may lose up to four ×4 DQS/DQ groups, depending on your configuration scheme.

(5) All I/O pin counts include eight dedicated clock inputs (`CLK1p`, `CLK1n`, `CLK3p`, `CLK3n`, `CLK8p`, `CLK8n`, `CLK10p`, and `CLK10n`).

Figure 7–10. Number of DQS/DQ Groups per Bank in EP4SGX230, EP4SGX290, EP4SGX360, and EP4SGX530 Devices in the 1517-Pin FineLine BGA Package *(Note 1)*,*(2)*



DLL1	I/O Bank 8A *(2)* 40 User I/Os x4=6 x8/x9=3 x16/x18=1	I/O Bank 8B 24 User I/Os x4=4 x8/x9=2 x16/x18=1	I/O Bank 8C 32 User I/Os x4=3 x8/x9=1 x16/x18=0	I/O Bank 7C 32 User I/Os x4=3 x8/x9=1 x16//x18=0	I/O Bank 7B 24 User I/Os x4=4 x8/x9=2 x16/x18=1	I/O Bank 7A *(2)* 40 User I/Os x4=6 x8/x9=3 x16/x18=1	DLL4
I/O Bank 1A *(2)* 48 User I/Os *(3)* x4=7 x8/x9=3 x16/x18=1							I/O Bank 6A *(2)* 48 User I/Os x4=7 x8/x9=3 x6/x18=1
I/O Bank 1C *(4)* 42 User I/Os *(3)* x4=6 x8/x9=3 x16/x18=1			EP4SGX230, EP4SGX290, EP4SGX360, and EP4SGX530 Devices in the 1517-Pin FineLine BGA				I/O Bank 6C 42 User I/Os *(3)* x4=6 x8/x9=3 x16/x18=1
I/O Bank 2C 42 User I/Os *(3)* x4=6 x8/x9=3 x16/x18=1							I/O Bank 5C 42 User I/Os *(3)* x4=6 x8/x9=3 x16/x18=1
I/O Bank 2A *(2)* 48 User I/Os x4=7 x8/x9=3 x16/x18=1							I/O Bank 5A *(2)* 48 User I/Os *(3)* x4=7 x8/x9=3 x6/x18=1
DLL2	I/O Bank 3A *(2)* 40 User I/Os x4=6 x8/x9=3 x16/x18=1	I/O Bank 3B 24 User I/Os x4=4 x8/x9=2 x16/x18=1	I/O Bank 3C 32 User I/Os x4=3 x8/x9=1 x16/x18=0	I/O Bank 4C 32 User I/Os x4=3 x8/x9=1 x16/x18=0	I/O Bank 4B 24 User I/Os x4=4 x8/x9=2 x16/x18=1	I/O Bank 4A *(2)* 40 User I/Os x4=6 x8/x9=3 x16/x18=1	DLL3

Notes to Figure 7–10:

(1) These numbers are preliminary until the devices are available.

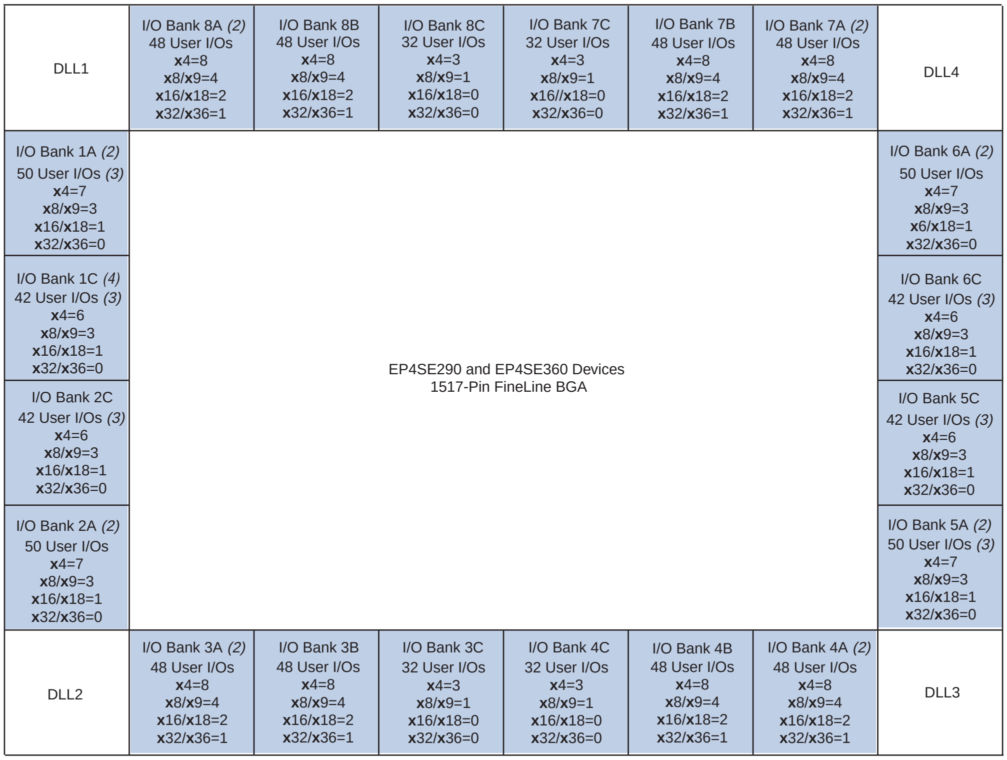
(2) This device does not support ×32/×36 mode. To interface with a ×36 QDRII+/QDRII SRAM device, refer to “Combining ×16/×18 DQS/DQ groups for ×36 QDRII+/QDRII SRAM Interface” on page 7–25.

(3) You can also use DQS/DQSn pins in some of the ×4 groups as R_{UP}/R_{DN} pins. You cannot use a ×4 group for memory interfaces if two pins of the group are being used as R_{UP} and R_{DN} pins for OCT calibration. You can still use the ×16/×18 or ×32/×36 groups that include these ×4 groups. However, there are restrictions on using ×8/×9 groups that include these ×4 groups. Only EP4SGX530 has OCT calibration blocks in I/O banks 3C and 8C.

(4) All I/O pin counts include eight dedicated clock inputs (`CLK1p`, `CLK1n`, `CLK3p`, `CLK3n`, `CLK8p`, `CLK8n`, `CLK10p`, and `CLK10n`) that you can use for data inputs.

(5) You can also use some of the DQS/DQ pins in I/O Bank 1C as configuration pins. You cannot use a ×4 DQS/DQ group with any of its pin members used for configuration purposes. Make sure that the DQS/DQ groups that you have chosen are not also used for configuration as you may lose up to four ×4 DQS/DQ groups, depending on your configuration scheme.

Figure 7–11. Number of DQS/DQ Groups per Bank in EP4SE290 and EP4SE360 Devices in the 1517-Pin FineLine BGA Package *(Note 1)*



Notes to Figure 7–11:

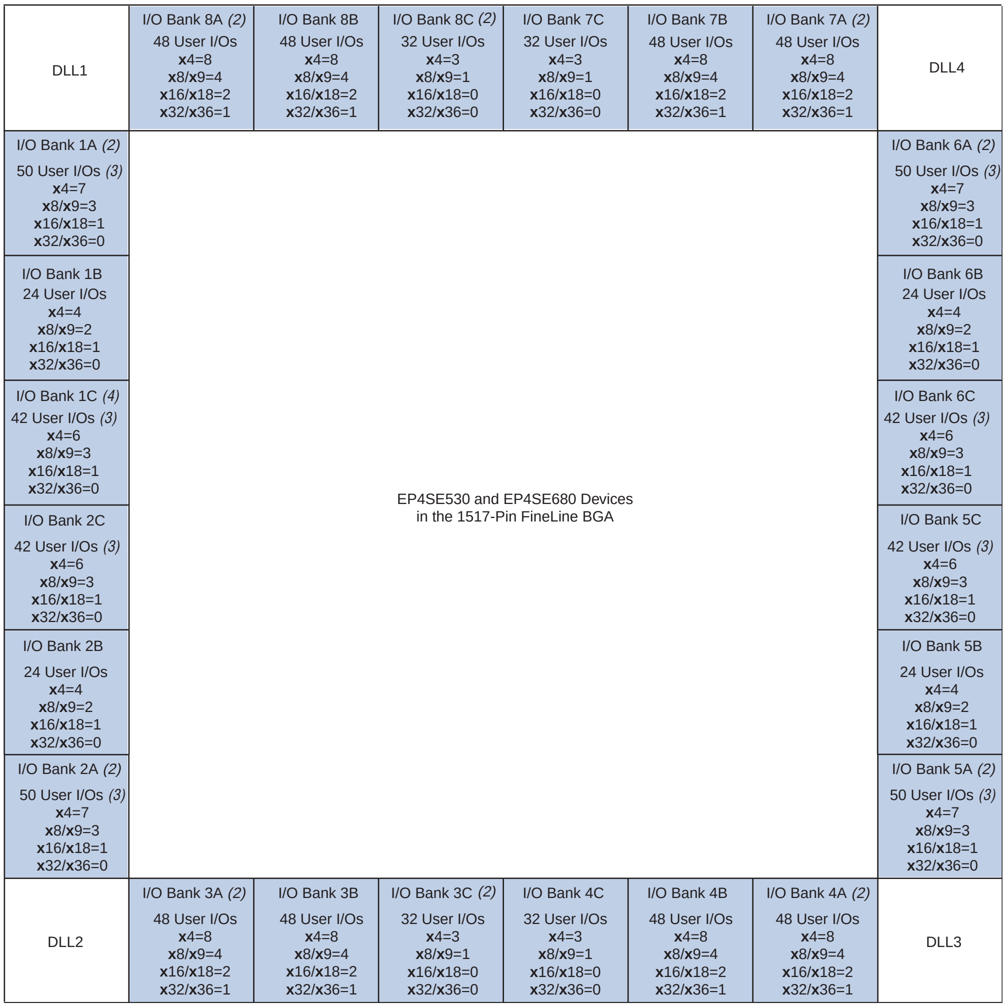
(1) These numbers are preliminary until the devices are available.

(2) You can also use DQS/DQSn pins in some of the ×4 groups as R_{UP}/R_{DN} pins. You cannot use a ×4 group for memory interfaces if two pins of the group are being used as R_{UP} and R_{DN} pins for OCT calibration. You can still use the ×16/×18 or ×32/×36 groups that include these ×4 groups. However, there are restrictions on using ×8/×9 groups that include these ×4 groups.

(3) All I/O pin counts include eight dedicated clock inputs (CLK1p, CLK1n, CLK3p, CLK3n, CLK8p, CLK8n, CLK10p, and CLK10n) and eight dedicated corner PLL clock inputs (PLL_L1_CLKp, PLL_L1_CLKn, PLL_L4_CLKp, PLL_L4_CLKn, PLL_R4_CLKp, PLL_R4_CLKn, PLL_R1_CLKp, and PLL_R1_CLKn) that you can use for data inputs.

(4) You can also use some of the DQS/DQ pins in I/O Bank 1C as configuration pins. You cannot use a ×4 DQS/DQ group with any of its pin members used for configuration purposes. Make sure that the DQS/DQ groups that you have chosen are not also used for configuration as you may lose up to four ×4 DQS/DQ groups, depending on your configuration scheme.

Figure 7–12. Number of DQS/DQ Groups per Bank in EP4SE530 and EP4SE680 Devices in the 1517-pin FineLine BGA Package *(Note 1)*



DLL1

I/O Bank 8A *(2)*
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 8B
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 8C *(2)*
32 User I/Os
x4=3
x8/x9=1
x16/x18=0
x32/x36=0

I/O Bank 7C
32 User I/Os
x4=3
x8/x9=1
x16/x18=0
x32/x36=0

I/O Bank 7B
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 7A *(2)*
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

DLL4

I/O Bank 1A *(2)*
50 User I/Os *(3)*
x4=7
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 6A *(2)*
50 User I/Os *(3)*
x4=7
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 1B
24 User I/Os
x4=4
x8/x9=2
x16/x18=1
x32/x36=0

I/O Bank 6B
24 User I/Os
x4=4
x8/x9=2
x16/x18=1
x32/x36=0

I/O Bank 1C *(4)*
42 User I/Os *(3)*
x4=6
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 6C
42 User I/Os *(3)*
x4=6
x8/x9=3
x16/x18=1
x32/x36=0

EP4SE530 and EP4SE680 Devices
in the 1517-Pin FineLine BGA

I/O Bank 2C
42 User I/Os *(3)*
x4=6
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 5C
42 User I/Os *(3)*
x4=6
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 2B
24 User I/Os
x4=4
x8/x9=2
x16/x18=1
x32/x36=0

I/O Bank 5B
24 User I/Os
x4=4
x8/x9=2
x16/x18=1
x32/x36=0

I/O Bank 2A *(2)*
50 User I/Os *(3)*
x4=7
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 5A *(2)*
50 User I/Os *(3)*
x4=7
x8/x9=3
x16/x18=1
x32/x36=0

DLL2

I/O Bank 3A *(2)*
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 3B
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 3C *(2)*
32 User I/Os
x4=3
x8/x9=1
x16/x18=0
x32/x36=0

I/O Bank 4C
32 User I/Os
x4=3
x8/x9=1
x16/x18=0
x32/x36=0

I/O Bank 4B
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 4A *(2)*
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

DLL3

Notes to Figure 7–12:

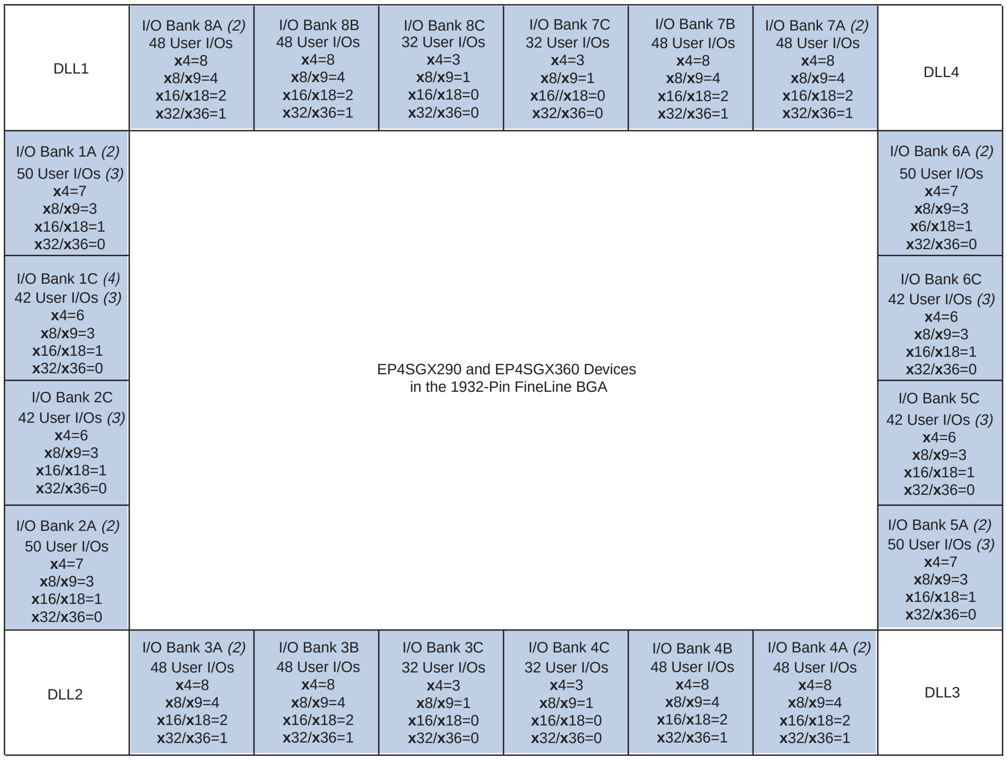
(1) These numbers are preliminary until the devices are available.

(2) You can also use DQS/DQSn pins in some of the ×4 groups as R_{UP}/R_{DN} pins. You cannot use a ×4 group for memory interfaces if two pins of the group are being used as R_{UP} and R_{DN} pins for OCT calibration. You can still use the ×16/×18 or ×32/×36 groups that include these ×4 groups. However, there are restrictions on using ×8/×9 groups that include these ×4 groups.

(3) All I/O pin counts include eight dedicated clock inputs (`CLK1p`, `CLK1n`, `CLK3p`, `CLK3n`, `CLK8p`, `CLK8n`, `CLK10p`, and `CLK10n`) and eight dedicated corner PLL clock inputs (`PLL_L1_CLKp`, `PLL_L1_CLKn`, `PLL_L4_CLKp`, `PLL_L4_CLKn`, `PLL_R4_CLKp`, `PLL_R4_CLKn`, `PLL_R1_CLKp`, and `PLL_R1_CLKn`) that you can use for data inputs.

(4) You can also use some of the DQS/DQ pins in I/O Bank 1C as configuration pins. You cannot use a ×4 DQS/DQ group with any of its pin members used for configuration purposes. Make sure that the DQS/DQ groups that you have chosen are not also used for configuration as you may lose up to four ×4 DQS/DQ groups, depending on your configuration scheme.

Figure 7–13. Number of DQS/DQ Groups per Bank in EP4SGX290 and EP4SGX360 Devices in the 1932-Pin FineLine BGA Package *(Note 1)*


DLL1
I/O Bank 8A (2) 48 User I/Os x4=8 x8/x9=4 x16/x18=2 x32/x36=1
I/O Bank 8B 48 User I/Os x4=8 x8/x9=4 x16/x18=2 x32/x36=1
I/O Bank 8C 32 User I/Os x4=3 x8/x9=1 x16/x18=0 x32/x36=0
I/O Bank 7C 32 User I/Os x4=3 x8/x9=1 x16//x18=0 x32/x36=0
I/O Bank 7B 48 User I/Os x4=8 x8/x9=4 x16/x18=2 x32/x36=1
I/O Bank 7A (2) 48 User I/Os x4=8 x8/x9=4 x16/x18=2 x32/x36=1
DLL4
I/O Bank 1A (2) 50 User I/Os (3) x4=7 x8/x9=3 x16/x18=1 x32/x36=0
I/O Bank 6A (2) 50 User I/Os x4=7 x8/x9=3 x6/x18=1 x32/x36=0
I/O Bank 1C (4) 42 User I/Os (3) x4=6 x8/x9=3 x16/x18=1 x32/x36=0
I/O Bank 6C 42 User I/Os (3) x4=6 x8/x9=3 x16/x18=1 x32/x36=0
EP4SGX290 and EP4SGX360 Devices in the 1932-Pin FineLine BGA
I/O Bank 2C 42 User I/Os (3) x4=6 x8/x9=3 x16/x18=1 x32/x36=0
I/O Bank 5C 42 User I/Os (3) x4=6 x8/x9=3 x16/x18=1 x32/x36=0
I/O Bank 2A (2) 50 User I/Os x4=7 x8/x9=3 x16/x18=1 x32/x36=0
I/O Bank 5A (2) 50 User I/Os (3) x4=7 x8/x9=3 x16/x18=1 x32/x36=0
DLL2
I/O Bank 3A (2) 48 User I/Os x4=8 x8/x9=4 x16/x18=2 x32/x36=1
I/O Bank 3B 48 User I/Os x4=8 x8/x9=4 x16/x18=2 x32/x36=1
I/O Bank 3C 32 User I/Os x4=3 x8/x9=1 x16/x18=0 x32/x36=0
I/O Bank 4C 32 User I/Os x4=3 x8/x9=1 x16/x18=0 x32/x36=0
I/O Bank 4B 48 User I/Os x4=8 x8/x9=4 x16/x18=2 x32/x36=1
I/O Bank 4A (2) 48 User I/Os x4=8 x8/x9=4 x16/x18=2 x32/x36=1
DLL3

Notes to Figure 7–13:

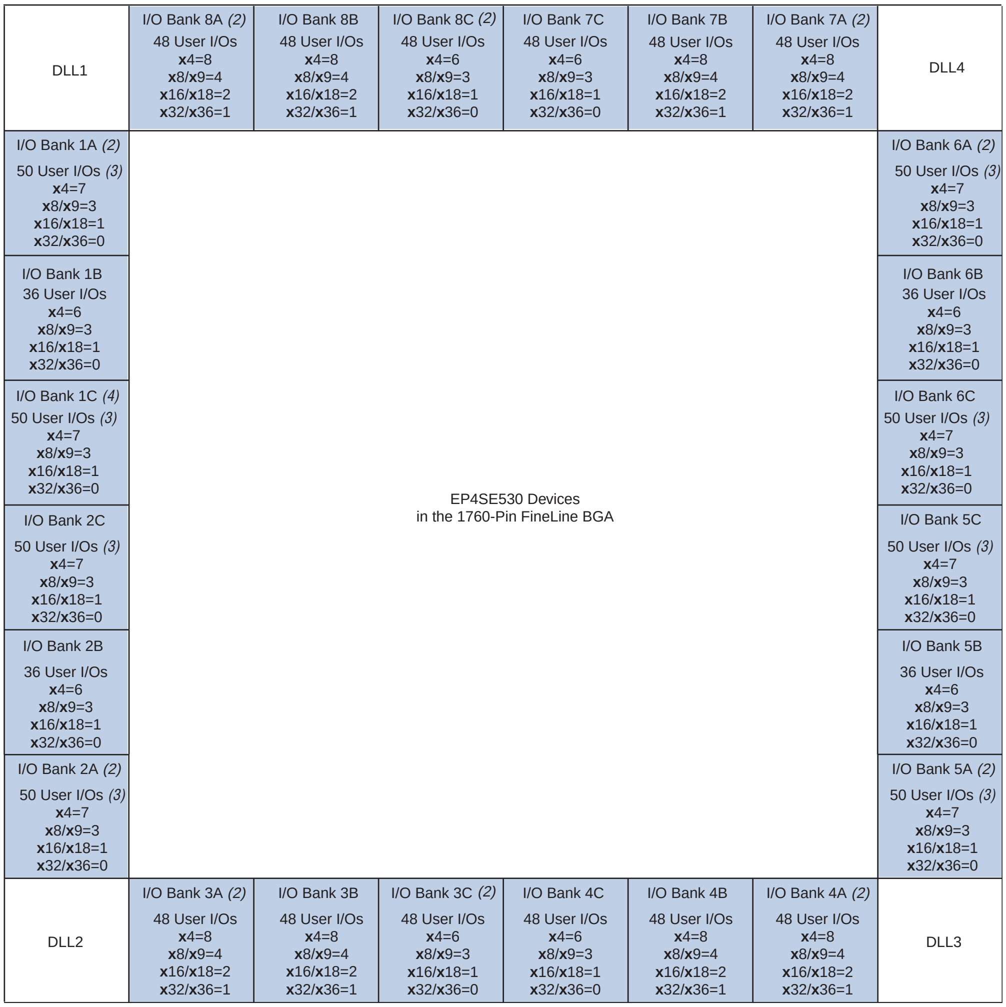
(1) These numbers are preliminary until the devices are available.

(2) You can also use DQS/DQSn pins in some of the ×4 groups as R_{UP}/R_{DN} pins. You cannot use a ×4 group for memory interfaces if two pins of the group are being used as R_{UP} and R_{DN} pins for OCT calibration. You can still use the ×16/×18 or ×32/×36 groups that include these ×4 groups. However, there are restrictions on using ×8/×9 groups that include these ×4 groups.

(3) All I/O pin counts include eight dedicated clock inputs (CLK1p, CLK1n, CLK3p, CLK3n, CLK8p, CLK8n, CLK10p, and CLK10n) and eight dedicated corner PLL clock inputs (PLL_L1_CLKp, PLL_L1_CLKn, PLL_L4_CLKp, PLL_L4_CLKn, PLL_R4_CLKp, PLL_R4_CLKn, PLL_R1_CLKp, and PLL_R1_CLKn) that you can use for data inputs.

(4) You can also use some of the DQS/DQ pins in I/O Bank 1C as configuration pins. You cannot use a ×4 DQS/DQ group with any of its pin members used for configuration purposes. Make sure that the DQS/DQ groups that you have chosen are not also used for configuration as you may lose up to four ×4 DQS/DQ groups, depending on your configuration scheme.

Figure 7–14. Number of DQS/DQ Groups per Bank in EP4SE530 Devices in the 1760-Pin FineLine BGA Package *(Note 1)*



DLL1	I/O Bank 8A *(2)* 48 User I/Os x4=8 x8/x9=4 x16/x18=2 x32/x36=1	I/O Bank 8B 48 User I/Os x4=8 x8/x9=4 x16/x18=2 x32/x36=1	I/O Bank 8C *(2)* 48 User I/Os x4=6 x8/x9=3 x16/x18=1 x32/x36=0	I/O Bank 7C 48 User I/Os x4=6 x8/x9=3 x16/x18=1 x32/x36=0	I/O Bank 7B 48 User I/Os x4=8 x8/x9=4 x16/x18=2 x32/x36=1	I/O Bank 7A *(2)* 48 User I/Os x4=8 x8/x9=4 x16/x18=2 x32/x36=1	DLL4
I/O Bank 1A *(2)* 50 User I/Os *(3)* x4=7 x8/x9=3 x16/x18=1 x32/x36=0							I/O Bank 6A *(2)* 50 User I/Os *(3)* x4=7 x8/x9=3 x16/x18=1 x32/x36=0
I/O Bank 1B 36 User I/Os x4=6 x8/x9=3 x16/x18=1 x32/x36=0							I/O Bank 6B 36 User I/Os x4=6 x8/x9=3 x16/x18=1 x32/x36=0
I/O Bank 1C *(4)* 50 User I/Os *(3)* x4=7 x8/x9=3 x16/x18=1 x32/x36=0			EP4SE530 Devices in the 1760-Pin FineLine BGA				I/O Bank 6C 50 User I/Os *(3)* x4=7 x8/x9=3 x16/x18=1 x32/x36=0
I/O Bank 2C 50 User I/Os *(3)* x4=7 x8/x9=3 x16/x18=1 x32/x36=0							I/O Bank 5C 50 User I/Os *(3)* x4=7 x8/x9=3 x16/x18=1 x32/x36=0
I/O Bank 2B 36 User I/Os x4=6 x8/x9=3 x16/x18=1 x32/x36=0							I/O Bank 5B 36 User I/Os x4=6 x8/x9=3 x16/x18=1 x32/x36=0
I/O Bank 2A *(2)* 50 User I/Os *(3)* x4=7 x8/x9=3 x16/x18=1 x32/x36=0							I/O Bank 5A *(2)* 50 User I/Os *(3)* x4=7 x8/x9=3 x16/x18=1 x32/x36=0
DLL2	I/O Bank 3A *(2)* 48 User I/Os x4=8 x8/x9=4 x16/x18=2 x32/x36=1	I/O Bank 3B 48 User I/Os x4=8 x8/x9=4 x16/x18=2 x32/x36=1	I/O Bank 3C *(2)* 48 User I/Os x4=6 x8/x9=3 x16/x18=1 x32/x36=0	I/O Bank 4C 48 User I/Os x4=6 x8/x9=3 x16/x18=1 x32/x36=0	I/O Bank 4B 48 User I/Os x4=8 x8/x9=4 x16/x18=2 x32/x36=1	I/O Bank 4A *(2)* 48 User I/Os x4=8 x8/x9=4 x16/x18=2 x32/x36=1	DLL3

Notes to Figure 7–14:

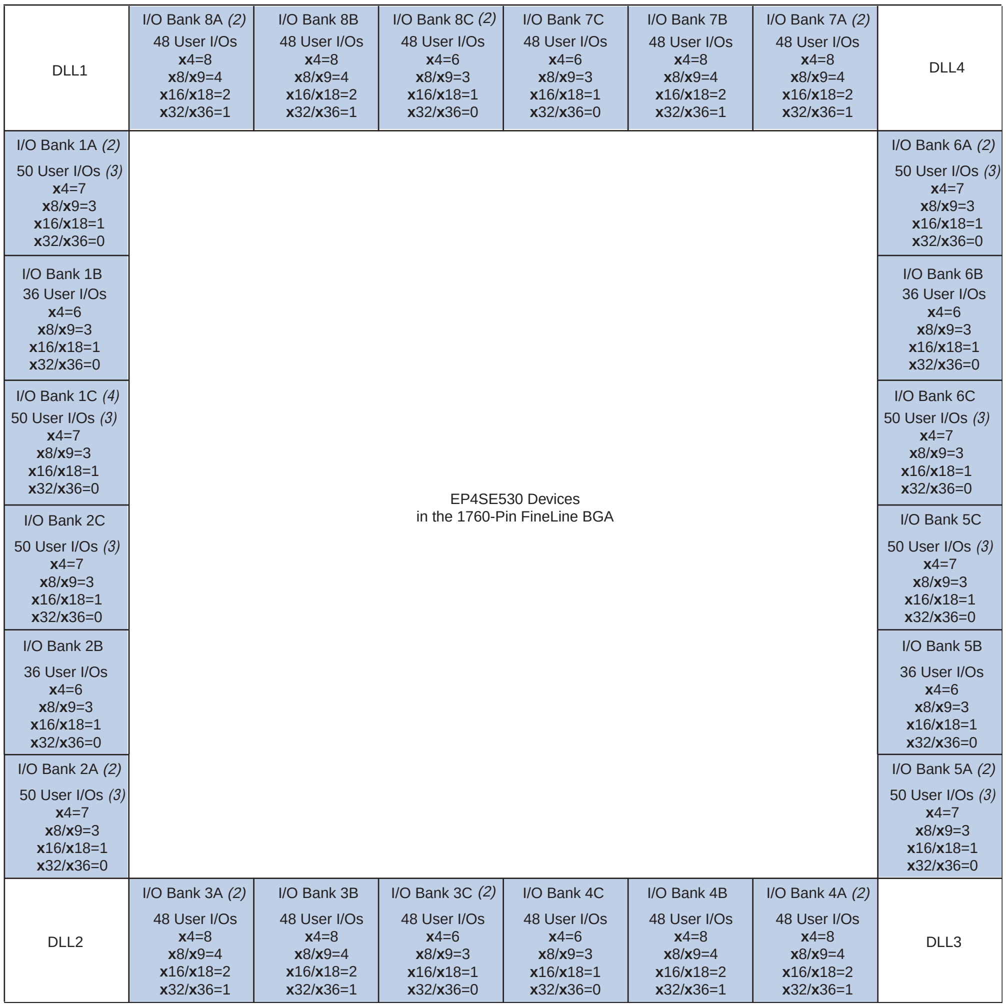
(1) These numbers are preliminary until the devices are available.

(2) You can also use DQS/DQSn pins in some of the ×4 groups as R_{UP}/R_{DN} pins. You cannot use a ×4 group for memory interfaces if two pins of the group are being used as R_{UP} and R_{DN} pins for OCT calibration. You can still use the ×16/×18 or ×32/×36 groups that include these ×4 groups. However, there are restrictions on using ×8/×9 groups that include these ×4 groups.

(3) All I/O pin counts include eight dedicated clock inputs (`CLK1p`, `CLK1n`, `CLK3p`, `CLK3n`, `CLK8p`, `CLK8n`, `CLK10p`, and `CLK10n`) and eight dedicated corner PLL clock inputs (`PLL_L1_CLKp`, `PLL_L1_CLKn`, `PLL_L4_CLKp`, `PLL_L4_CLKn`, `PLL_R4_CLKp`, `PLL_R4_CLKn`, `PLL_R1_CLKp`, and `PLL_R1_CLKn`) that you can use for data inputs.

(4) You can also use some of the DQS/DQ pins in I/O Bank 1C as configuration pins. You cannot use a ×4 DQS/DQ group with any of its pin members used for configuration purposes. Make sure that the DQS/DQ groups that you have chosen are not also used for configuration as you may lose up to four ×4 DQS/DQ groups, depending on your configuration scheme.

Figure 7–15. Number of DQS/DQ Groups per Bank in EP4SE680 Devices in the 1760-pin FineLine BGA Package *(Note 1)*



DLL1

I/O Bank 8A *(2)*
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 8B
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 8C *(2)*
48 User I/Os
x4=6
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 7C
48 User I/Os
x4=6
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 7B
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 7A *(2)*
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

DLL4

I/O Bank 1A *(2)*
50 User I/Os *(3)*
x4=7
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 1B
36 User I/Os
x4=6
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 1C *(4)*
50 User I/Os *(3)*
x4=7
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 2C
50 User I/Os *(3)*
x4=7
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 2B
36 User I/Os
x4=6
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 2A *(2)*
50 User I/Os *(3)*
x4=7
x8/x9=3
x16/x18=1
x32/x36=0

EP4SE530 Devices
in the 1760-Pin FineLine BGA

I/O Bank 6A *(2)*
50 User I/Os *(3)*
x4=7
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 6B
36 User I/Os
x4=6
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 6C
50 User I/Os *(3)*
x4=7
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 5C
50 User I/Os *(3)*
x4=7
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 5B
36 User I/Os
x4=6
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 5A *(2)*
50 User I/Os *(3)*
x4=7
x8/x9=3
x16/x18=1
x32/x36=0

DLL2

I/O Bank 3A *(2)*
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 3B
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 3C *(2)*
48 User I/Os
x4=6
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 4C
48 User I/Os
x4=6
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 4B
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 4A *(2)*
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

DLL3

Notes to Figure 7–15:

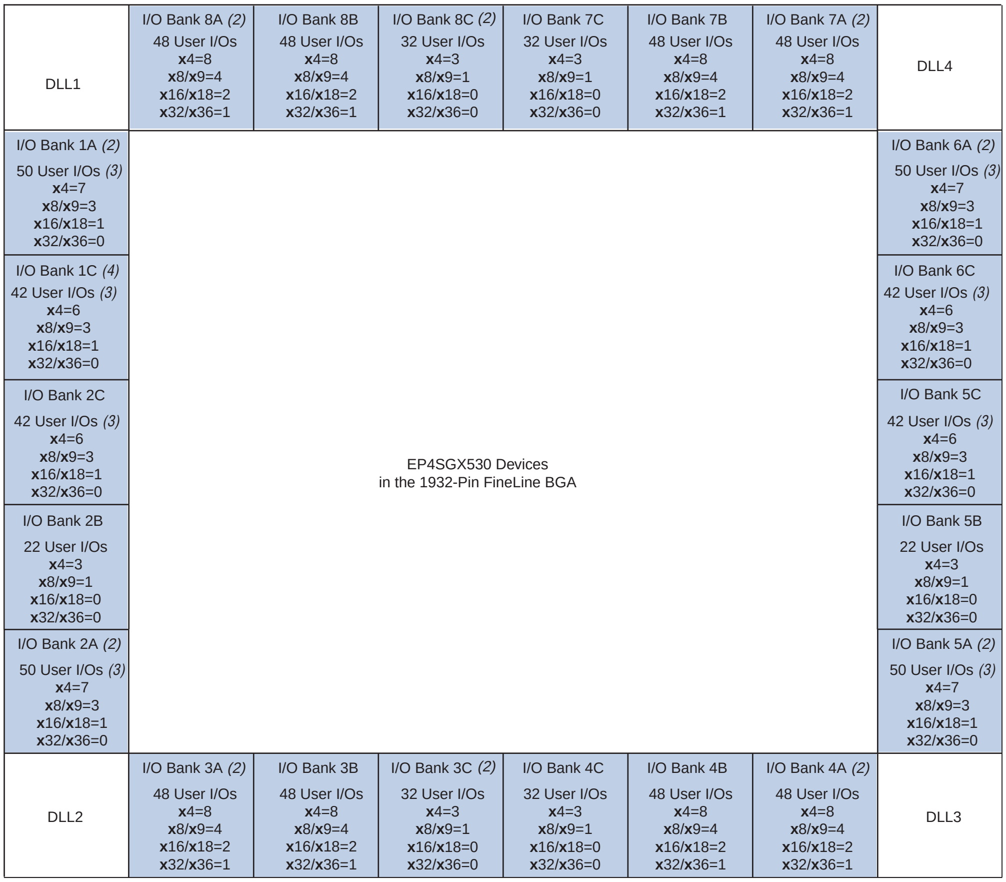
(1) These numbers are preliminary until the devices are available.

(2) You can also use DQS/DQSn pins in some of the ×4 groups as R_{UP}/R_{DN} pins. You cannot use a ×4 group for memory interfaces if two pins of the group are being used as R_{UP} and R_{DN} pins for OCT calibration. You can still use the ×16/×18 or ×32/×36 groups that include these ×4 groups. However, there are restrictions on using ×8/×9 groups that include these ×4 groups.

(3) All I/O pin counts include eight dedicated clock inputs (`CLK1p`, `CLK1n`, `CLK3p`, `CLK3n`, `CLK8p`, `CLK8n`, `CLK10p`, and `CLK10n`) and eight dedicated corner PLL clock inputs (`PLL_L1_CLKp`, `PLL_L1_CLKn`, `PLL_L4_CLKp`, `PLL_L4_CLKn`, `PLL_R4_CLKp`, `PLL_R4_CLKn`, `PLL_R1_CLKp`, and `PLL_R1_CLKn`) that you can use for data inputs.

(4) You can also use some of the DQS/DQ pins in I/O Bank 1C as configuration pins. You cannot use a ×4 DQS/DQ group with any of its pin members used for configuration purposes. Make sure that the DQS/DQ groups that you have chosen are not also used for configuration as you may lose up to four ×4 DQS/DQ groups, depending on your configuration scheme.

Figure 7–16. Number of DQS/DQ Groups per Bank in EP4SGX530 Devices in the 1932-Pin FineLine BGA Package *(Note 1)*



DLL1

I/O Bank 8A *(2)*
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 8B
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 8C *(2)*
32 User I/Os
x4=3
x8/x9=1
x16/x18=0
x32/x36=0

I/O Bank 7C
32 User I/Os
x4=3
x8/x9=1
x16/x18=0
x32/x36=0

I/O Bank 7B
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 7A *(2)*
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

DLL4

I/O Bank 1A *(2)*
50 User I/Os *(3)*
x4=7
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 1C *(4)*
42 User I/Os *(3)*
x4=6
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 2C
42 User I/Os *(3)*
x4=6
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 2B
22 User I/Os
x4=3
x8/x9=1
x16/x18=0
x32/x36=0

I/O Bank 2A *(2)*
50 User I/Os *(3)*
x4=7
x8/x9=3
x16/x18=1
x32/x36=0

EP4SGX530 Devices
in the 1932-Pin FineLine BGA

I/O Bank 6A *(2)*
50 User I/Os *(3)*
x4=7
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 6C
42 User I/Os *(3)*
x4=6
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 5C
42 User I/Os *(3)*
x4=6
x8/x9=3
x16/x18=1
x32/x36=0

I/O Bank 5B
22 User I/Os
x4=3
x8/x9=1
x16/x18=0
x32/x36=0

I/O Bank 5A *(2)*
50 User I/Os *(3)*
x4=7
x8/x9=3
x16/x18=1
x32/x36=0

DLL2

I/O Bank 3A *(2)*
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 3B
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 3C *(2)*
32 User I/Os
x4=3
x8/x9=1
x16/x18=0
x32/x36=0

I/O Bank 4C
32 User I/Os
x4=3
x8/x9=1
x16/x18=0
x32/x36=0

I/O Bank 4B
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

I/O Bank 4A *(2)*
48 User I/Os
x4=8
x8/x9=4
x16/x18=2
x32/x36=1

DLL3

Notes to Table 7–16:

(1) These numbers are preliminary until the devices are available.

(2) You can also use DQS/DQSn pins in some of the ×4 groups as R_{UP}/R_{DN} pins. You cannot use a ×4 group for memory interfaces if two pins of the group are being used as R_{UP} and R_{DN} pins for OCT calibration. You can still use the ×16/×18 or ×32/×36 groups that include these ×4 groups. However, there are restrictions on using ×8/×9 groups that include these ×4 groups.

(3) All I/O pin counts include eight dedicated clock inputs (CLK1p, CLK1n, CLK3p, CLK3n, CLK8p, CLK8n, CLK10p, and CLK10n) and eight dedicated corner PLL clock inputs (PLL_L1_CLKp, PLL_L1_CLKn, PLL_L4_CLKp, PLL_L4_CLKn, PLL_R4_CLKp, PLL_R4_CLKn, PLL_R1_CLKp, and PLL_R1_CLKn) that you can use for data inputs.

(4) You can also use some of the DQS/DQ pins in I/O Bank 1C as configuration pins. You cannot use a ×4 DQS/DQ group with any of its pin members used for configuration purposes. Make sure that the DQS/DQ groups that you have chosen are not also used for configuration as you may lose up to four ×4 DQS/DQ groups, depending on your configuration scheme.

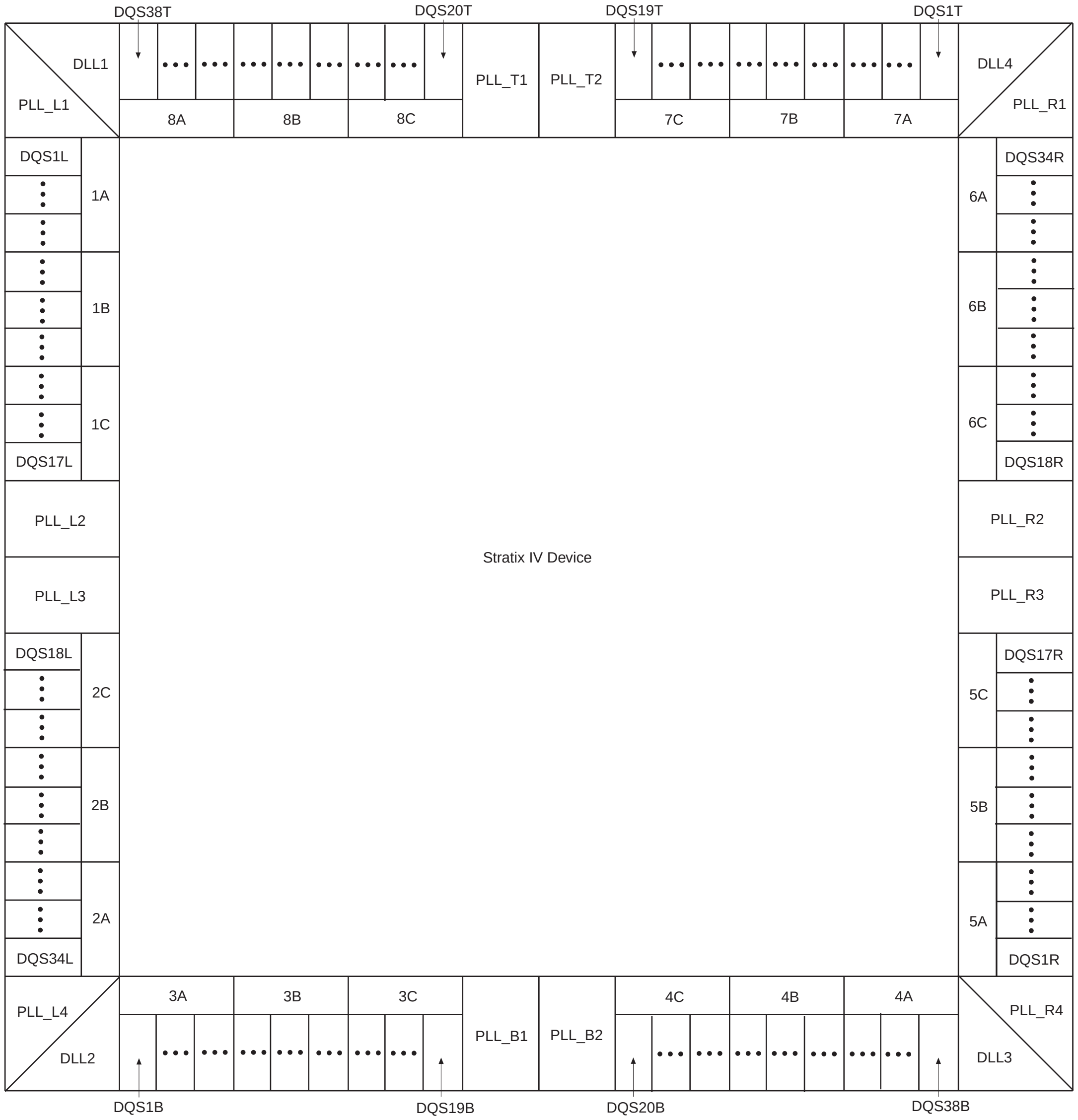
The DQS and DQSn pins are listed in the Stratix IV pin tables as DQSXY and DQSnXY, respectively, where X denotes the DQS/DQ grouping number and Y denotes whether the group is located on the top (T), bottom (B), left (L), or right (R) side of the device. The DQS/DQ pin numbering is based on ×4 mode.

The corresponding DQ pins are marked as DQXY, where X indicates to which DQS group the pins belong to and Y indicates whether the group is located on the top (T), bottom (B), left (L), or right (R) side of the device. For example, DQS1L indicates a DQS pin located on the left side of the device. The DQ pins belonging to that group are shown as DQ1L in the pin table. See Figure 7–17 for illustrations.

☞ The parity, DM, BWSn, NWSn, ECC, and QVLD pins are shown as DQ pins in the pin table.

The numbering scheme starts from the top-left side of the device going counter-clockwise in a die-top view. Figure 7–17 shows how the DQS/DQ groups are numbered in a die-top view of the device. The top and bottom sides of the device can contain up to 38 ×4 DQS/DQ groups. The left and right sides of the device can contain up to 34 ×4 DQS/DQ groups.

Figure 7–17. DQS Pins in Stratix IV I/O Banks



Using R_{UP}/R_{DN} Pins in a DQS/DQ Group Used for Memory Interfaces

You can also use DQS/DQSn pins in some of the ×4 groups as R_{UP}/R_{DN} pins (listed in the pin table). You cannot use a ×4 DQS/DQ group for memory interfaces if any of its pin members are being used as R_{UP} and R_{DN} pins for OCT calibration. You may be able to use the ×8/×9 group that includes this ×4 DQS/DQ group, if either of the following applies:

- You are not using DM pins with your differential DQS pins
- You are not using complementary or differential DQS pins

This is because a DQS/DQ ×8/×9 group actually comprises 12 pins, as the groups are formed by stitching two DQS/DQ groups in ×4 mode with six total pins each (see Table 7–4). A typical ×8 memory interface consists of one DQS, one DM, and eight DQ pins which add up to 10 pins. If you choose your pin assignment carefully, you can use the two extra pins for R_{UP} and R_{DN}. In a DDR3 SDRAM interface, you have to use differential DQS, which means that you only have one extra pin. In this case, pick different pin locations for R_{UP} and R_{DN} pins (for example, in the bank that contains address and command pins).

You cannot use R_{UP} and R_{DN} pins shared with DQS/DQ group pins when using ×9 QDRII+/QDRII SRAM devices, as the R_{UP} and R_{DN} pins are dual purpose with the CQn pins. In this case, pick different pin locations for R_{UP} and R_{DN} pins to avoid conflict with memory interface pin placement. In this case, you have the choice of placing the R_{UP} and R_{DN} pins in the data-write group or in the same bank as the address and command pins.

There is no restriction of using ×16/×18 or ×32/×36 DQS/DQ groups that include the ×4 groups whose pin members are being used as R_{UP} and R_{DN} pins as there are enough extra pins that can be used as DQS pins.

You must pick your DQS and DQ pins manually for the ×8, ×16/×18, or ×32/×36 DQS/DQ group whose members are being used for R_{UP} and R_{DN}, because the Quartus® II software might not be able to place this correctly when there are no specific pin assignments and might give you a "no-fit" instead.

Combining ×16/×18 DQS/DQ groups for ×36 QDRII+/QDRII SRAM Interface

This implementation combines ×16/×18 DQS/DQ groups to interface with a ×36 QDRII+/QDRII SRAM device. The ×36 read data bus uses two ×16/×18 groups while the ×36 write data uses another two ×16/×18 groups. The CQ/CQn signal traces are split on the board trace to connect to two pairs of CQ/CQn pins in the FPGA. This is the only connection on the board that you need to change for this implementation. Other QDRII+/QDRII SRAM interface rules for Stratix IV devices also apply for this implementation.

The ALTMEMPHY megafunction does not use the QVLD signal, so you can leave the QVLD signal unconnected as in any QDRII+/QDRII SRAM interfaces in Stratix IV devices.

For more information about the ALTMEMPHY megafunction, refer to the *ALTMEMPHY Megafunction User Guide*.

Table 7–6 summarizes the maximum clock rate supported with this implementation.

Table 7–6. Stratix IV Maximum Clock Rate Support with the ×36 Mode Emulation *(Note 1)*, *(2)*, *(3)*

Memory Standards	-2/2X Speed Grade (MHz)		-3 Speed Grade (MHz)		-4 Speed Grade (MHz)	
	Column I/O Banks	Row I/O Banks	Column I/O Banks	Row I/O Banks	Column I/O Banks	Row I/O Banks
QDRII+ SRAM (2.5 clock-cycle latency only) *(4)*	300	250	250	167	250	167
QDRII SRAM (1.5-V and 1.8-V HSTL)	300	250	250	167	250	167

Notes to Table 7–6:

(1) Numbers, based on using the half-rate controller, are preliminary until characterization is final. The supported operating frequencies listed here are memory interface maximums for the FPGA device family. Your design's actual achievable performance is based on design- and system-specific factors, as well as static timing analysis of the completed design.

(2) The performance listed in this table is lower than the performance listed in Table 7–1 due to double loading of the CQ/CQn pins. Double loading causes degradation in the signal slew rate, which affects FPGA delay. Furthermore, due to the difference in slew rate, there is a shift in the setup and hold-time window. You can perform an IBIS simulation to illustrate the shift in the clock signals.

(3) Column I/Os refer to top and bottom I/Os. Row I/Os refer to left and right I/Os.

(4) QDRII+ SRAM devices with 2.0 clock cycle latency are not supported, due to hardware limitations.

Rules to Combine Groups

In 780-, 1152-pin, and some of the 1517-pin package devices there is at most one ×16/×18 group per I/O sub-bank. You can combine two ×16/×18 groups from a single side of the device for a ×36 interface.

For devices that do not have four ×16/×18 groups in a single side of the device to form two ×36 groups for read and write data, you can form one ×36 group on one side of the device, and another ×36 group on the other side of the device. The two ×36 groups must be either on column I/O banks (top and bottom) or row I/O banks (left and right) only. Forming a ×36 group from column I/O banks and another ×36 group from row I/O banks for the read and write data buses is not supported.

For vertical migration with the ×36 emulation implementation, check if migration is possible by enabling device migration. Table 7–7 shows the possible I/O sub-bank combinations to form two ×36 groups on Stratix IV devices that do not have ×36 groups. Other Stratix IV devices in the 1517-, 1760- and 1932-pin packages support this implementation as well.

Table 7–7. I/O Sub-Bank Combinations to Form Two ×36 Groups for Stratix IV Devices Without x36 Groups (Part 1 of 2)

Package	Device	I/O Sub-Bank Combinations
780-Pin FBGA	EP4SGX70, EP4SGX110, EP4SGX230, EP4SGX290, EP4SGX360	3A & 4A, 7A & 8A (bottom and top banks)
	EP4SE110, EP4SE230, EP4SE290, EP4SE360	1A & 2A, 5A & 6A (left and right banks) 3A & 4A, 7A & 8A (bottom and top banks)

Table 7–7. I/O Sub-Bank Combinations to Form Two ×36 Groups for Stratix IV Devices Without x36 Groups (Part 2 of 2)

Package	Device	I/O Sub-Bank Combinations
1152-Pin FBGA	EP4SGX110	3A & 4A, 7A & 8A (bottom and top banks)
	EP4SGX230, EP4SGX290, EP4SGX360, EP4SGX530	1A & 1C, 6A & 6C (left and right banks) 3A & 3B, 4A & 4B (bottom banks) 7A & 7B, 8A & 8B (top banks)
	EP4SE290, EP4SE360, EP4SE530, EP4SE680	1A & 1C, 2A & 2C (left banks) 3A & 3B, 4A & 4B (bottom banks) 5A & 5C, 6A & 6C (right banks) 7A & 7B, 8A & 8B (top banks)
1517-Pin FBGA	EP4SGX230, EP4SGX290, EP4SGX360	1A & 1C, 2A & 2C (left banks) 3A & 3B, 4A & 4B (bottom banks) 5A & 5C, 6A & 6C (right banks) 7A & 7B, 8A & 8B (top banks)

Stratix IV External Memory Interface Features

Stratix IV devices are rich with features that allow robust high-performance external memory interfacing. The ALTMEMPHY megafunction allows you to use these external memory interface features and helps set up the physical interface (PHY) best suited for your system. This section describes each Stratix IV device feature that is used in external memory interfaces from the DQS phase-shift circuitry, DQS logic block, leveling multiplexers, and dynamic OCT control block.

☞ The ALTMEMPHY megafunction and the Altera® memory controller MegaCore® functions can run at half the frequency of the I/O interface of the memory devices to allow better timing management in high-speed memory interfaces. Stratix IV devices have built-in registers in the IOE to convert data from full-rate (the I/O frequency) to half-rate (the controller frequency) and vice versa. You can bypass these registers if your memory controller is not running at half the rate of the I/O frequency. When using the Altera memory controller MegaCore functions, the ALTMEMPHY megafunction is instantiated for you.

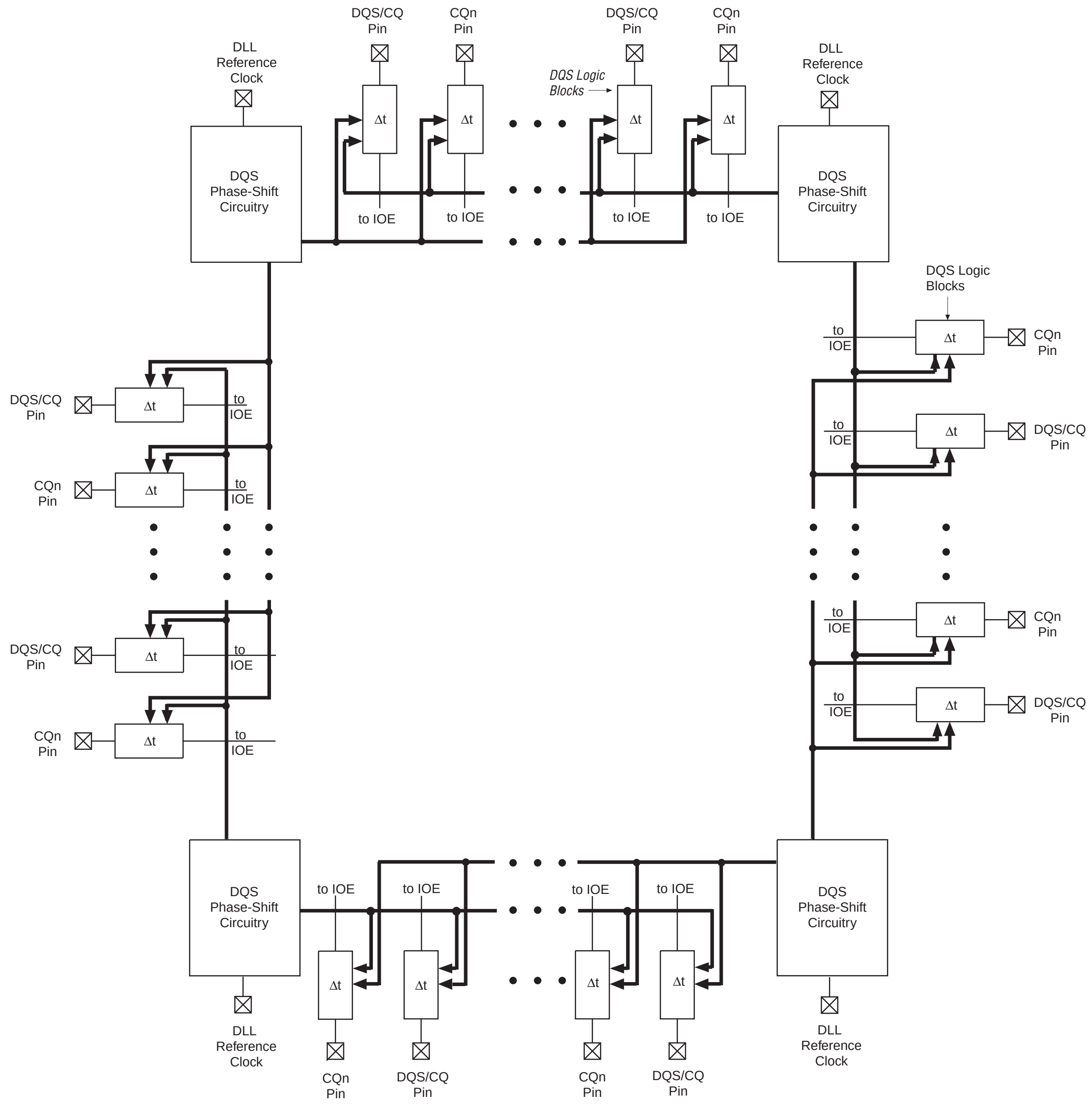
For more information about the ALTMEMPHY megafunction, refer to the *ALTMEMPHY Megafunction User Guide*.

DQS Phase-Shift Circuitry

Stratix IV phase-shift circuitry provides phase shift to the DQS/CQ and CQn pins on read transactions when the DQS/CQ and CQn pins are acting as input clocks or strobes to the FPGA. The DQS phase-shift circuitry consists of DLLs that are shared between multiple DQS pins and the phase-offset module to further fine-tune the DQS phase shift for different sides of the device.

Figure 7–18 shows how the DQS phase-shift circuitry is connected to the DQS/CQ and CQn pins in the device where memory interfaces are supported on all sides of the Stratix IV device.

Figure 7–18. DQS/CQ and CQn Pins and DQS Phase-Shift Circuitry *(Note 1)*, *(2)*



Notes to Figure 7–18:

(1) Refer to "DLL" on page 7–29 for possible reference input clock pins for each DLL.

(2) You can configure each DQS/CQ and CQn pin with a phase shift based on one of two possible DLL output settings.

DQS phase-shift circuitry is connected to DQS logic blocks that control each DQS/CQ or CQn pin. The DQS logic blocks allow the DQS delay settings to be updated concurrently at every DQS/CQ or CQn pin.

DLL

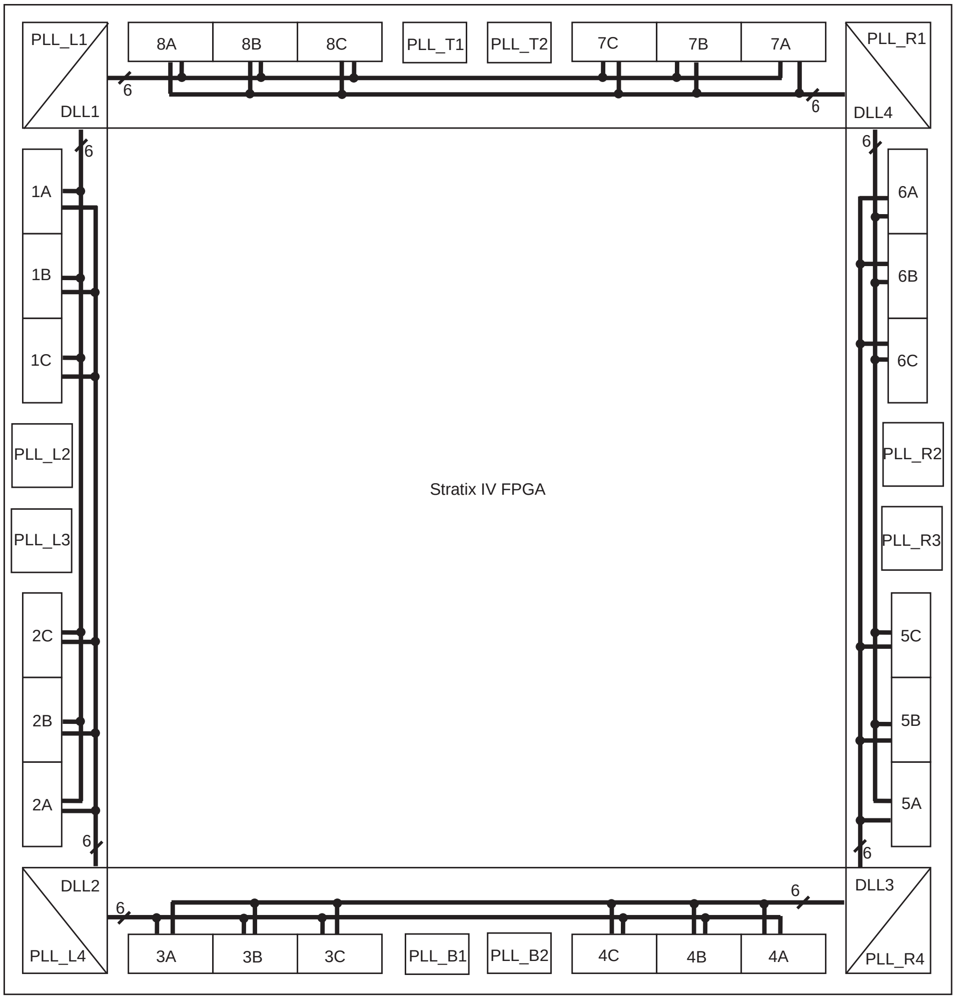
DQS phase-shift circuitry uses a DLL to dynamically control the clock delay needed by the DQS/CQ and CQn pin. The DLL, in turn, uses a frequency reference to dynamically generate control signals for the delay chains in each of the DQS/CQ and CQn pins, allowing it to compensate for PVT variations. The DQS delay settings are Gray-coded to reduce jitter when the DLL updates the settings. The phase-shift circuitry needs a maximum of 1280 clock cycles to lock and calculate the correct input clock period. Do not send data during these clock cycles because there is no guarantee that it will be captured properly. As the settings from the DLL may not be stable until this lock period has elapsed, be aware that anything using these settings (including the leveling delay system) may be unstable during this period.

☞ You can still use the DQS phase-shift circuitry for any memory interfaces that are less than 100 MHz. However, the DQS signal may not shift over 2.5 ns. Even if the DQS signal is not shifted exactly to the middle of the DQ valid window, the I/O element should still be able to capture the data in low-frequency applications in which a large amount of timing margin is available.

There are a maximum of four DLLs in a Stratix IV device, located in each corner of the device. These four DLLs support a maximum of four unique frequencies, with each DLL running at one frequency. Each DLL can have two outputs with different phase offsets, which allows one Stratix IV device to have eight different DLL phase shift settings.

Figure 7–19 shows the DLL and I/O bank locations in Stratix IV devices from a die-top view if all sides of the device support external memory interfaces.

Figure 7–19. Stratix IV DLL and I/O Bank Locations (Die-Top View)



The DLL can access the two adjacent sides from its location within the device. For example, DLL1 on the top left of the device can access the top side (I/O banks 7A, 7B, 7C, 8A, 8B, and 8C) and the left side of the device (I/O banks 1A, 1B, 1C, 2A, 2B, and 2C). This means that each I/O bank is accessible by two DLLs, giving more flexibility to create multiple frequencies and multiple-types interfaces. You can have two different interfaces with the same frequency on the two sides adjacent to a DLL, where the DLL controls the DQS delay settings for both interfaces.

☞ Interfaces with data groups that use a combination of both row and column I/Os of the device are not supported.

Each bank can use settings from either or both DLLs the bank is adjacent to. For example, `DQS1L` can get its phase-shift settings from DLL1, while `DQS2L` can get its phase-shift settings from DLL2. Table 7–8 lists the DLL location and supported I/O banks for Stratix IV devices.

☞ You can only have one memory interface in each I/O sub-bank (such as I/O sub-banks 1A, 1B, and 1C) when you use leveling delay chains. This is because there is only one leveling delay chain per I/O sub-bank.

Table 7–8. DLL Location and Supported I/O Banks

DLL	Location	Accessible I/O Banks *(1)*
DLL1	Top-left corner	1A, 1B, 1C, 2A, 2B, 2C, 7A, 7B, 7C, 8A, 8B, 8C
DLL2	Bottom-left corner	1A, 1B, 1C, 2A, 2B, 2C, 3A, 3B, 3C, 4A, 4B, 4C
DLL3	Bottom-right corner	3A, 3B, 3C, 4A, 4B, 4C, 5A, 5B, 5C, 6A, 6B, 6C
DLL4	Top-right corner	5A, 5B, 5C, 6A, 6B, 6C, 7A, 7B, 7C, 8A, 8B, 8C

Note to Table 7–8:

(1) The DLL can access these I/O banks if they are available for memory interfacing.

The reference clock for each DLL may come from PLL output clocks or any of the two dedicated clock input pins located in either side of the DLL. Table 7–9 through Table 7–18 show the available DLL reference clock input resources for the Stratix IV device family.

☞ When you have a dedicated PLL that only generates the DLL input reference clock, set the PLL mode to **No Compensation** to achieve better performance or the Quartus II software changes it automatically. Because the PLL does not use any other outputs, it does not need to compensate for any clock paths.

Table 7–9. DLL Reference Clock Input for EP4SGX70, EP4SGX110, and EP4SGX230 Devices in the 780-Pin FineLine BGA Package (Part 1 of 2)

DLL	CLKIN (Top/Bottom)	CLKIN (Left/Right)	PLL (Top/Bottom)	PLL (Left/Right)	PLL (Corner)
DLL1	CLK12P CLK13P CLK14P CLK15P	CLK0P CLK1P CLK2P CLK3P	PLL_T1	PLL_L2	Not Available
DLL2	CLK4P CLK5P CLK6P CLK7P	CLK0P CLK1P CLK2P CLK3P	PLL_B1	Not Available	Not Available

Table 7–9. DLL Reference Clock Input for EP4SGX70, EP4SGX110, and EP4SGX230 Devices in the 780-Pin FineLine BGA Package (Part 2 of 2)

DLL	CLKIN (Top/Bottom)	CLKIN (Left/Right)	PLL (Top/Bottom)	PLL (Left/Right)	PLL (Corner)
DLL3	CLK4P CLK5P CLK6P CLK7P	Not Available	PLL_B1	Not Available	Not Available
DLL4	CLK12P CLK13P CLK14P CLK15P	Not Available	PLL_T1	Not Available	Not Available

Table 7–10. DLL Reference Clock Input for EP4SE110, EP4SE230, EP4SE290, and EP4SE360 Devices in the 780-Pin FineLine BGA Package

DLL	CLKIN (Top/Bottom)	CLKIN (Left/Right)	PLL (Top/Bottom)	PLL (Left/Right)	PLL (Corner)
DLL1	CLK12P CLK13P CLK14P CLK15P	CLK0P CLK1P CLK2P CLK3P	PLL_T1	PLL_L2	Not Available
DLL2	CLK4P CLK5P CLK6P CLK7P	CLK0P CLK1P CLK2P CLK3P	PLL_B1	PLL_L2	Not Available
DLL3	CLK4P CLK5P CLK6P CLK7P	CLK8P CLK9P CLK10P CLK11P	PLL_B1	PLL_R2	Not Available
DLL4	CLK12P CLK13P CLK14P CLK15P	CLK8P CLK9P CLK10P CLK11P	PLL_T1	PLL_R2	Not Available

Table 7–11. DLL Reference Clock Input for EP4SGX290 and EP4SGX360 Devices in the 780-Pin FineLine BGA Package

DLL	CLKIN (Top/Bottom)	CLKIN (Left/Right)	PLL (Top/Bottom)	PLL (Left/Right)	PLL (Corner)
DLL1	CLK12P CLK13P CLK14P CLK15P	Not Available	PLL_T1	Not Available	Not Available
DLL2	CLK4P CLK5P CLK6P CLK7P	Not Available	PLL_B1	Not Available	Not Available
DLL3	CLK4P CLK5P CLK6P CLK7P	Not Available	PLL_B2	Not Available	Not Available
DLL4	CLK12P CLK13P CLK14P CLK15P	Not Available	PLL_T2	Not Available	Not Available

Table 7–12. DLL Reference Clock Input for EP4SGX110 Devices in the 1152-Pin FineLine BGA Package

DLL	CLKIN (Top/Bottom)	CLKIN (Left/Right)	PLL (Top/Bottom)	PLL (Left/Right)	PLL (Corner)
DLL1	CLK12P CLK13P CLK14P CLK15P	CLK0P CLK1P	PLL_T1	PLL_L2	Not Available
DLL2	CLK4P CLK5P CLK6P CLK7P	CLK0P CLK1P	PLL_B1	Not Available	Not Available
DLL3	CLK4P CLK5P CLK6P CLK7P	CLK10P CLK11P	PLL_B1	Not Available	Not Available
DLL4	CLK12P CLK13P CLK14P CLK15P	CLK10P CLK11P	PLL_T1	PLL_R2	Not Available

Table 7–13. DLL Reference Clock Input for EEP4SGX290 and EP4SGX360 Devices in the 780-Pin FineLine BGA Package

DLL	CLKIN (Top/Bottom)	CLKIN (Left/Right)	PLL (Top/Bottom)	PLL (Left/Right)	PLL (Corner)
DLL1	CLK12P CLK13P CLK14P CLK15P	Not Available	PLL_T1	Not Available	Not Available
DLL2	CLK4P CLK5P CLK6P CLK7P	Not Available	PLL_B1	Not Available	Not Available
DLL3	CLK4P CLK5P CLK6P CLK7P	Not Available	PLL_B2	Not Available	Not Available
DLL4	CLK12P CLK13P CLK14P CLK15P	Not Available	PLL_T2	Not Available	Not Available

Table 7–14. DLL Reference Clock Input for EP4SGX230, EP4SGX290, EP4SGX360, and EP4SGX530 Devices in the 1152-Pin FineLine BGA Package (with 16 Transceivers)

DLL	CLKIN (Top/Bottom)	CLKIN (Left/Right)	PLL (Top/Bottom)	PLL (Left/Right)	PLL (Corner)
DLL1	CLK12P CLK13P CLK14P CLK15P	CLK0P CLK1P	PLL_T1	PLL_L2	Not Available
DLL2	CLK4P CLK5P CLK6P CLK7P	CLK0P CLK1P	PLL_B1	Not Available	Not Available
DLL3	CLK4P CLK5P CLK6P CLK7P	CLK10P CLK11P	PLL_B2	Not Available	Not Available
DLL4	CLK12P CLK13P CLK14P CLK15P	CLK10P CLK11P	PLL_T2	PLL_R2	Not Available

Table 7–15. DLL Reference Clock Input for EEP4SGX290 and EP4SGX360 Devices in the 780-Pin FineLine BGA Package

DLL	CLKIN (Top/Bottom)	CLKIN (Left/Right)	PLL (Top/Bottom)	PLL (Left/Right)	PLL (Corner)
DLL1	CLK12P CLK13P CLK14P CLK15P	Not Available	PLL_T1	Not Available	Not Available
DLL2	CLK4P CLK5P CLK6P CLK7P	Not Available	PLL_B1	Not Available	Not Available
DLL3	CLK4P CLK5P CLK6P CLK7P	Not Available	PLL_B2	Not Available	Not Available
DLL4	CLK12P CLK13P CLK14P CLK15P	Not Available	PLL_T2	Not Available	Not Available

Table 7–16. DLL Reference Clock Input for EP4SE290, EP4SE360, EP4SE530, and EPSE680 Devices in the 1152-, 1517-, and 1760-Pin FineLine BGA Packages *(Note 1)* (Part 1 of 2)

DLL	CLKIN (Top/Bottom)	CLKIN (Left/Right)	PLL (Top/Bottom)	PLL (Left/Right)	PLL (Corner)
DLL1	CLK12P CLK13P CLK14P CLK15P	CLK0P CLK1P CLK2P CLK3P	PLL_T1	PLL_L2	Not Available
DLL2	CLK4P CLK5P CLK6P CLK7P	CLK0P CLK1P CLK2P CLK3P	PLL_B1	PLL_L3	Not Available
DLL3	CLK4P CLK5P CLK6P CLK7P	CLK8P CLK9P CLK10P CLK11P	PLL_B2	PLL_R3	Not Available

Table 7–16. DLL Reference Clock Input for EP4SE290, EP4SE360, EP4SE530, and EPSE680 Devices in the 1152-, 1517-, and 1760-Pin FineLine BGA Packages *(Note 1)* (Part 2 of 2)

DLL	CLKIN (Top/Bottom)	CLKIN (Left/Right)	PLL (Top/Bottom)	PLL (Left/Right)	PLL (Corner)
DLL4	CLK12P CLK13P CLK14P CLK15P	CLK8P CLK9P CLK10P CLK11P	PLL_T2	PLL_R2	Not Available

Note to Table 7–16:

(1) For the 1152-pin package, these devices have 24 transceivers.

Table 7–17. DLL Reference Clock Input for EP4SGX230, EP4SGX290, EP4SGX360, and EP4SGX530 Devices in the 1517-Pin FineLine BGA Package

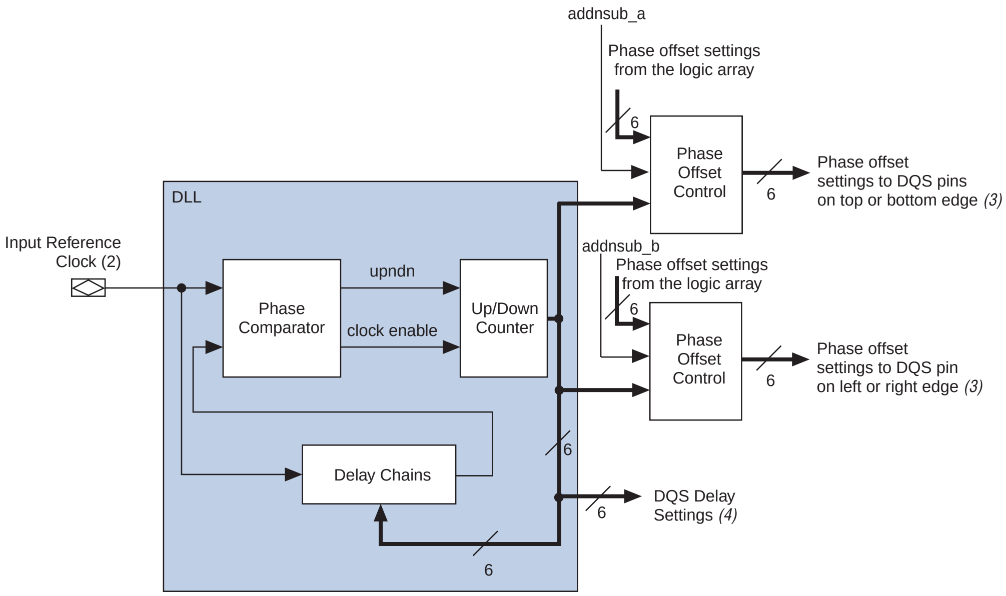
DLL	CLKIN (Top/Bottom)	CLKIN (Left/Right)	PLL (Top/Bottom)	PLL (Left/Right)	PLL (Corner)
DLL1	CLK12P CLK13P CLK14P CLK15P	CLK0P CLK1P CLK2P CLK3P	PLL_T1	PLL_L2	Not Available
DLL2	CLK4P CLK5P CLK6P CLK7P	CLK0P CLK1P CLK2P CLK3P	PLL_B1	PLL_L3	Not Available
DLL3	CLK4P CLK5P CLK6P CLK7P	CLK8P CLK9P CLK10P CLK11P	PLL_B2	PLL_R3	Not Available
DLL4	CLK12P CLK13P CLK14P CLK15P	CLK8P CLK9P CLK10P CLK11P	PLL_T2	PLL_R2	Not Available

Table 7–18. DLL Reference Clock Input for EP4SGX290, EP4SGX360, and EP4SGX530 Devices in the 1932-Pin FineLine BGA Package

DLL	CLKIN (Top/Bottom)	CLKIN (Left/Right)	PLL (Top/Bottom)	PLL (Left/Right)	PLL (Corner)
DLL1	CLK12P CLK13P CLK14P CLK15P	CLK0P CLK1P CLK2P CLK3P	PLL_T1	PLL_L2	Not Available
DLL2	CLK4P CLK5P CLK6P CLK7P	CLK0P CLK1P CLK2P CLK3P	PLL_B1	PLL_L3	Not Available
DLL3	CLK4P CLK5P CLK6P CLK7P	CLK8P CLK9P CLK10P CLK11P	PLL_B2	PLL_R3	Not Available
DLL4	CLK12P CLK13P CLK14P CLK15P	CLK8P CLK9P CLK10P CLK11P	PLL_T2	PLL_R2	Not Available

Figure 7–20 shows a simple block diagram of the DLL. The input reference clock goes into the DLL to a chain of up to 16 delay elements. The phase comparator compares the signal coming out of the end of the delay chain block to the input reference clock. The phase comparator then issues the upndn signal to the Gray-code counter. This signal increments or decrements a six-bit delay setting (DQS delay settings) that increases or decreases the delay through the delay element chain to bring the input reference clock and the signals coming out of the delay element chain in phase.

Figure 7–20. Simplified Diagram of the DQS Phase-Shift Circuitry *(Note 1)*


addnsub_a
Phase offset settings from the logic array
6
Phase Offset Control
6
Phase offset settings to DQS pins on top or bottom edge (3)
DLL
Input Reference Clock (2)
Phase Comparator
upndn
clock enable
Up/Down Counter
addnsub_b
Phase offset settings from the logic array
6
Phase Offset Control
6
Phase offset settings to DQS pin on left or right edge (3)
Delay Chains
6
6
DQS Delay Settings (4)
6

Notes to Figure 7–20:

(1) All features of the DQS phase-shift circuitry are accessible from the ALTMEMPHY megafunction in the Quartus II software.

(2) The input reference clock for the DQS phase-shift circuitry can come from a PLL output clock or an input clock pin. Refer to Table 7–9 through Table 7–12 for exact PLL and input clock pin.

(3) Phase offset settings can only go to the DQS logic blocks.

(4) DQS delay settings can go to the logic array, DQS logic block, and leveling circuitry.

You can reset the DLL from either the logic array or a user I/O pin. Each time the DLL is reset, you must wait for 1280 clock cycles for the DLL to lock before you can capture the data properly.

Depending on the DLL frequency mode, the DLL can shift the incoming DQS signals by 0°, 22.5°, 30°, 36°, 45°, 60°, 67.5°, 72°, 90°, 108°, 120°, 135°, 144°, or 180°. The shifted DQS signal is then used as the clock for the DQ IOE input registers.

All DQS/CQ and CQn pins, referenced to the same DLL, can have their input signal phase shifted by a different degree amount but all must be referenced at one particular frequency. For example, you can have a 90° phase shift on `DQS1T` and a 60° phase shift on `DQS2T`, referenced from a 200-MHz clock. Not all phase-shift combinations are supported. The phase shifts on the DQS pins referenced by the same DLL must all be a multiple of 22.5° (up to 90°), 30° (up to 120°), 36° (up to 144°), or 45° (up to 180°).

There are seven different frequency modes for the Stratix IV DLL, as shown in Table 7–19. Each frequency mode provides different phase shift selections. In frequency mode 0, 1, 2, and 3, the 6-bit DQS delay settings vary with PVT to implement the phase-shift delay. In frequency modes 4, 5, and 6, only 5 bits of the DQS delay settings vary with PVT to implement the phase-shift delay; the most significant bit of the DQS delay setting is set to `0`.

Table 7–19. Stratix IV DLL Frequency Modes

Frequency Mode	DQS Delay Setting Bus Width	Available Phase Shift	Number of Delay Chains
0	6 bits	22.5, 45, 67.5, 90	16
1	6 bits	30, 60, 90, 120	12
2	6 bits	36, 72, 108, 144	10
3	6 bits	45, 90, 135, 180	8
4	5 bits	30, 60, 90, 120	12
5	5 bits	36, 72, 108, 144	10
6	5 bits	45, 90, 135, 180	8

For the frequency range of each mode, refer to the *DC and Switching Characteristics of the Stratix IV Family* chapter in volume 4 of the *Stratix IV Device Handbook*.

For 0° shift, the DQS/CQ signal bypasses both the DLL and DQS logic blocks. The Quartus II software automatically sets the DQ input delay chains so that the skew between the DQ and DQS/CQ pin at the DQ IOE registers is negligible when 0° shift is implemented. You can feed the DQS delay settings to the DQS logic block and logic array.

The shifted DQS/CQ signal goes to the DQS bus to clock the IOE input registers of the DQ pins. The signal can also go into the logic array for resynchronization if you are not using IOE resynchronization registers. The shifted CQn signal can only go to the negative-edge input register in the DQ IOE and is only used for QDRII+ and QDRII SRAM interfaces.

Phase Offset Control

Each DLL has two phase-offset modules and can provide two separate DQS delay settings with independent offset, one for the top and bottom I/O bank and one for the left and right I/O bank, so you can fine-tune the DQS phase-shift settings between two different sides of the device. Even though you have independent phase offset control, the frequency of the interface using the same DLL has to be the same. Use the phase offset control module for making small shifts to the input signal and use the DQS phase-shift circuitry for larger signal shifts. For example, if the DLL only offers a multiple of 30° phase shift, but your interface needs a 67.5° phase shift on the DQS signal, you can use two delay chains in the DQS logic blocks to give you 60° phase shift and use the phase offset control feature to implement the extra 7.5° phase shift.

You can either use a static phase offset or a dynamic phase offset to implement the additional phase shift. The available additional phase shift is implemented in 2s: complement in Gray-code between settings –64 to +63 for frequency mode 0, 1, 2, and 3, and between settings –32 to +31 for frequency modes 4, 5, and 6. An additional bit indicates whether the setting has a positive or negative value. The settings are linear, each phase offset setting adds a delay amount specified in the *DC and Switching Characteristics of the Stratix IV Family* chapter in volume 4 of the *Stratix IV Device Handbook*. The DQS phase shift is the sum of the DLL delay settings and the user-selected phase offset settings whose top setting is 64 for frequency modes 0, 1, 2, and 3; and 32 for frequency modes 4, 5, and 6, so the actual physical offset setting range is 64 or 32 subtracted by the DQS delay settings from the DLL.

When using this feature, you need to monitor the DQS delay settings to know how many offsets you can add and subtract in the system. Note that the DQS delay settings output by the DLL are also Gray coded.

For example, if the DLL determines that DQS delay settings of 28 is needed to achieve a 30° phase shift in DLL frequency mode 1, you can subtract up to 28 phase offset settings and you can add up to 35 phase offset settings to achieve the optimal delay that you need. However, if the same DQS delay settings of 28 is needed to achieve 30° phase shift in DLL frequency mode 4, you can still subtract up to 28 phase offset settings, but you can only add up to 3 phase offset settings before the DQS delay settings reach their maximum settings, because DLL frequency mode 4 only uses 5-bit DLL delay settings.

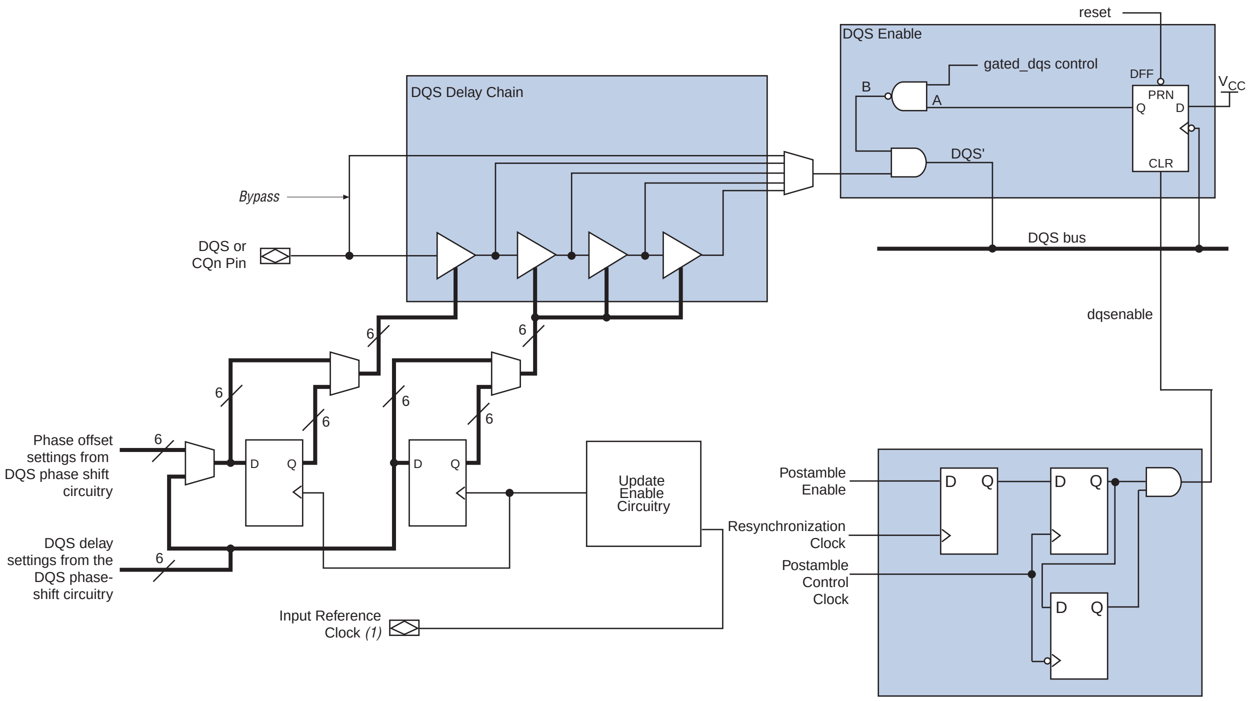
For more information about the value for each step, refer to the *DC and Switching Characteristics of the Stratix IV Family* chapter in volume 4 of the *Stratix IV Device Handbook*.

When using static phase offset, you can specify the phase offset amount in the ALTMEMPHY megafunction as a positive number for addition or a negative number for subtraction. You can also have a dynamic phase offset that is always added to, subtracted from, or both added to and subtracted from the DLL phase shift. When you always add or subtract, you can dynamically input the phase offset amount into the `dll_offset[5..0]` port. When you want to both add and subtract dynamically, you control the `addnsub` signal in addition to the `dll_offset[5..0]` signals.

DQS Logic Block

Each DQS and CQn pin is connected to a separate DQS logic block, which consists of the DQS delay chains, update enable circuitry, and DQS postamble circuitry (see Figure 7–21).

Figure 7–21. Stratix IV DQS Logic Block



Notes to Figure 7–21:

(1) The input reference clock for the DQS phase-shift circuitry can come from a PLL output clock or an input clock pin. Refer to Table 7–9 through Table 7–18 for the exact PLL and input clock pin.

(2) The `dqsenable` signal can also come from the Stratix IV FPGA fabric.

DQS Delay Chain

DQS delay chains consist of a set of variable delay elements to allow the input DQS/CQ and CQn signals to be shifted by the amount specified by the DQS phase-shift circuitry or the logic array. There are four delay elements in the DQS delay chain; the first delay chain closest to the DQS/CQ pin can either be shifted by the DQS delay settings or by the sum of the DQS delay setting and the phase-offset setting. The number of delay chains required is transparent because the ALTMEMPHY megafunction automatically sets it when you choose the operating frequency. The DQS delay settings can come from the DQS phase-shift circuitry on either end of the I/O banks or from the logic array.

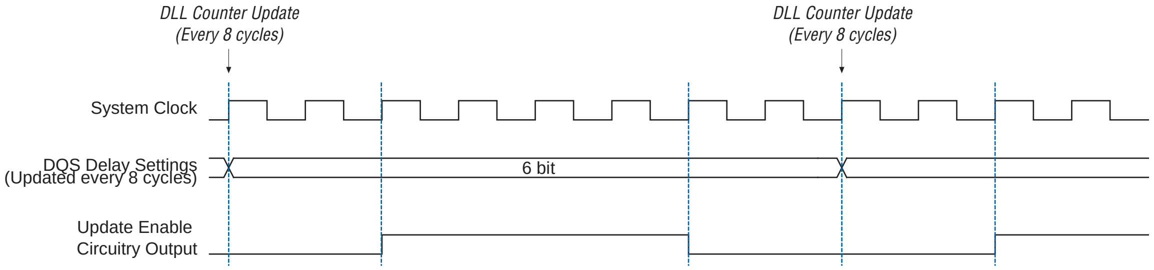
The delay elements in the DQS logic block have the same characteristics as the delay elements in the DLL. When the DLL is not used to control the DQS delay chains, you can input your own Gray-coded 6-bit or 5-bit settings using the `dqs_delayctrlin[5..0]` signals available in the ALTMEMPHY megafunction. These settings control 1, 2, 3, or all 4 delay elements in the DQS delay chains. The ALTMEMPHY megafunction can also dynamically choose the number of DQS delay chains needed for the system. The amount of delay is equal to the sum of the delay element's intrinsic delay and the product of the number of delay steps and the value of the delay steps.

You can also bypass the DQS delay chain to achieve 0° phase shift.

Update Enable Circuitry

Both the DQS delay settings and the phase-offset settings pass through a register before going into the DQS delay chains. The registers are controlled by the update enable circuitry to allow enough time for any changes in the DQS delay setting bits to arrive at all the delay elements. This allows them to be adjusted at the same time. The update enable circuitry enables the registers to allow enough time for the DQS delay settings to travel from the DQS phase-shift circuitry or core logic to all the DQS logic blocks before the next change. It uses the input reference clock or a user clock from the core to generate the update enable output. The ALTMEMPHY megafunction uses this circuit by default. See Figure 7–22 for an example waveform of the update enable circuitry output.

Figure 7–22. DQS Update Enable Waveform

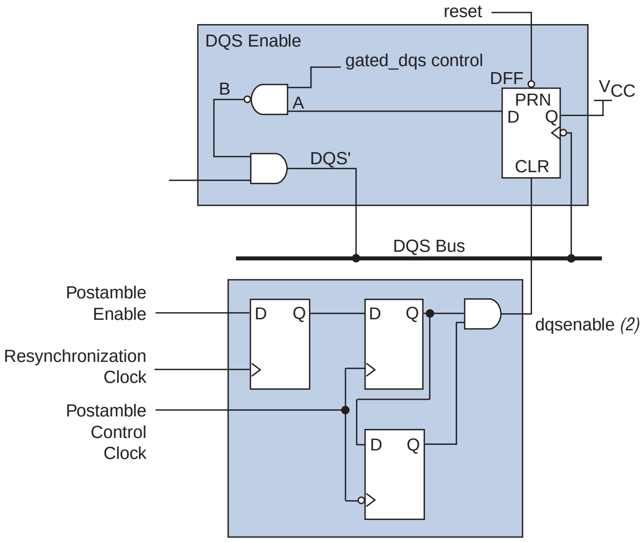


DQS Postamble Circuitry

For external memory interfaces that use a bidirectional read strobe such as in DDR3, DDR2, and DDR SDRAM, the DQS signal is low before going to or coming from a high-impedance state. The state in which DQS is low, just after a high-impedance state, is called the preamble; the state in which DQS is low, just before it returns to a high-impedance state, is called the postamble. There are preamble and postamble specifications for both read and write operations in DDR3, DDR2, and DDR SDRAM. The DQS postamble circuitry, featured in Figure 7–23, ensures that data is not lost if there is noise on the DQS line at the end of a read postamble time.

Stratix IV devices have dedicated postamble registers that can be controlled to ground the shifted DQS signal used to clock the DQ input registers at the end of a read operation. This ensures that any glitches on the DQS input signals at the end of the read postamble time do not affect the DQ IOE registers.

Figure 7–23. Stratix IV DQS Postamble Circuitry *(Note 1)*



Notes to Figure 7–23:

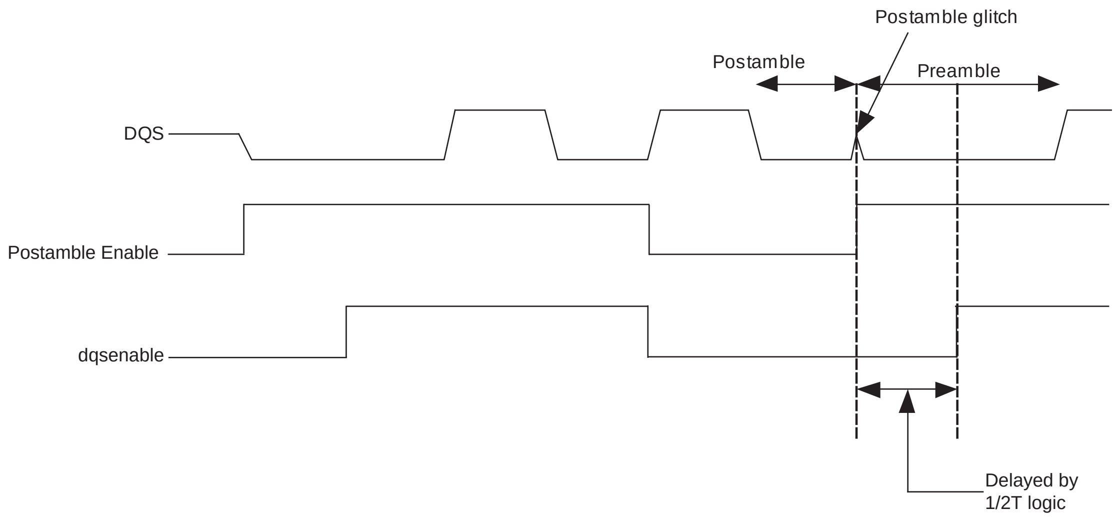
(1) The postamble clock can come from any of the delayed resynchronization clock taps although it is not necessarily of the same phase as the resynchronization clock.

(2) The `dqsenable` signal can also come from the Stratix IV FPGA fabric.

In addition to the dedicated postamble register, Stratix IV devices also have an HDR block inside the postamble enable circuitry. These registers are used if the controller is running at half the frequency of the I/Os.

Using the HDR block as the first stage capture register in the postamble enable circuitry block in Figure 7–23 is optional. The HDR block is clocked by the half-rate resynchronization clock, which is the output of the I/O clock divider circuit (shown in Figure 7–29). There is an AND gate after the postamble register outputs that is used to avoid postamble glitches from a previous read burst on a non-consecutive read burst. This scheme allows a half-a-clock cycle latency for `dqsenable` assertion and zero latency for `dqsenable` deassertion, as shown in Figure 7–24.

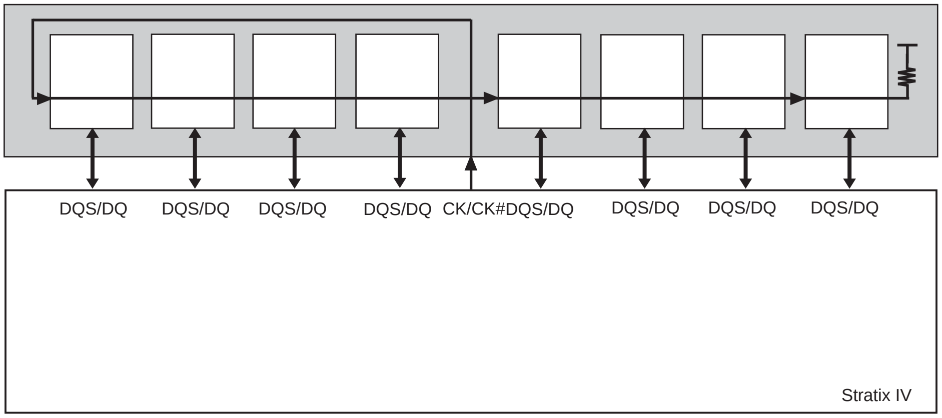
Figure 7–24. Avoiding Glitch on a Non-Consecutive Read Burst Waveform



Leveling Circuitry

DDR3 SDRAM unbuffered modules use a fly-by clock distribution topology for better signal integrity. This means that the CK/CK# signals arrive at each DDR3 SDRAM device in the module at different times. The difference in arrival time between the first DDR3 SDRAM device and the last device on the module can be as long as 1.6 ns. Figure 7–25 shows the clock topology in DDR3 SDRAM unbuffered modules.

Figure 7–25. DDR3 SDRAM Unbuffered Module Clock Topology



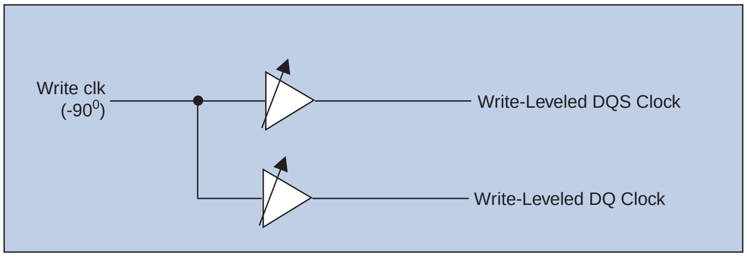
Because the data and read strobe signals are still point-to-point, take special care to ensure that the timing relationship between the CK/CK# and DQS signals (`tDQSS`, `tDSS`, and `tDSH`) during a write is met at every device on the modules. Furthermore, read data coming back into the FPGA from the memory is also staggered in a similar way.

Stratix IV FPGAs have leveling circuitry to address these two situations. There is one group of leveling circuitry per I/O bank, with the same I/O number (for example, there is one leveling circuitry shared between I/O bank 1A, 1B, and 1C) located in the middle of the I/O bank. These delay chains are PVT-compensated by the same DQS delay settings as the DLL and DQS delay chains.

For frequencies equal to and above 400 MHz, the DLL uses eight delay chains, such that each delay chain generates a 45° delay. The generated clock phases are distributed to every DQS logic block that is available in the I/O bank. The delay chain taps then feeds a multiplexer controlled by the ALTMEMPHY megafunction to select which clock phases are to be used for that ×4 or × 8 DQS group. Each group can use a different tap output from the read-leveling and write-leveling delay chains to compensate for the different CK/CK# delay going into each device on the module.

Figure 7–26 and Figure 7–27 show the Stratix IV write- and read-leveling circuitry.

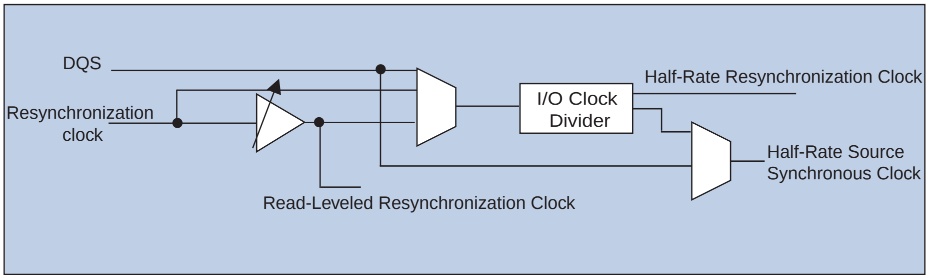
Figure 7–26. Stratix IV Write-Leveling Delay Chains and Multiplexers *(Note 1)*



Note to Figure 7–26:

(1) There is only one leveling delay chain per I/O bank with the same I/O number (for example, I/O banks 1A, 1B, and 1C). You can only have one memory controller in these I/O banks when you use the leveling delay chains.

Figure 7–27. Stratix IV Read-Leveling Delay Chains and Multiplexers *(Note 1)*



Note to Figure 7–27:

(1) There is only one leveling delay chain per I/O bank with the same I/O number (for example, I/O banks 1A, 1B, and 1C). You can only have one memory controller in these I/O banks when you use the leveling delay chains.

The –90° write clock of the ALTMEMPHY megafunction feeds the write-leveling circuitry to produce the clock to generate the DQS and DQ signals. During initialization, the ALTMEMPHY megafunction picks the correct write-leveled clock for the DQS and DQ clocks for each DQS/DQ group after sweeping all the available clocks in the write calibration process. The DQ clock output is –90° phase-shifted compared to the DQS clock output.

Similarly, the resynchronization clock feeds the read-leveling circuitry to produce the optimal resynchronization and postamble clock for each DQS/DQ group in the calibration process. The resynchronization and postamble clocks can use different clock outputs from the leveling circuitry. The output from the read-leveling circuitry can also generate the half-rate resynchronization clock that goes to the FPGA fabric.

☞ The ALTMEMPHY megafunction dynamically calibrates the alignment for read and write-leveling during the initialization process.

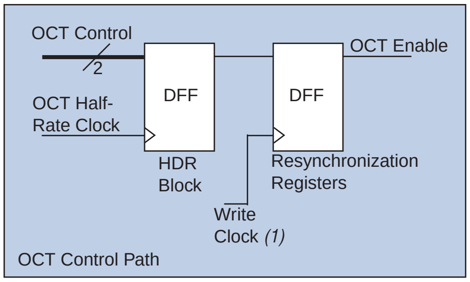
For more information about the ALTMEMPHY megafunction, refer to the *ALTMEMPHY Megafunction User Guide*.

Dynamic On-Chip Termination Control

Figure 7–28 shows the dynamic OCT control block. The block includes all the registers needed to dynamically turn on OCT RT during a read and turn OCT RT off during a write.

For more information about dynamic on-chip termination control, refer to“Dynamic On-Chip Termination Control” on page 7–46 or to the *I/O Features in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook*.

Figure 7–28. Stratix IV Dynamic OCT Control Block



Note to Figure 7–28:

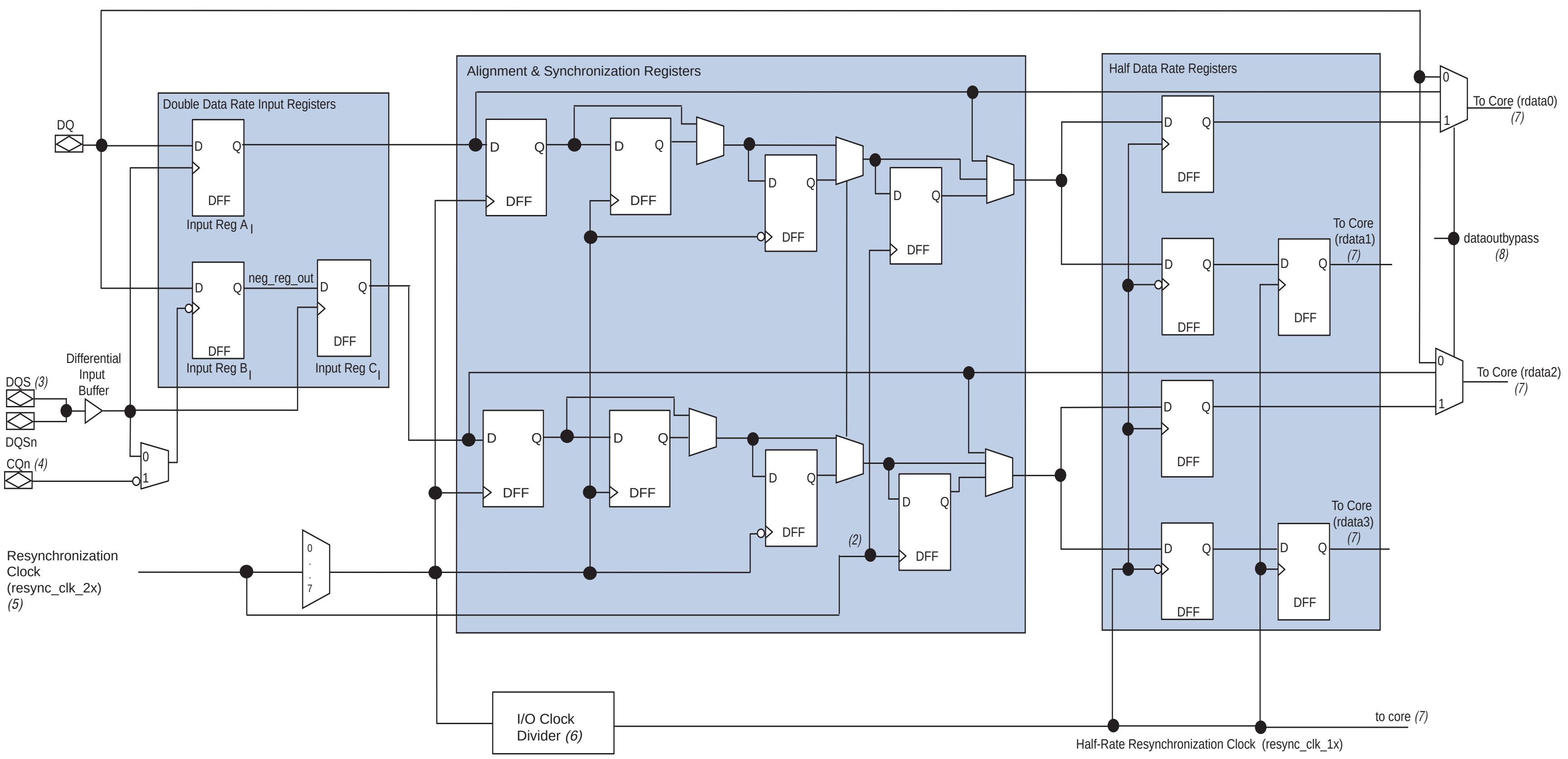
(1) The write clock comes from either the PLL or the write-leveling delay chain.

I/O Element Registers

The IOE registers are expanded to allow source-synchronous systems to have faster register-to-register transfers and resynchronization. Both top and bottom and left and right IOEs have the same capability. Left and right IOEs have extra features to support LVDS data transfer.

Figure 7–29 shows the registers available in the Stratix IV input path. The input path consists of the DDR input registers, resynchronization registers, and HDR block. You can bypass each block of the input path.

Figure 7–29. Stratix IV IOE Input Registers *(Note 1)*



Notes to Figure 7–29:

(1) You can bypass each register block in this path.

(2) This is the 0-phase resynchronization clock (from the read-leveling delay chain).

(3) The input clock can be from the DQS logic block (whether the postamble circuitry is bypassed or not) or from a global clock line.

(4) This input clock comes from the CQn logic block.

(5) This resynchronization clock can come either from the PLL or from the read-leveling delay chain.

(6) The I/O clock divider resides adjacent to the DQS logic block. In addition to the PLL and read-leveled resync clock, the I/O clock divider can also be fed by the DQS bus or CQn bus.

(7) The half-rate data and clock signals feed into a dual-port RAM in the FPGA core.

(8) You can dynamically change the `dataoutbypass` signal after configuration.

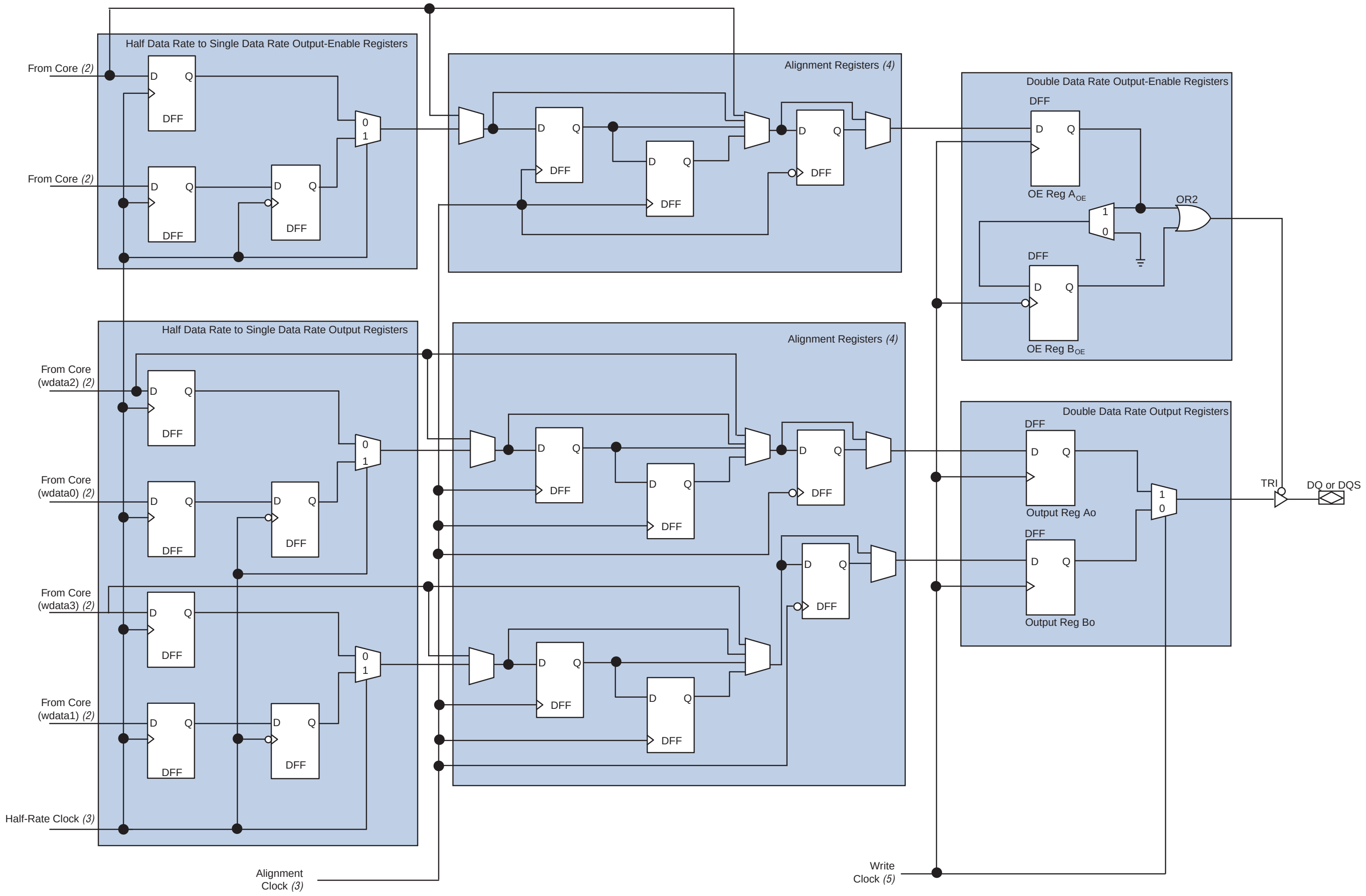
There are three registers in the DDR input registers block. Two registers capture data on the positive and negative edges of the clock, while the third register aligns the captured data. You can choose to use the same clock for the positive edge and negative edge registers, or two complementary clocks (DQS/CQ for the positive-edge register and DQSn/CQn for the negative-edge register). The third register that aligns the captured data uses the same clock as the positive edge registers.

The resynchronization registers consist of up to three levels of registers to resynchronize the data to the system clock domain. These registers are clocked by the resynchronization clock that is either generated by the PLL or the read-leveling delay chain. The outputs of the resynchronization registers can go straight to the core or to the HDR blocks, which are clocked by the divided-down resynchronization clock.

For more information about the read-leveling delay chain, refer to "Leveling Circuitry" on page 7–44.

Figure 7–30 shows the registers available in the Stratix IV output and output-enable paths. The path is divided into the HDR block, resynchronization registers, and output and output-enable registers. The device can bypass each block of the output and output-enable path.

Figure 7–30. Stratix IV IOE Output and Output-Enable Path Registers *(Note 1)*



Notes to Figure 7–30:

(1) You can bypass each register block of the output and output-enable paths.

(2) Data coming from the FPGA core are at half the frequency of the memory interface clock frequency in half-rate mode.

(3) The half-rate clock comes from the PLL, while the alignment clock comes from the write-leveling delay chains.

(4) These registers are only used in DDR3 SDRAM interfaces for write-leveling purposes.

(5) The write clock can come from either the PLL or from the write-leveling delay chain. The DQ write clock and DQS write clock have a 90° offset between them.

The output path is designed to route combinatorial or registered SDR outputs and full-rate or half-rate DDR outputs from the FPGA core. Half-rate data is converted to full-rate using the HDR block, clocked by the half-rate clock from the PLL. The resynchronization registers are also clocked by the same 0° system clock, except in the DDR3 SDRAM interface. In DDR3 SDRAM interfaces, the leveling registers are clocked by the write-leveling clock.

For more information about the write-leveling delay chain, refer to "Leveling Circuitry" on page 7–44.

The output-enable path has a structure similar to the output path. You can have a combinatorial or registered output in SDR applications and you can use half-rate or full-rate operation in DDR applications. You also have the resynchronization registers like the output path registers structure, ensuring that the output enable path goes through the same delay and latency as the output path.

Conclusion

Stratix IV devices have many features available to support existing and emerging external memory interfaces. The ALTMEMPHY megafunction, built to support the Stratix IV memory interface features, allows you to easily implement your data path for use with either your own controller or Altera's IP controller.

In Stratix IV devices, most of the critical data transfers are taken care of for you in the IOE, alleviating the burden of having to close timing in the FPGA fabric. Furthermore, because most of the registers are in the IOE, data delays between registers are short, allowing the circuitry to work at a higher frequency. Dynamically calibrated OCT, slew rate adjustment, and programmable drive strength improve signal integrity, especially at higher frequencies of operation.

In addition, programmable delay chains, used for deskew, allow Stratix IV devices to achieve better margin for high-performance memory interfaces. Dynamic calibration of resynchronization and postamble clocks guarantee high performance over PVT variations. Leveling circuitry enables Stratix IV to support DDR3 modules, thus offering customers the choice of highest performance memory technologies. Stratix IV devices also offer memory interface support in any of 24 modular I/O banks with up to four different frequencies of operations.

Referenced Documents

This chapter references the following documents:

- *ALTMEMPHY Megafunction User Guide*
- *Clock Networks and PLLs in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook*
- *DC and Switching Characteristics of the Stratix IV Family* chapter in volume 4 of the *Stratix IV Device Handbook*
- *I/O Features in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook*

Document Revision History

Table 7–20 shows the revision history for this document.

Table 7–20. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	■ Updated Table 7–1, Table 7–2, Table 7–3, Table 7–4, Table 7–5, and Table 7–6 ■ Added Table 7–7 ■ Updated Figure 7–1 ■ Updated "Combining ×16/×18 DQS/DQ groups for ×36 QDRII+/QDRII SRAM Interface" on page 7–25 ■ Updated "Rules to Combine Groups" on page 7–26 ■ Updated "DQS Phase-Shift Circuitry" on page 7–27 ■ Updated Figure 7–18 ■ Updated Table 7–9, Table 7–10, Table 7–11, Table 7–12, Table 7–13, Table 7–14, Table 7–15, Table 7–16, Table 7–17, and Table 7–18 ■ Updated Figure 7–29 ■ Updated Figure 7–30 ■ Made minor editorial changes	—
May 2008 v1.0	Initial release.	—

8. High-Speed Differential I/O Interfaces and DPA in Stratix IV Devices

SIV51008-2.0

Introduction

High-speed differential I/Os offer significant advantages over single-ended I/Os and contribute to the overall system bandwidth achievable with Stratix® IV FPGAs.

The Stratix IV family has two offerings:

- Stratix IV GX devices with high-speed CDR based transceivers
- Stratix IV E devices without high-speed CDR based transceivers

Both Stratix IV GX devices and Stratix IV E devices have built-in SERDES circuitry that supports high-speed LVDS interfaces at data rates between 5 Mbps and 1.6 Gbps. The SERDES circuitry is configurable to support source-synchronous communication protocols such as Utopia, Rapid I/O, XSBI, SFI, and SPI, as well as asynchronous protocols such as SGMII and Gigabit Ethernet.

☞ All references to Stratix IV in this document apply to both Stratix IV GX devices and Stratix IV E devices.

The Stratix IV family has the following dedicated circuitry for high-speed differential I/O support:

- Differential I/O buffer
- Transmitter serializer
- Receiver deserializer
- Data realignment
- Dynamic phase aligner (DPA)
- Synchronizer (FIFO buffer)
- PLLs (located on left and right sides of the device)

For high-speed differential interfaces, the Stratix IV family supports the following differential I/O standards:

- Low voltage differential signaling (LVDS)
- Mini-LVDS
- Reduced swing differential signaling (RSDS)

Table 8–1 shows the data rate range supported by these three standards.

Table 8–1. Supported Data Rate Range *(Note 1)*

I/O Standards	LVDS		Mini-LVDS		RSDS	
	(Min)	(Max)	(Min)	(Max)	(Min)	(Max)
Data Rate Range in Non-DPA Mode *(2)* (Mbps)	5	Note(1)	5	340	5	230
Data Rate Range in DPA and Soft-CDR Mode *(2)* (Mbps)	150	1600	150	340	150	230

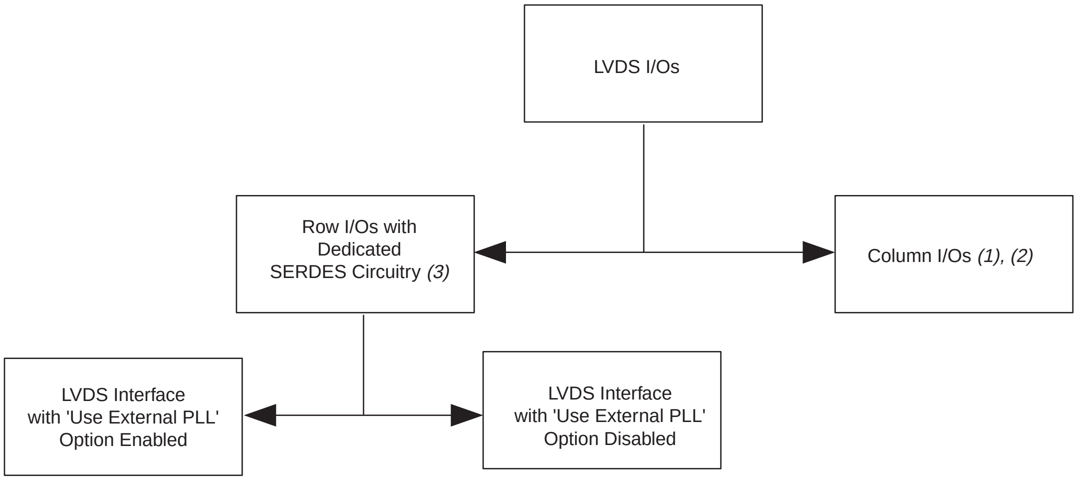
Notes to Table 8–1:

(1) RSKM > 0 is a necessity for the non-DPA mode to work correctly, where RSKM is receiver skew margin. The supported data rate for non-DPA mode depends on the system parameters (for example RCCS (receiver channel-to-channel skew)), which are used for RSKM calculations. To make calculations for the RSKM, follow the steps described in the section "Receiver Skew Margin for Non-DPA Mode" on page 8–30.

(2) There are three LVDS receiver modes: Non-DPA, DPA, and Soft-CDR. These three LVDS modes are described in the section "Receiver Data Path Modes" on page 8–20.

In the Stratix IV family, I/Os are divided into row and column I/Os. Figure 8–1 shows I/O bank support for the Stratix IV family. Row I/Os provide dedicated SERDES circuitry.

Figure 8–1. I/O Bank Support in Stratix IV Family


LVDS I/Os
Row I/Os with Dedicated SERDES Circuitry *(3)*
Column I/Os *(1), (2)*
LVDS Interface with 'Use External PLL' Option Enabled
LVDS Interface with 'Use External PLL' Option Disabled

Notes to Figure 8–1:

(1) Column input buffers are true LVDS buffers, but do not support 100-Ω differential on-chip termination.

(2) Column output buffers are single ended and need external termination schemes to support LVDS, mini-LVDS and RSDS standards. Refer to the *I/O Features in Stratix IV Devices* chapter in the *Stratix IV Device Handbook* for more details.

(3) Row input buffers are true LVDS buffers, and support the 100-Ω differential on-chip termination resistors required for LVDS signals.

The ALTLVDS transmitter/receiver requires various clock and load enable signals from a left/right PLL. The Quartus® II software provides the following two choices when configuring the LVDS SERDES circuitry with respect to the PLL usage:

- LVDS Interface with **Use External PLL** option enabled

You can control the PLL settings such as dynamically reconfiguring the PLL to support different data rates, dynamic phase shift, and so on. In this case, the **Use External PLL** option in the ALTLVDS megafunction should be enabled, using the ALTLVDS MegaWizard® Plug-in Manager. You also must instantiate an ALTPLL megafunction to generate the various clocks and load enable signals. Refer to the section "Stratix IV LVDS Interface with 'Use External PLL' Option Enabled" on page 8–24 for more information.

- LVDS Interface with **Use External PLL** Option Disabled

  The alternate choice is to disable the option **Use External PLL**. In this case, the Quartus II software configures the PLL settings automatically. The software also takes care of generating the various clock and load enable signals based on the input reference clock and data rate selected.

☞ Both choices target the same physical PLL; the only difference is the additional flexibility provided when an LVDS interface has the **Use External PLL** option enabled.

This chapter contains the following sections:

- "Locations of I/O Banks" on page 8–3
- "Stratix IV LVDS Channels" on page 8–5
- "Stratix IV LVDS SERDES Block Diagram" on page 8–7
- "Stratix IV ALTLVDS Port List" on page 8–8
- "Stratix IV Differential Transmitter" on page 8–11
- "Stratix IV Differential Receiver" on page 8–15
- "Stratix IV LVDS Interface with 'Use External PLL' Option Enabled" on page 8–24
- "Stratix IV Left/Right PLLs (PLL_Lx/ PLL_Rx)" on page 8–26
- "Stratix IV Clocking" on page 8–26
- "Stratix IV Source-Synchronous Timing Budget" on page 8–27
- "Differential Pin Placement Guidelines" on page 8–34

Locations of I/O Banks

The Stratix IV family I/Os are divided into 16 to 24 I/O banks. The dedicated circuitry that supports high-speed differential I/Os is located in banks in the right and left side of the device. Figure 8–2 shows a high-level chip overview of the Stratix IV E device.

Figure 8–2. High-Speed Differential I/Os with DPA Locations in Stratix IV E Devices

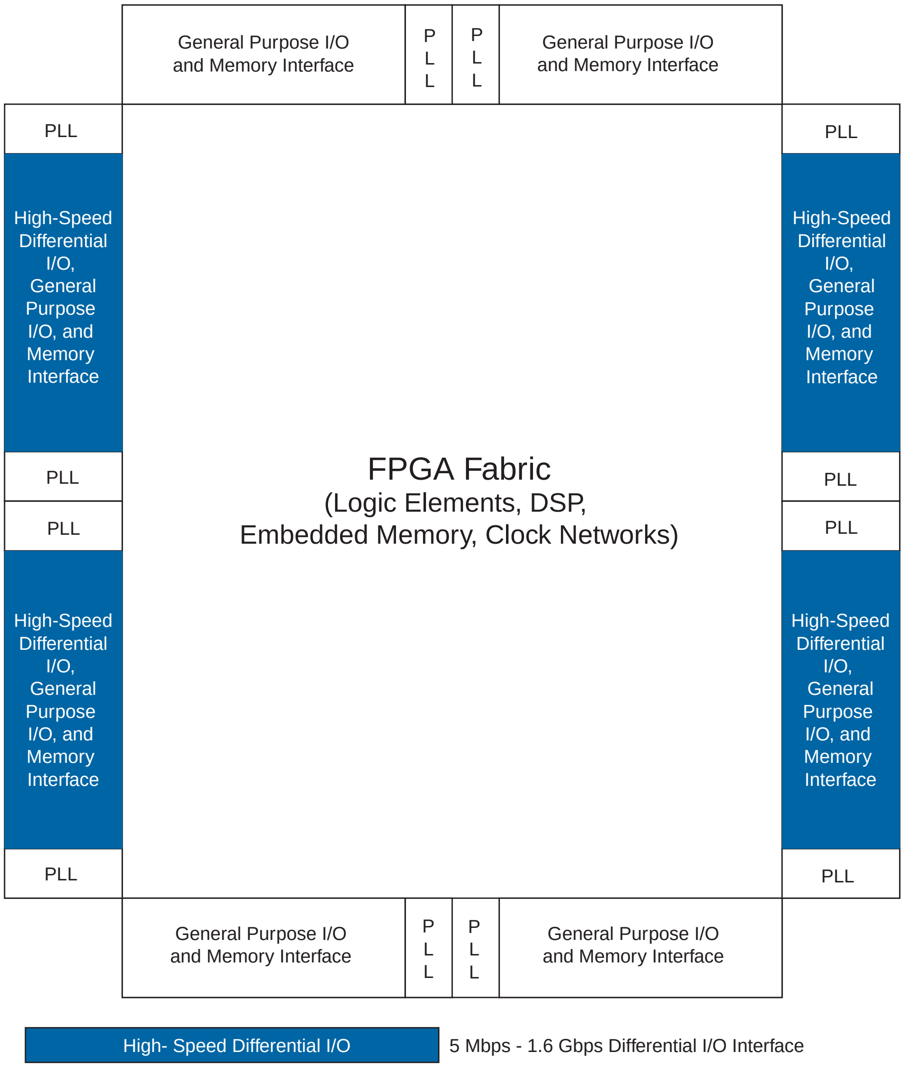
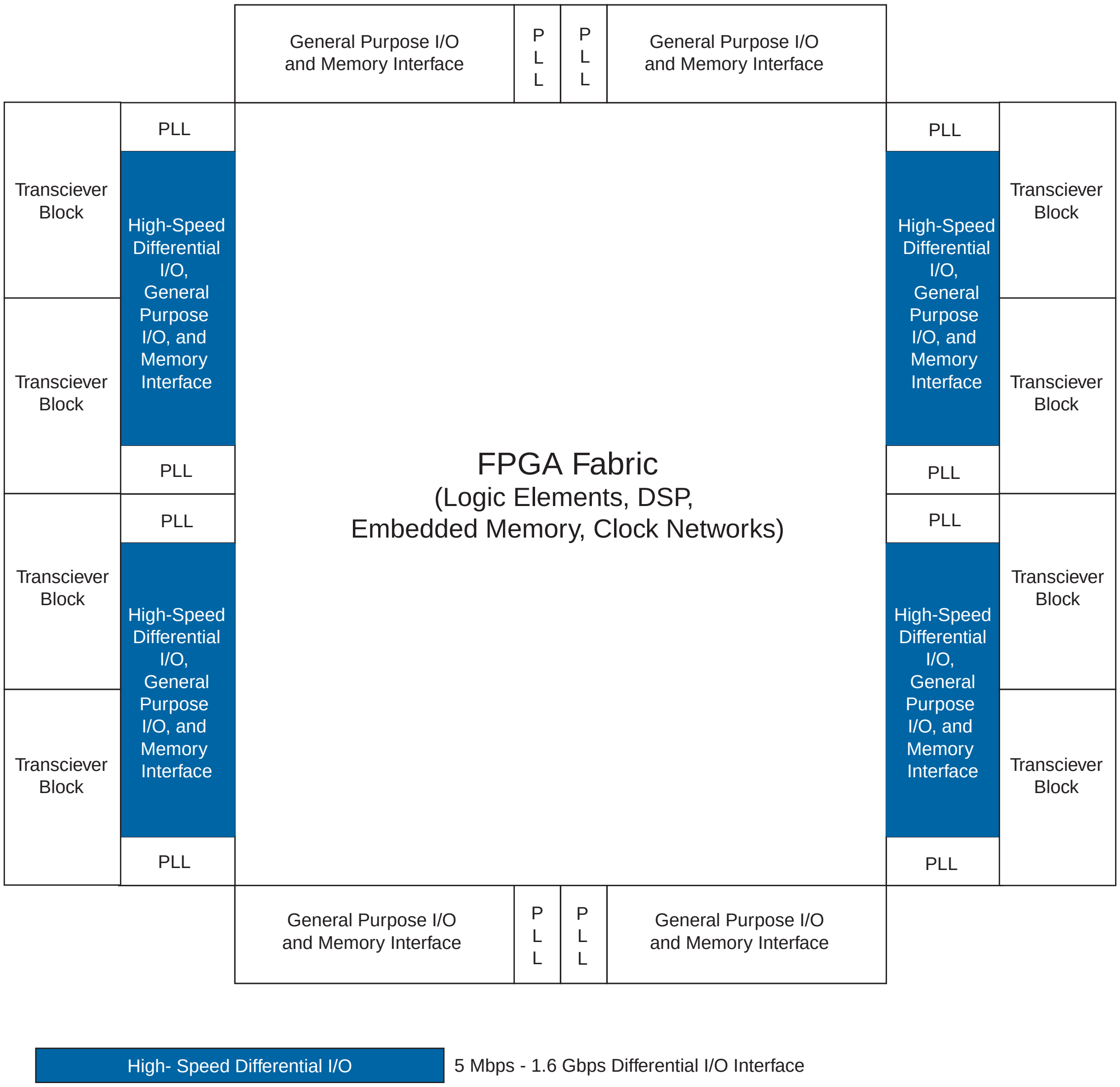

General Purpose I/O and Memory Interface
P L L
P L L
General Purpose I/O and Memory Interface
PLL
High-Speed Differential I/O, General Purpose I/O, and Memory Interface
PLL
PLL
High-Speed Differential I/O, General Purpose I/O, and Memory Interface
PLL
PLL
High-Speed Differential I/O, General Purpose I/O, and Memory Interface
PLL
PLL
High-Speed Differential I/O, General Purpose I/O, and Memory Interface
PLL
FPGA Fabric
(Logic Elements, DSP, Embedded Memory, Clock Networks)
General Purpose I/O and Memory Interface
P L L
P L L
General Purpose I/O and Memory Interface
High- Speed Differential I/O
5 Mbps - 1.6 Gbps Differential I/O Interface

Figure 8–3 shows a high-level chip overview of the Stratix IV GX device.

Figure 8–3. High-Speed Differential I/Os with DPA Locations in Stratix IV GX Devices


General Purpose I/O and Memory Interface
PLL
PLL
General Purpose I/O and Memory Interface
Transciever Block
Transciever Block
Transciever Block
Transciever Block
PLL
High-Speed Differential I/O, General Purpose I/O, and Memory Interface
PLL
PLL
High-Speed Differential I/O, General Purpose I/O, and Memory Interface
PLL
FPGA Fabric
(Logic Elements, DSP, Embedded Memory, Clock Networks)
PLL
High-Speed Differential I/O, General Purpose I/O, and Memory Interface
PLL
PLL
High-Speed Differential I/O, General Purpose I/O, and Memory Interface
PLL
Transciever Block
Transciever Block
Transciever Block
Transciever Block
General Purpose I/O and Memory Interface
PLL
PLL
General Purpose I/O and Memory Interface
High- Speed Differential I/O
5 Mbps - 1.6 Gbps Differential I/O Interface

Stratix IV LVDS Channels

The Stratix IV family supports LVDS on both row I/O banks and column I/O banks. Row I/Os are true input/output LVDS buffers, and provide dedicated SERDES circuitry. Column input buffers are true LVDS buffers. Column output buffers are single-ended, and need external termination schemes to support LVDS, mini-LVDS, and RSDS standards. Table 8–2 and Table 8–3 show the total number of row and column LVDS I/Os supported in Stratix IV E devices.

Table 8–2. LVDS Channels Supported in Stratix IV E Device Row I/O Banks

Device	780 - Pin FineLine BGA	1152 - Pin FineLine BGA	1517 - Pin FineLine BGA	1760 - Pin FineLine BGA
EP4SE110	56Rx + 56Tx	—	—	—
EP4SE230	56Rx + 56Tx	—	—	—

Table 8–2. LVDS Channels Supported in Stratix IV E Device Row I/O Banks

Device	780 - Pin FineLine BGA	1152 - Pin FineLine BGA	1517 - Pin FineLine BGA	1760 - Pin FineLine BGA
EP4SE290	56Rx + 56Tx	88Rx + 88Tx	88Rx + 88Tx	—
EP4SE360	56Rx + 56Tx	88Rx + 88Tx	88Rx + 88Tx	—
EP4SE530	—	88Rx + 88Tx	112Rx + 112Tx	112Rx + 112Tx
EP4SE680	—	88Rx + 88Tx	112Rx + 112Tx	132Rx + 132Tx

Table 8–3. LVDS Channels Supported in Stratix IV E Device Column I/O Banks *(Note 1)*

Device	780 - Pin FineLine BGA	1152 - Pin FineLine BGA	1517 - Pin FineLine BGA	1760 - Pin FineLine BGA
EP4SE110	64Rx + 64Tx	—	—	—
EP4SE230	64Rx + 64Tx	—	—	—
EP4SE290	64Rx + 64Tx	96Rx + 96Tx	128Rx + 128Tx	—
EP4SE360	64Rx + 64Tx	96Rx + 96Tx	128Rx + 128Tx	—
EP4SE530	—	96Rx + 96Tx	128Rx + 128Tx	128Rx + 128Tx
EP4SE680	—	96Rx + 96Tx	128Rx + 128Tx	144Rx + 144Tx

Note to Table 8–3:

(1) Column input buffers are true LVDS buffers. Column output buffers are single-ended, and need external termination schemes to support LVDS, mini-LVDS, and RSDS standards.

Table 8–4 and Table 8–5 show the total number of row and column LVDS I/Os supported in Stratix IV GX devices.

Table 8–4. LVDS Channels Supported in Stratix IV GX Device Row I/O Banks

Device	780 - Pin FineLine BGA	1152 - Pin FineLine BGA	1517 - Pin FineLine BGA	1932- Pin FineLine BGA
EP4SGX70	28Rx + 28Tx	—	—	—
EP4SGX110	28Rx + 28Tx	28Rx + 28Tx	—	—
EP4SGX230	28Rx + 28Tx	44Rx + 44Tx	88Rx + 88Tx	—
EP4SGX290	0	44Rx + 44Tx	88Rx + 88Tx	—
EP4SGX360	0	44Rx + 44Tx	88Rx + 88Tx	—
EP4SGX530	—	44Rx + 44Tx	88Rx + 88Tx	98Rx + 98Tx

Table 8–5. LVDS Channels Supported in Stratix IV GX Device Column I/O Banks (Part 1 of 2) *(Note 1)*

Device	780 - Pin FineLine BGA	1152 - Pin FineLine BGA	1517 - Pin FineLine BGA	1932 - Pin FineLine BGA
EP4SGX70	64Rx + 64Tx	—	—	—
EP4SGX110	64Rx + 64Tx	64Rx + 64Tx	—	—
EP4SGX230	64Rx + 64Tx	96Rx + 96Tx	96Rx + 96Tx	—
EP4SGX290	72Rx + 72Tx	96Rx + 96Tx	96Rx + 96Tx	—

Table 8–5. LVDS Channels Supported in Stratix IV GX Device Column I/O Banks (Part 2 of 2) *(Note 1)*

Device	780 - Pin FineLine BGA	1152 - Pin FineLine BGA	1517 - Pin FineLine BGA	1932 - Pin FineLine BGA
EP4SGX360	72Rx + 72Tx	96Rx + 96Tx	96Rx + 96Tx	—
EP4SGX530	—	96Rx + 96Tx	96Rx + 96Tx	128Rx + 128Tx

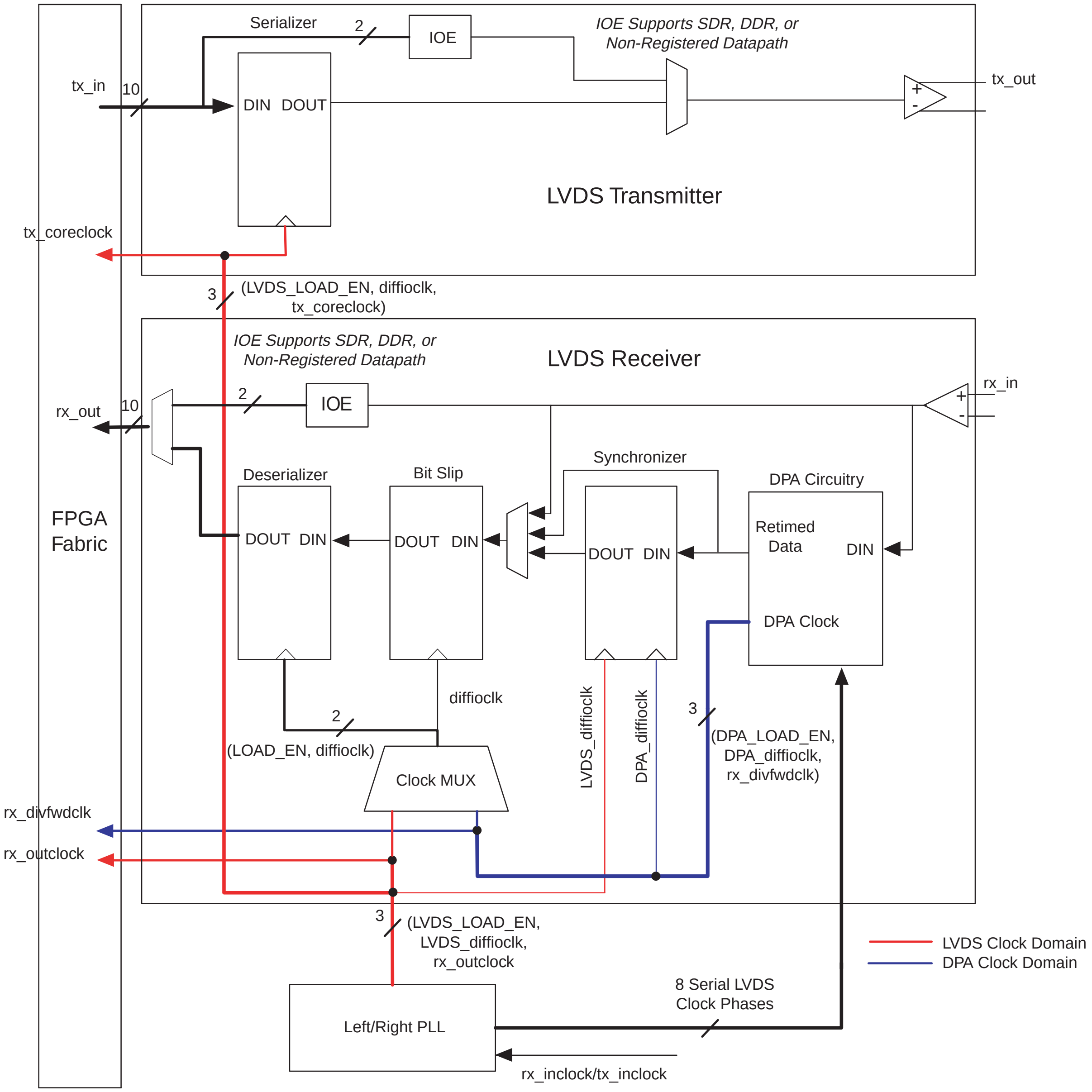
Note to Table 8–5:

(1) Column input buffers are true LVDS buffers. Column output buffers are single-ended, and need external termination schemes to support LVDS, mini-LVDS, and RSDS standards.

Stratix IV LVDS SERDES Block Diagram

Figure 8–4 shows a transmitter and receiver block diagram for LVDS SERDES circuitry in the left and right banks. This diagram shows the interface signals of the transmitter and receiver data path. For more details, refer to the sections "Stratix IV Differential Transmitter" on page 8–11 and "Stratix IV Differential Receiver" on page 8–15.

Figure 8–4. LVDS SERDES Block Diagram *(Note 1)*, *(2)*, *(3)*



Note to Figure 8–4:

(1) This diagram shows a shared PLL between the transmitter and receiver. If the transmitter and receiver are not sharing the same PLL, the two left/right PLLs are required.

(2) In SDR mode and DDR mode, the data width is 1 and 2, respectively.

(3) The `tx_in` and `rx_out` ports have a maximum data width of 10.

Stratix IV ALTLVDS Port List

Table 8–6 shows the interface signals for an LVDS transmitter and receiver.

Table 8–6. Port List of the LVDS Interface (ALTLVDS) *(Note 1)*, *(2)*, *(3)* (Part 1 of 3)

Port Name	Input / Output	Description
PLL Signals		
`pll_areset`	Input	Asynchronous reset to LVDS transmitter/receiver PLL. Minimum pulse width requirement for this signal is 10 ns.
LVDS Transmitter Interface Signals		
`tx_in[ ]`	Input	The data bus width per channel is the same as the SERIALIZATION FACTOR (SF). The input data must be synchronous to the `tx_coreclock` signal.
`tx_inclock`	Input	Reference clock input for the transmitter PLL. The allowed frequency range for the reference clock is between 5 MHz and 625 MHz and must satisfy the following condition: 5 Mbps < ((Input Clock Frequency) x (PLL Multiplication Factor)) < 1600 Mbps. The ALTLVDS MegaWizard Plug-In Manager automatically selects the appropriate PLL multiplication factor based on the data rate and reference clock frequency selection.
`tx_enable` *(3)*	Input	This port is instantiated only when you select the **Use External PLL** option in the MegaWizard Plug-In Manager. This input port must be driven by the PLL instantiated though the ALTPLL MegaWizard Plug-In Manager.
`tx_out`	Output	LVDS Transmitter serial data output port. `tx_out` is clocked by a serial clock generated by the left/right PLL.
`tx_outclock`	Output	The frequency of this clock is programmable to be the same as the data rate (up to 717 MHz), half the data rate, or one-fourth the data rate. The phase offset of this clock with respect to the serial data is programmable in increments of 45°.
`tx_coreclock` *(3)*	Output	FPGA fabric-transmitter interface clock. The parallel transmitter data generated in the FPGA fabric should be clocked with this clock. This port is not available when you select the **Use External PLL** option in the MegaWizard Plug-In Manager. The FPGA fabric-transmitter interface clock must be driven by the PLL instantiated through the ALTPLL MegaWizard Plug-In Manager.
`tx_locked`	Output	When `HIGH`, it indicates that the transmitter PLL is locked to the input reference clock.
LVDS Receiever Interface Signals		
`rx_in`	Input	LVDS Receiver serial data input port.
`rx_inclock`	Input	Reference clock input for the receiver PLL. The allowed frequency range for the reference clock is between 5 MHz and 625 MHz and must satisfy the following condition: 5 Mbps < ((Input Clock Frequency) x (PLL Multiplication Factor)) < 1600 Mbps. The ALTLVDS MegaWizard Plug-In Manager automatically selects the appropriate PLL multiplication factor based on the data rate and reference clock frequency selection.

Table 8–6. Port List of the LVDS Interface (ALTLVDS) *(Note 1)*, *(2)*, *(3)* (Part 2 of 3)

Port Name	Input / Output	Description
`rx_channel_data_align`	Input	Edge-sensitive bit slip control signal. Each rising edge on this signal causes the data re-alignment circuitry to shift the word boundary by one bit. The minimum pulse width requirement is one parallel clock cycle. There is no maximum pulse width requirement.
`rx_dpll_enable`	Input	In DPA mode, this signal dynamically enables or disables the DPA circuitry. When `HIGH`, this signal enables the DPA circuitry. When `LOW`, this signal disables the DPA circuitry. This port is not available in non-DPA and soft-CDR modes.
`rx_dpll_hold`	Input	When `LOW`, the DPA tracks any dynamic phase variations between the clock and data. When `HIGH`, the DPA holds the last locked phase and does not track any dynamic phase variations between the clock and data. This port is not available in non-DPA mode.
`rx_enable` *(3)*	Input	This port is instantiated only when you select the **Use External PLL** option in the MegaWizard Plug-In Manager. This input port must be driven by the PLL instantiated though the ALTPLL MegaWizard Plug-In Manager.
`rx_out[ ]`	Output	Receiver parallel data output. The data bus width per channel is same as the DESERIALIZATION FACTOR (DF). The output data is synchronous to the `rx_outclock` signal in non-DPA and DPA modes. It is synchronous to the `rx_divfwdclk` signal in soft-CDR mode.
`rx_outclock`	Output	Parallel output clock from the receiver PLL. The parallel data output from the receiver is synchronous to this clock in non-DPA and DPA modes. This port is not available when you select the **Use External PLL** option in the MegaWizard Plug-In Manager. The FPGA fabric-receiver interface clock must be driven by the PLL instantiated through the ALTPLL MegaWizard Plug-In Manager.
`rx_locked`	Output	When `HIGH`, it indicates that the receiver PLL is locked to `rx_inclock`.
`rx_dpa_locked`	Output	This signal only indicates an initial DPA lock condition to the optimum phase after power up or reset. This signal does not get de-asserted if the DPA selects a new phase out of the eight clock phases to sample the received data. You must not use the `rx_dpa_locked` signal to determine a DPA loss of lock condition.
`rx_cda_max`	Output	Data re-alignment (Bit Slip) roll-over signal. When `HIGH` for one parallel clock cycle, it indicates that the user-programmed number of bits for the word boundary to roll-over have been slipped.
`rx_divfwdclk`	Output	Parallel DPA clock to the FPGA fabric logic array. The parallel receiver output data to the FPGA fabric logic array is synchronous to this clock in soft-CDR mode. This signal is not available in non-DPA and DPA modes.
Reset Signals		
`rx_reset`	Input	Asynchronous reset to the DPA circuitry and FIFO. The minimum pulse width requirement for this reset is one parallel clock cycle. This signal resets DPA and FIFO blocks.
`rx_fifo_reset`	Input	Asynchronous reset to the FIFO between DPA and data realignment circuits. The synchronizer block should be reset after DPA loses lock condition, and data checker shows corrupted received data. The minimum pulse width requirement for this reset is one parallel clock cycle. This signal resets FIFO block.

Table 8–6. Port List of the LVDS Interface (ALTLVDS) *(Note 1)*, *(2)*, *(3)* (Part 3 of 3)

Port Name	Input / Output	Description
`rx_cda_reset`	Input	Asynchronous reset to the data realignment circuitry. The minimum pulse width requirement for this reset is one parallel clock cycle. This signal resets data realignment block.

Note to Table 8–6:

(1) Unless stated, signals are valid in all three modes (non-DPA, DPA, and soft-CDR) for a single channel.

(2) All reset and control signals are active high.

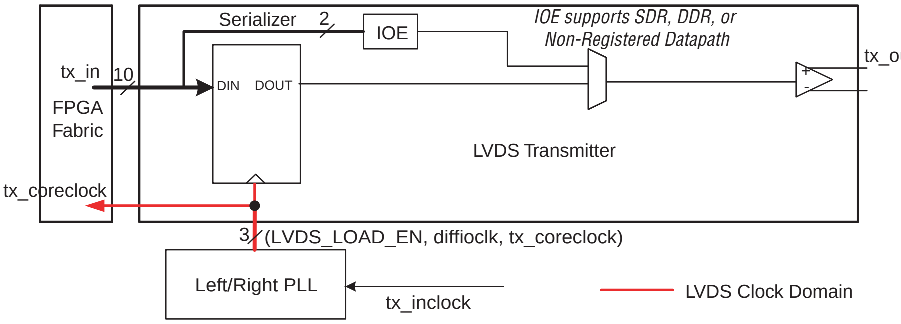
(3) Refer to the "Stratix IV LVDS Interface with 'Use External PLL' Option Enabled" section.

Stratix IV Differential Transmitter

The Stratix IV transmitter has dedicated circuitry to provide support for LVDS signaling. The dedicated circuitry consists of a differential buffer, a serializer, and left/right PLLs, which can be shared between the transmitter and receiver. The differential buffer can drive out LVDS, mini-LVDS, and RSDS signaling levels. The serializer takes up to 10-bits wide parallel data from the FPGA fabric, clocks it into the load registers, and serializes it using shift registers clocked by the left/right PLL before sending the data to the differential buffer. The most significant bit (MSB) of the parallel data is transmitted first.

The load and shift registers are clocked by the load enable (`LVDS_LOAD_EN`) signal and the `diffioclk` (clock running at serial data rate) signal generated from PLL_Lx (left PLL) or PLL_Rx (right PLL). The serialization factor can be statically set to ×4, ×6, ×7, ×8, or ×10 using the Quartus II software. The load enable signal is derived from the serialization factor setting. Figure 8–5 shows a block diagram of the Stratix IV transmitter.

Figure 8–5. *Stratix IV Transmitter Block Diagram* *(Note 1)*, *(2)*



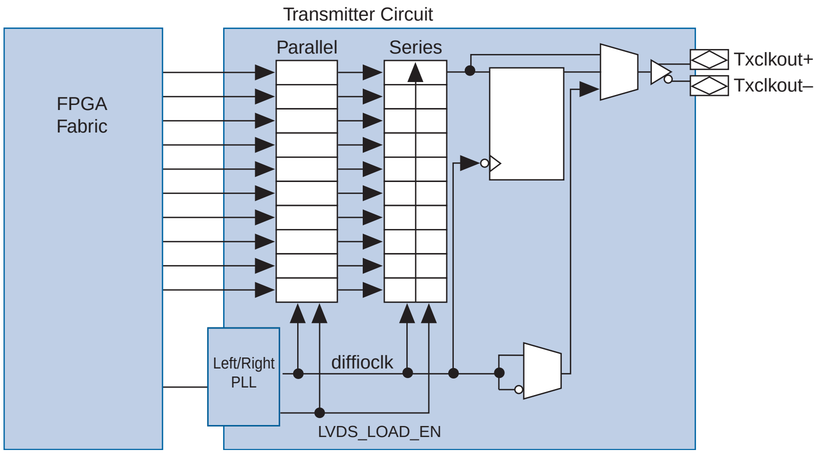
Notes to Figure 8–5:

(1) In SDR mode and DDR mode, the data width is 1 and 2, respectively.

(2) The `tx_in` port has a maximum data width of 10.

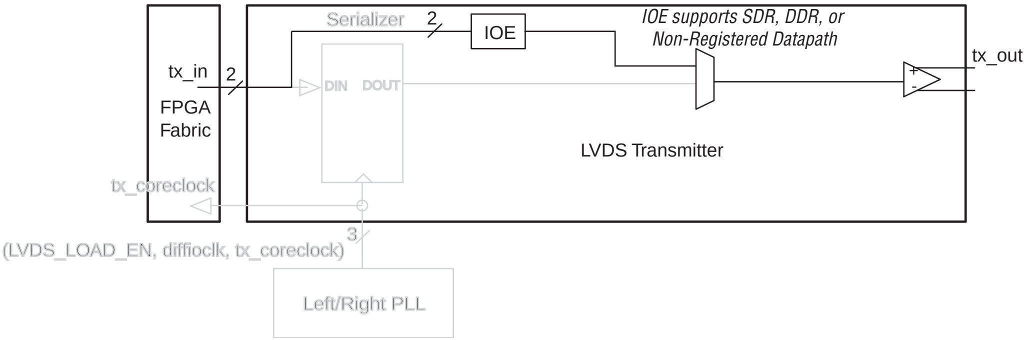
Any Stratix IV transmitter data channel can be configured to generate a source-synchronous transmitter clock output. This flexibility allows the placement of the output clock near the data outputs to simplify board layout and reduce clock-to-data skew. Different applications often require specific clock-to-data alignments or specific data rate to clock rate factors. The transmitter can output a clock signal at the same rate as the data with a maximum frequency of 717 MHz. The output clock can also be divided by a factor of 1, 2, 4, 6, 8, or 10, depending on the serialization factor. The phase of the clock in relation to the data can be set at 0° or 180° (edge or center aligned). The left and right PLLs (PLL_Lx/PLL_Rx) provide additional support for other phase shifts in 45° increments. These settings are made statically in the Quartus II MegaWizard Plug-In Manager software. Figure 8–6 shows the Stratix IV transmitter in clock output mode. In clock output mode, an LVDS channel can be used as a clock output channel.

Figure 8–6. Stratix IV Transmitter in Clock Output Mode



The Stratix IV serializer can be by passed to support DDR (×2) and SDR (×1) operations to achieve a serialization factor of 2 and 1, respectively. The I/O element (IOE) contains two data output registers that can each operate in either DDR or SDR mode. Figure 8–7 shows the serializer bypass path.

Figure 8–7. *Stratix IV Serializer Bypass* *(Note 1), (2), (3)*



Note to Figure 8–7:

(1) All disabled blocks and signals are grayed out.

(2) In DDR mode, `tx_inclock` clocks the IOE register. In SDR mode, data is directly passed through IOE.

(3) In SDR mode and DDR mode, the data width to IOE is 1 and 2, respectively.

Programmable Pre-Emphasis and Programmable V_{OD}

Stratix IV LVDS transmitters support programmable pre-emphasis and programmable V_{OD}. Pre-emphasis increases the amplitude of the high frequency component of the output signal, and thus helps to compensate for the frequency dependent attenuation along the transmission line. Figure 8–8 shows the differential LVDS output. Figure 8–9 illustrates the LVDS output with pre-emphasis.

Figure 8–8. Differential V_{OD}

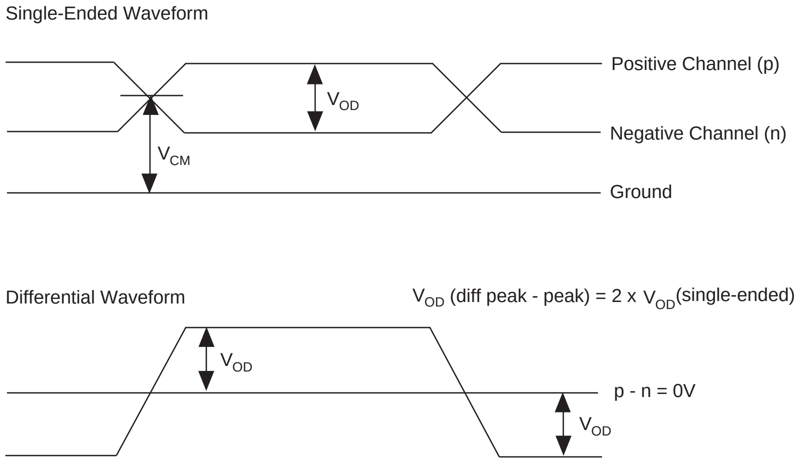
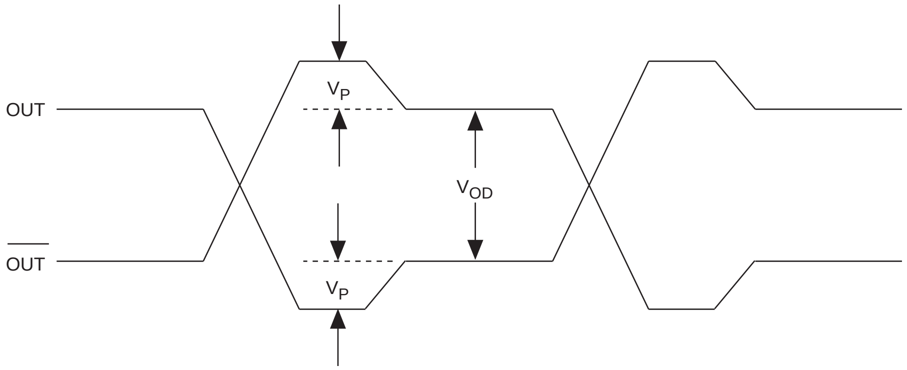


Figure 8–9. Programmable Pre-Emphasis *(Note 1)*



Note to Figure 8–9:

(1) V_P— voltage boost from pre-emphasis. V_{OD}— Differential output voltage (peak-peak).

Pre-emphasis is an important feature for high-speed transmission. Without pre-emphasis, the output current is limited by the V_{OD} setting and the output impedance of the driver. At high frequency, the slew rate may not be fast enough to reach the full V_{OD} before the next edge, producing a pattern dependent jitter.

With pre-emphasis, the output current is boosted momentarily during switching to increase the output slew rate. The overshoot introduced by the extra current happens only during switching and does not ring, unlike the overshoot caused by signal reflection. The amount of pre-emphasis needed depends on the attenuation of the high-frequency component along the transmission line.

Programmable VOD

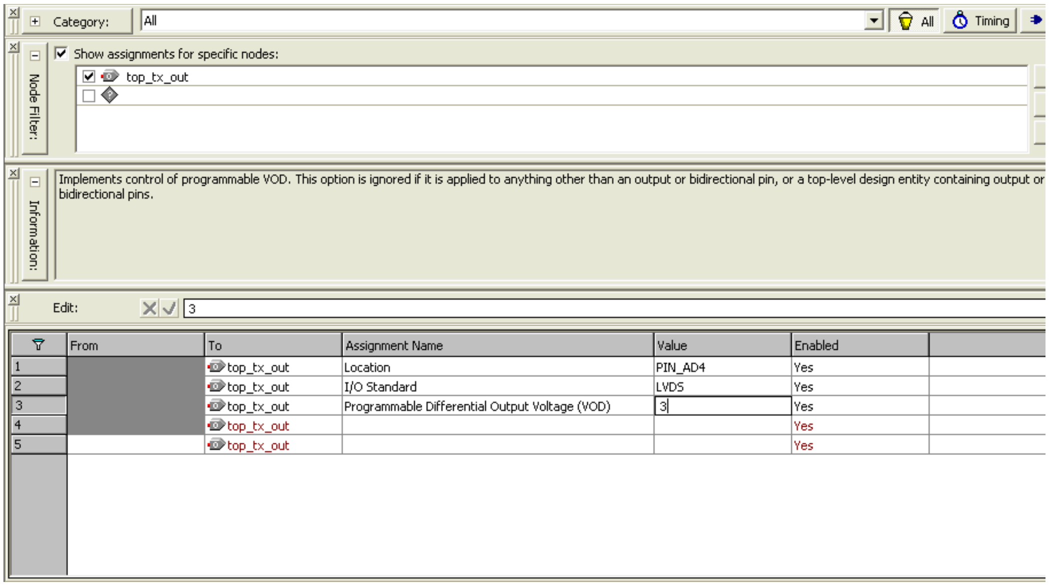
The V_{OD} settings can be statically assigned from the Assignment Editor. Table 8–7 shows the assignment name for programmable V_{OD} and its possible values in Quartus II software Assignment Editor.

Table 8–7. Quartus II Software Assignment Editor

To	`tx_out`
Assignment name	Programmable Differential Output Voltage (V_{OD})
Allowed values	0, 1, 2, 3

Figure 8–10 shows assignment of programmable V_{OD} for a transmit data output from Quartus II software Assignment Editor.

Figure 8–10. Quartus II Software Assignment Editor – Programmable V_{OD}



☑ Show assignments for specific nodes:

☑ top_tx_out
□

Implements control of programmable VOD. This option is ignored if it is applied to anything other than an output or bidirectional pin, or a top-level design entity containing output or bidirectional pins.

Edit: 3

	From	To	Assignment Name	Value	Enabled
1		top_tx_out	Location	PIN_AD4	Yes
2		top_tx_out	I/O Standard	LVDS	Yes
3		top_tx_out	Programmable Differential Output Voltage (VOD)	3	Yes
4		top_tx_out			Yes
5		top_tx_out			Yes

Table 8–8 shows four possible settings and their corresponding V_{OD} values.

Table 8–8. V_{OD} Settings for LVDS Channels *(Note 1)*, *(2)*

V_{OD}	Quartus II Assignment Editor V_{OD} Settings			
	0	**1**	**2**	**3**
V_{OD} (mV) (Single-Ended (P-P))	280	370	420	500

Notes to Table 8–8:

(1) The default V_{OD} setting is **1**.

(2) Table 8–8 shows preliminary values for differential output voltage (V_{OD}). V_{OD} values are pending characterization.

Programmable Pre-Emphasis

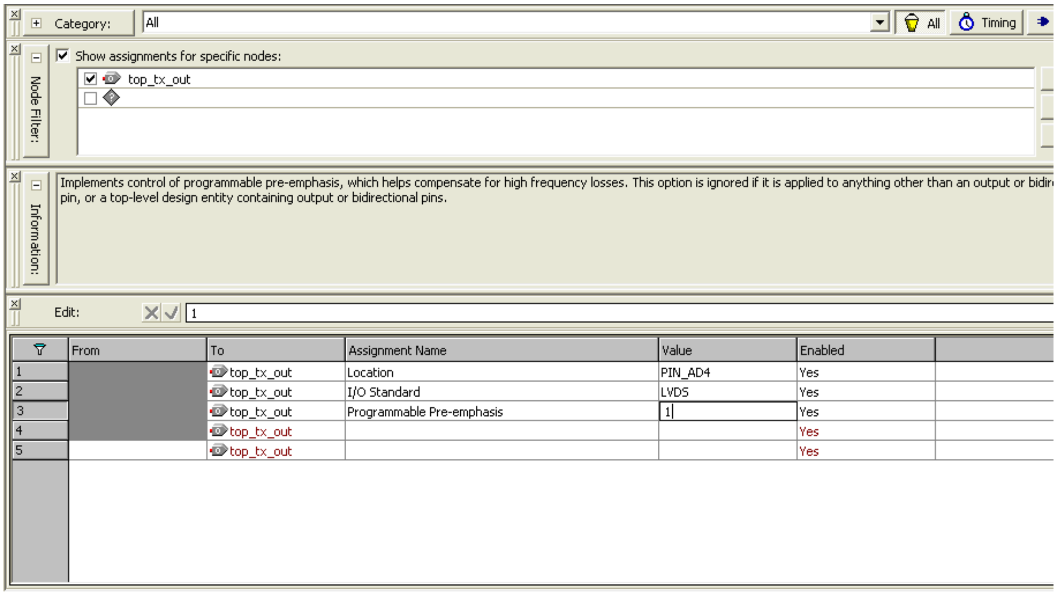
Four different settings are allowed for pre-emphasis from Assignment Editor for each LVDS output channel. Table 8–9 shows the assignment name and its possible values for programmable pre-emphasis in the Quartus II software Assignment Editor.

Table 8–9. Quartus II Software Assignment Editor

To	`tx_out`
Assignment name	Programmable Pre-emphasis
Allowed values	0,1,2,3

Figure 8–11 shows assignment of programmable pre-emphasis for a transmit data output port from Quartus II software Assignment Editor.

Figure 8–11. Quartus II Software Assignment Editor – Programmable Pre-Emphasis



Category: All

☑ Show assignments for specific nodes:

☑ top_tx_out

Implements control of programmable pre-emphasis, which helps compensate for high frequency losses. This option is ignored if it is applied to anything other than an output or bidir
pin, or a top-level design entity containing output or bidirectional pins.

Edit: 1

	From	To	Assignment Name	Value	Enabled
1		top_tx_out	Location	PIN_AD4	Yes
2		top_tx_out	I/O Standard	LVDS	Yes
3		top_tx_out	Programmable Pre-emphasis	1	Yes
4		top_tx_out			Yes
5		top_tx_out			Yes

Stratix IV Differential Receiver

The Stratix IV family has dedicated circuitry to receive high-speed differential signals in row I/Os. Figure 8–12 shows the hardware blocks of the Stratix IV receiver. The receiver has a differential buffer, left/right PLLs which can be shared between the transmitter and receiver, a dynamic phase alignment (DPA) block, a synchronizer, a data realignment block, and a deserializer. The differential buffer can receive LVDS, mini-LVDS, and RSDS signal levels, which are statically set in the Quartus II software Assignment Editor.

The left/right PLL receives the external clock input and generates different phases of the same clock. The DPA block chooses one of the clocks from the left/right PLL and aligns the incoming data on each channel. The synchronizer circuit is a 1-bit wide by 6-bit deep FIFO buffer that compensates for any phase difference between the DPA clock and the data realignment block. If necessary, the user-controlled data realignment circuitry inserts a single bit of latency in the serial bit stream to align to the word boundary. The deserializer includes shift registers and parallel load registers, and sends a maximum of 10 bits to the internal logic.

The Stratix IV family supports three different receiver modes:

- Non-DPA mode

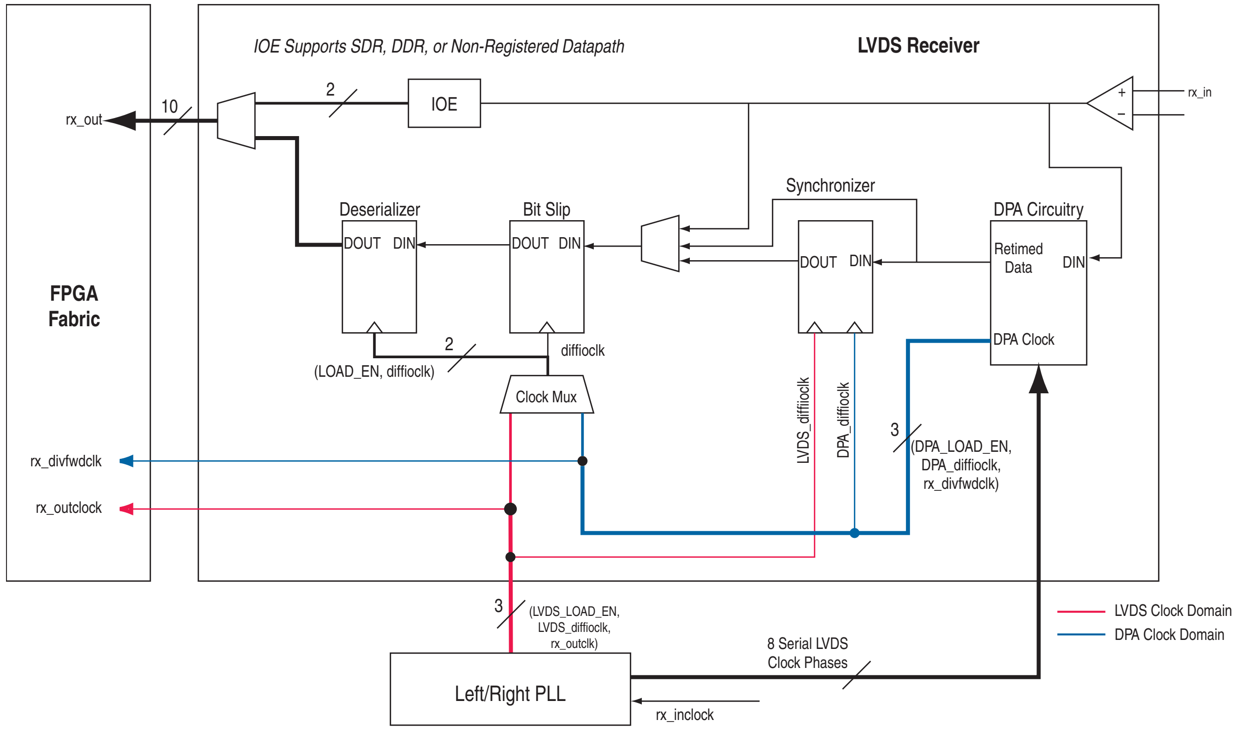
- DPA mode
- Soft-CDR mode

The physical medium connecting the transmitter and the receiver LVDS channels may introduce skew between the serial data and the source-synchronous clock. The instantaneous skew between each LVDS channel and the clock also varies with the jitter on the data and clock signals as seen by the receiver. The three different modes – non-DPA, DPA, and soft-CDR – provide different options to overcome skew between source synchronous clock (non-DPA, DPA) / reference clock (soft-CDR), and the serial data.

Only non-DPA mode needs manual adjustment of skew performed by the user.

The non-DPA mode allows you to statically select the optimal phase between the source synchronous clock and received serial data to compensate skew. In DPA mode, DPA circuitry automatically chooses the best phase to compensate for the skew between the source synchronous clock and received serial data. Soft-CDR mode provides opportunities for synchronous/asynchronous applications for chip-to-chip and short reach board-to-board applications for SGMII protocols.

Figure 8–12. Receiver Block Diagram *(Note 1)*, *(2)*



Note to Figure 8–12:

(1) In SDR mode and DDR mode, the data width from IOE is 1 and 2, respectively.

(2) The `rx_out` port has a maximum data width of 10.

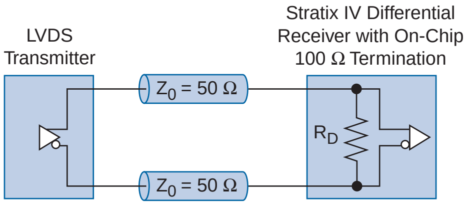
Differential I/O Termination

The Stratix IV family provides a 100-Ω, on-chip differential termination option on each differential receiver channel for LVDS standards. On-chip termination saves board space by eliminating the need to add external resistors on the board. You can enable on-chip termination in the Quartus II software Assignment Editor.

On-chip differential termination is supported on all row I/O pins and SERDES block clock pins: `CLK (0, 2, 9,` and `11)`. It is not supported for column I/O pins, high speed clock pins `CLK [1, 3, 8, 10]`, or the corner PLL clock inputs.

Figure 8–13 illustrates device on-chip termination.

Figure 8–13. On-Chip Differential I/O Termination



Receiver Hardware Blocks

The differential receiver has the following hardware blocks:

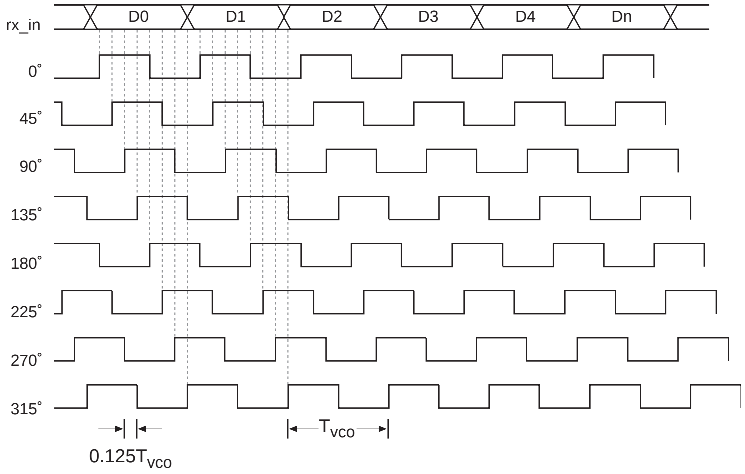
- Dynamic phase alignment (DPA) block
- Synchronizer
- Data realignment (Bit slip)
- Deserializer

Dynamic Phase Alignment (DPA) Block

The DPA block takes in high-speed serial data from the differential input buffer and selects one of the eight phases generated by the left/right PLL to sample the data. The DPA chooses a phase closest to the phase of the serial data. The maximum phase offset between the received data and the selected phase is 1/8 UI, which is the maximum quantization error of the DPA. The eight phases of the clock are equally divided, giving a 45° resolution.

Figure 8–14 shows the possible phase relationships between the DPA clocks and the incoming serial data.

Figure 8–14. DPA Clock Phase to Serial Data Timing Relationship *(Note 1)*



Note to Figure 8–14:

(1) T_{VCO} is defined as the PLL serial clock period.

The DPA block continuously monitors the phase of the incoming serial data and selects a new clock phase if needed. You can prevent the DPA from selecting a new clock phase by asserting the optional `RX_DPLL_HOLD` port, which is available for each channel.

The DPA block requires a training pattern and a training sequence of at least 256 repetitions. The training pattern is not fixed, so you can use any training pattern with at least one transition. An optional output port, `RX_DPA_LOCKED`, is available to indicate an initial DPA lock condition to the optimum phase after power up or reset. This signal is not de-asserted if the DPA selects a new phase out of the eight clock phases to sample the received data. Do not use the `rx_dpa_locked` signal to determine a DPA loss of lock condition. Use data checkers such as a cyclic redundancy check (CRC) or diagonal interleaved parity (DIP-4) to validate the data.

An independent reset port, `RX_RESET`, is available to reset the DPA circuitry. The DPA circuitry must be retrained after reset.

☞ DPA block is bypassed in non-DPA mode.

Synchronizer

The synchronizer is a 1-bit wide and 6-bit deep FIFO buffer that compensates for the phase difference between `DPA_diffioclk`, which is the optimal clock selected by the DPA block, and `LVDS_diffioclk`, which is produced by left/right PLL. The synchronizer can only compensate for phase differences, not frequency differences between the data and the receiver's input reference clock.

An optional port, `RX_FIFO_RESET`, is available to the internal logic to reset the synchronizer. The synchronizer is automatically reset when the DPA first locks to the incoming data. Altera® recommends using `RX_FIFO_RESET` to reset the synchronizer when the DPA signals a loss-of-lock condition, and the data checker indicates corrupted received data.

The synchronizer circuit is bypassed in non-DPA and soft-CDR mode.

Data Realignment Block (Bit Slip)

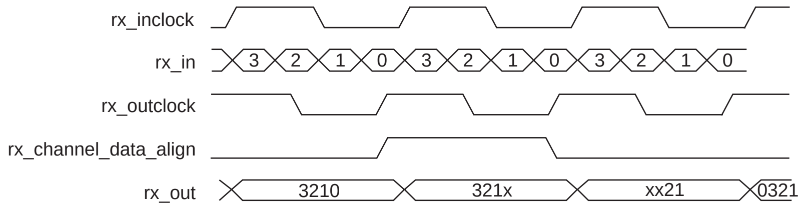
Skew in the transmitted data along with skew added by the link causes channel-to-channel skew on the received serial data streams. If the DPA is enabled, the received data is captured with different clock phases on each channel. This may cause the received data to be misaligned from channel-to-channel. To compensate for this channel-to-channel skew and establish the correct received word boundary at each channel, each receiver channel has a dedicated data realignment circuit that realigns the data by inserting bit latencies into the serial stream.

An optional RX_CHANNEL_DATA_ALIGN port controls the bit insertion of each receiver independently controlled from the internal logic. The data slips one bit on the rising edge of the RX_CHANNEL_DATA_ALIGN. The requirements for the RX_CHANNEL_DATA_ALIGN signal include:

- The minimum pulse width is one period of the parallel clock in the logic array
- The minimum low time between pulses is one period of the parallel clock
- This is an edge triggered signal
- Valid data is available two parallel clock cycles after the rising edge of RX_CHANNEL_DATA_ALIGN

Figure 8–15 shows receiver output (RX_OUT) after one bit slip pulse with the deserialization factor set to **4**.

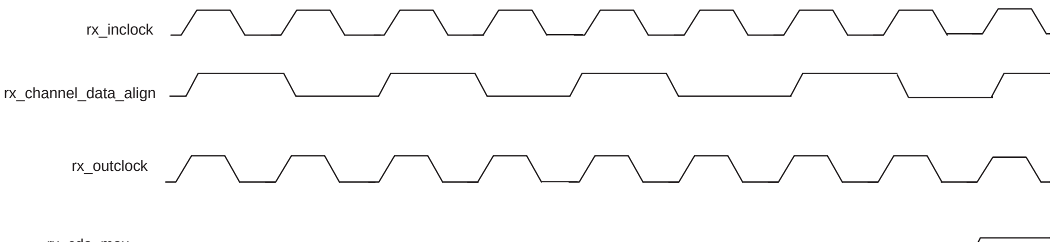
Figure 8–15. Data Realignment Timing



The data realignment circuit can have up to 11 bit-times of insertion before a rollover occurs. The programmable bit rollover point can be from 1 to 11 bit-times, independent of the deserialization factor. The programmable bit rollover point should be set to be equal to or greater than the deserialization factor, allowing enough depth in the word alignment circuit to slip through a full word. The value of bit rollover point can be set using the MegaWizard Plug-In Manager. An optional status port, RX_CDA_MAX, is available to the FPGA fabric from each channel to indicate when the preset rollover point is reached.

Figure 8–16 illustrates a preset value of four bit-times before rollover occurs. The rx_cda_max signal pulses for one rx_outclock cycle to indicate that the rollover has occurred.

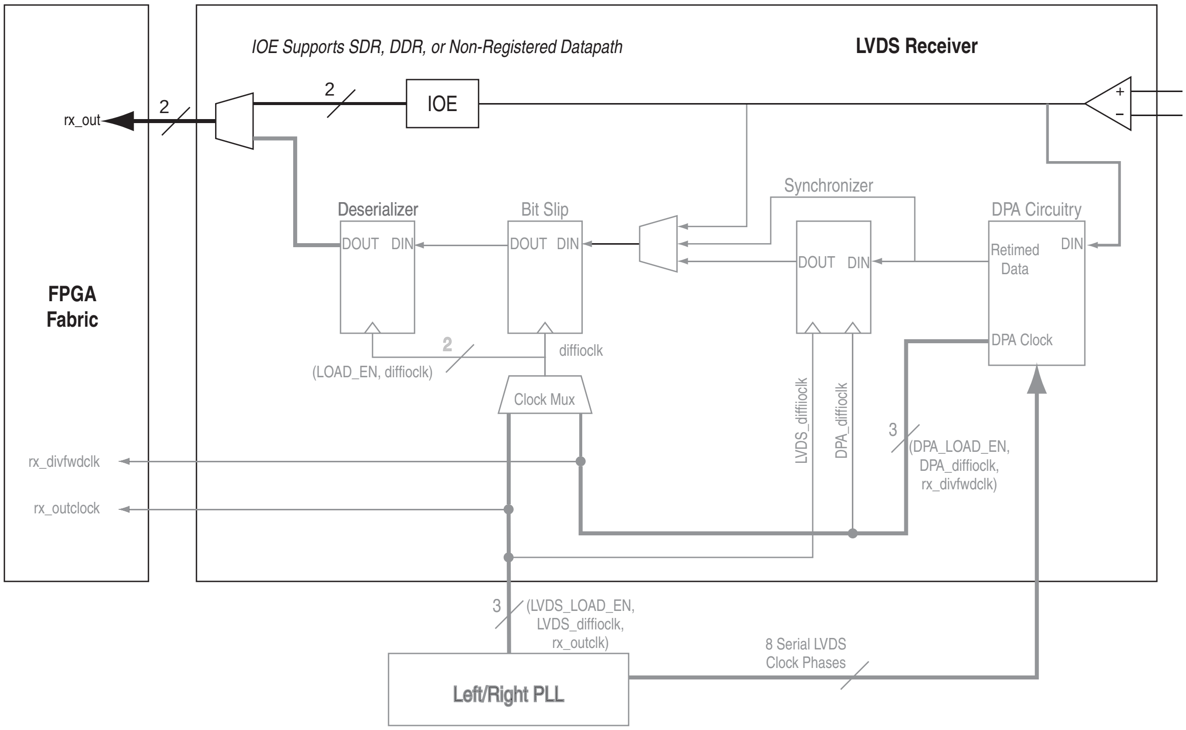
Figure 8–16. Receiver Data Re-alignment Rollover



Deserializer

The deserialization factor can be statically set to 4, 6, 7, 8, or 10 by using the Quartus II software. The Stratix IV deserializer can be bypassed in the Quartus II MegaWizard Plug-In Manager to support DDR (×2) or SDR (×1) operations, as shown Figure 8–17. The DPA and the data realignment circuit cannot be used when the deserializer is bypassed. The IOE contains two data input registers that can operate in DDR or SDR mode.

Figure 8–17. Stratix IV Deserializer Bypass *(Note 1)*, *(2)*, *(3)*



Note to Figure 8–17:

(1) All disabled blocks and signals are grayed out.

(2) In DDR mode, `rx_inclock` clocks the IOE register. In SDR mode, data is directly passed through IOE.

(3) In SDR mode and DDR mode, the data width from IOE is 1 and 2, respectively.

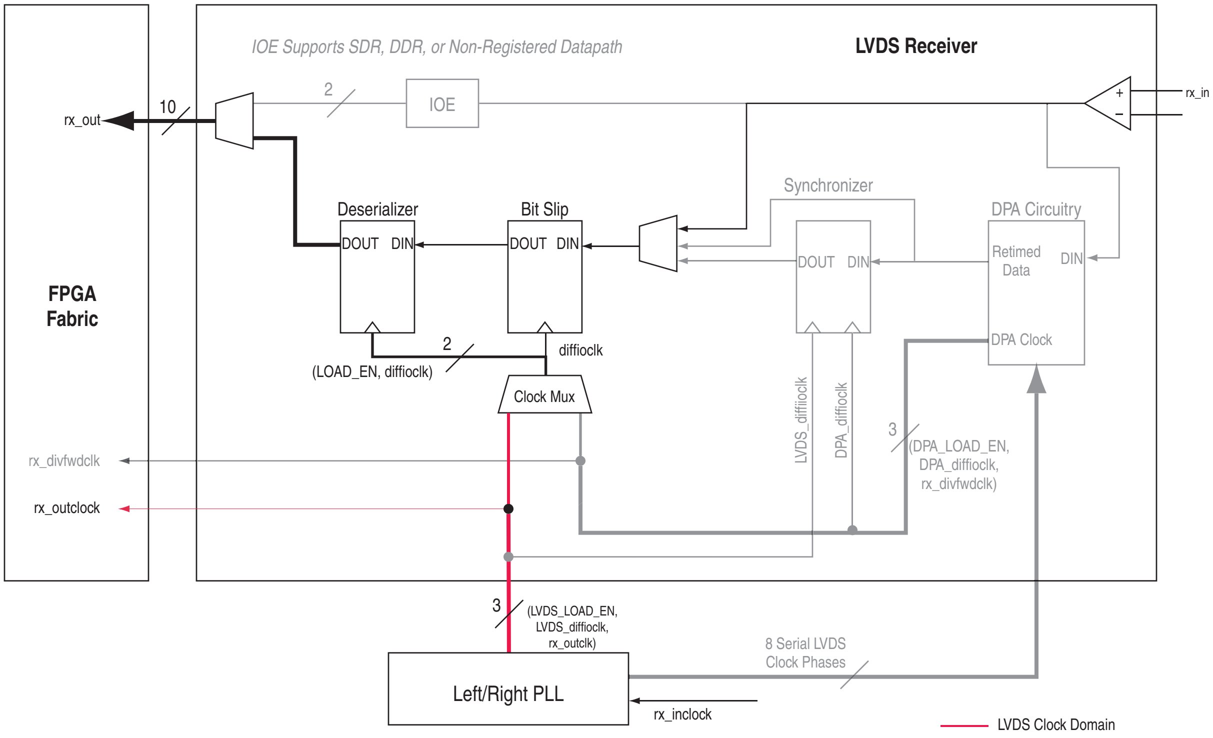
Receiver Data Path Modes

The Stratix IV family supports three receiver datapath modes: non-DPA mode, DPA mode and soft-CDR mode.

Non-DPA Mode

Figure 8–18 shows the non-DPA datapath block diagram. In non-DPA mode, DPA and synchronizer blocks are disabled. Input serial data is registered at the rising or falling edge of the serial `LVDS_diffioclk` clock produced by the left/right PLL. You can select the rising/falling edge option using ALTLDVS MegaWizard Plug-in Manager. Both data realignment and deserializer blocks are clocked by `LVDS_diffioclk` clock, which is generated by the left/right PLL.

Figure 8–18. Receiver Data Path in Non-DPA Mode *(Note 1)*, *(2)*



Note to Figure 8–18:

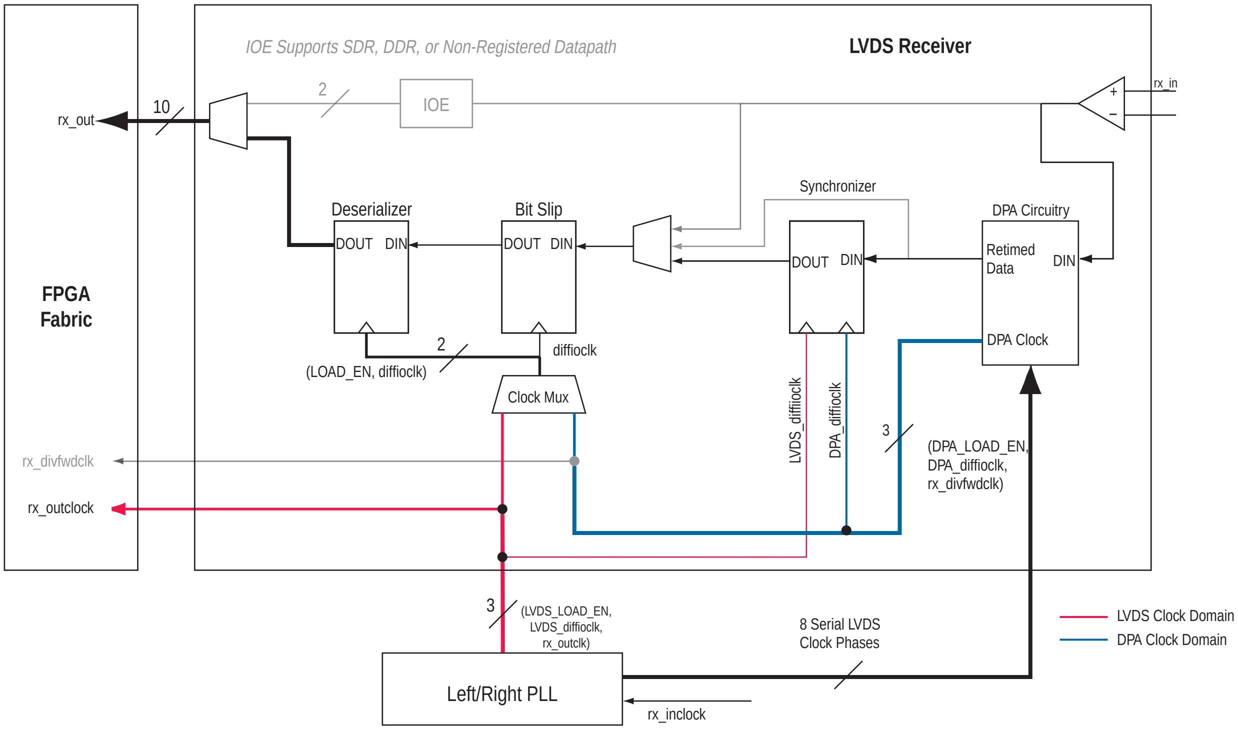
(1) In SDR mode and DDR mode, the data width from IOE is 1 and 2, respectively.

(2) The `rx_out` port has a maximum data width of 10.

DPA Mode

Figure 8–19 shows the DPA mode datapath, where all the hardware blocks mentioned in the section "Receiver Hardware Blocks" on page 8–17 are active. DPA block chooses the best possible clock (`DPA_diffioclk`) from the eight fast clocks sent by the left/right PLL. This serial `DPA_diffioclk` clock is used for writing the serial data into the synchronizer. A serial `LVDS_diffioclk` clock is used for reading the serial data from the synchronizer. The same `LVDS_diffioclk` is used in data realignment and deserializer blocks.

Figure 8–19. Receiver Datapath in DPA Mode *(Note 1)*, *(2)*, *(3)*



Note to Figure 8–19:

(1) All disabled blocks and signals are grayed out.

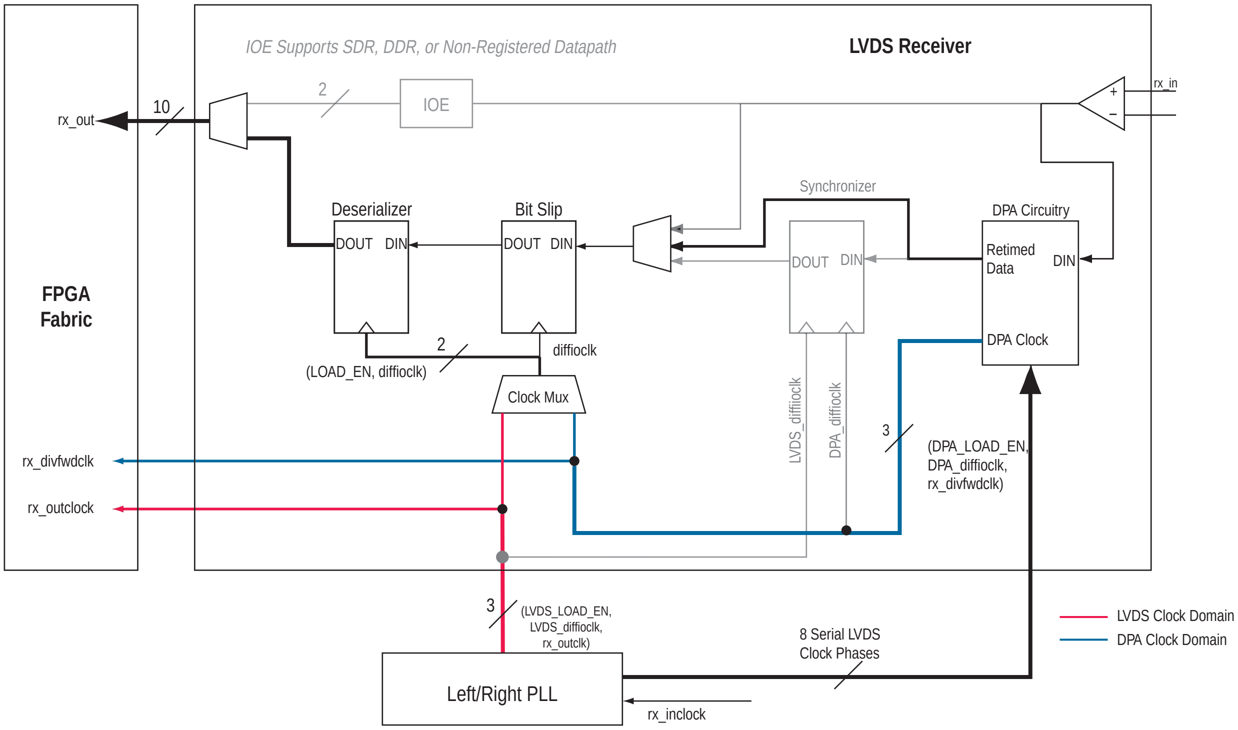
(2) In SDR mode and DDR mode, the data width from IOE is 1 and 2, respectively.

(3) The `rx_out` port has a maximum data width of 10.

Soft-CDR Mode

The Stratix IV LVDS channel offers the soft-CDR mode to support the Gigabit Ethernet/SGMII protocols. A receiver PLL uses the local clock source for reference. Figure 8–20 shows the soft-CDR mode datapath.

Figure 8–20. Receiver Datapath in Soft-CDR Mode *(Note 1), (2), (3)*



Note to Figure 8–20:

(1) All disabled blocks and signals are grayed out.

(2) In SDR mode and DDR mode, the data width from IOE is 1 and 2, respectively.

(3) The `rx_out` port has a maximum data width of 10.

In the soft-CDR mode, the synchronizer block is inactive. The DPA circuitry selects an optimal DPA clock phase to sample the data. The selected DPA clock is used for bit-slip operation and deserialization. The DPA block also forwards the selected DPA clock, divided down by the deserialization factor called `rx_divfwdclk`, to the FPGA fabric, along with the deserialized data. This clock signal is put on the PCLK (periphery clock) network.

For more information about PCLK networks, refer to the *Clock Networks and PLLs in Stratix IV Devices* chapter in the *Stratix IV Device Handbook*.

Every LVDS channel can be used in soft-CDR mode, and can drive the FPGA fabric via the PCLK network in the Stratix IV family. The `rx_dpa_locked` signal is not valid in soft-CDR mode, as the DPA continuously changes its phase to track PPM differences between the upstream transmitter and the local receiver input reference clocks. The parallel clock `rx_outclock`, generated by the left/right PLL, is also forwarded to the FPGA fabric.

Stratix IV LVDS Interface with 'Use External PLL' Option Enabled

The ALTLVDS MegaWizard Plug-In Manager provides an option for implementing the LVDS interface with the **Use External PLL** option. With this option enabled, you can control the PLL settings, such as dynamically reconfiguring the PLL to support different data rates, dynamic phase shift, and other settings. You also must instantiate an ALTPLL megafunction to generate the various clock and load enable signals.

When the **Use External PLL** option is enabled with the ALTLVDS transmitter/receiver, the following signals are required from the ALTPLL:

- Serial clock input to the serializer/deserializer of the ALTLVDS transmitter/receiver
- Load enable to the serializer/deserializer of the ALTLVDS transmitter/receiver
- Parallel clock used to clock the transmitter FPGA fabric logic and parallel clock used for receiver `rx_syncclock` port/receiver FPGA fabric logic
- Asynchronous PLL reset port of the ALTLVDS receiver

Table 8–10 specifies the signal interface between the output ports of the ALTPLL and input ports of the ALTLVDS transmitter/receiver.

Table 8–10. Signal Interface Between ALTPLL and ALTLVDS

From ALTPLL *(1)*	To ALTLVDS Transmitter	To ALTLVDS Receiver
Serial clock output (c0)	`tx_inclock` (serial clock input to the transmitter)	`rx_inclock` (serial clock input)
Load enable output (c1)	`tx_enable` (load enable to the transmitter)	`rx_enable` (load enable for the deserializer)
Parallel clock output (c2)	Parallel clock used inside the transmitter core logic in the FPGA fabric	`rx_syncclock` (parallel clock input) and parallel clock used inside the receiver core logic in the FPGA fabric
~(locked)	—	`pll_areset` *(2)* (asynchronous PLL reset port)

Note to Table 8–10:

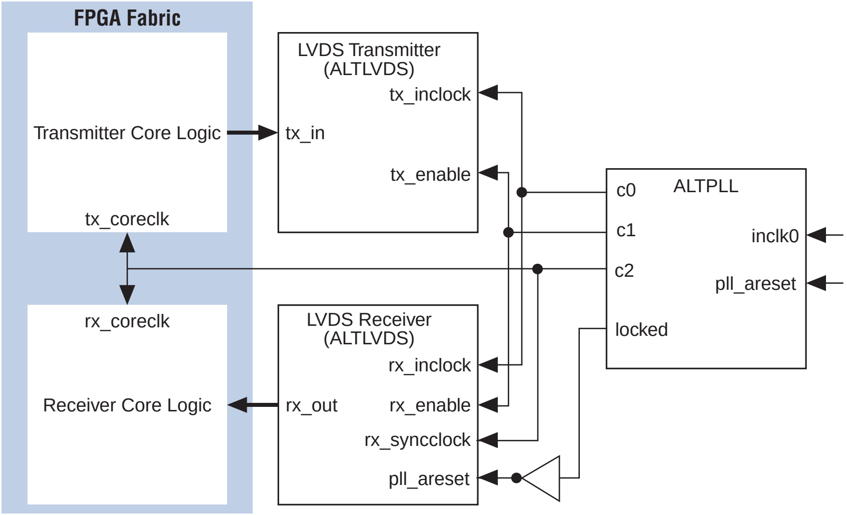
(1) The table shows the serial clock output, load enable output, and parallel clock output generated on ports c0, c1, and c2, respectively, along with the locked signal of the ALTPLL instance as an example. You can choose any of the PLL output clock ports to generate the interface clocks.

(2) The `pll_areset` signal automatically gets enabled for the LVDS receiver in external PLL mode. This signal does not exist for LVDS transmitter instantiation, when the external PLL option is enabled.

☞ The `rx_syncclock` port gets enabled automatically in an LVDS receiver in external PLL mode. The Quartus II compiler errors out if this port is not connected as shown in Figure 8–21.

When generating the ALTPLL megafunction, the **Left/Right PLL** option is configured to set up the PLL in LVDS mode. Figure 8–21 illustrates the connection between ALTPLL and ALTLVDS:

Figure 8–21. LVDS Interface with the ALTPLL *(Note 1)*



Note to Figure 8–21:

(1) Instantiation of `pll_areset` is optional for the ALTPLL instantiation.

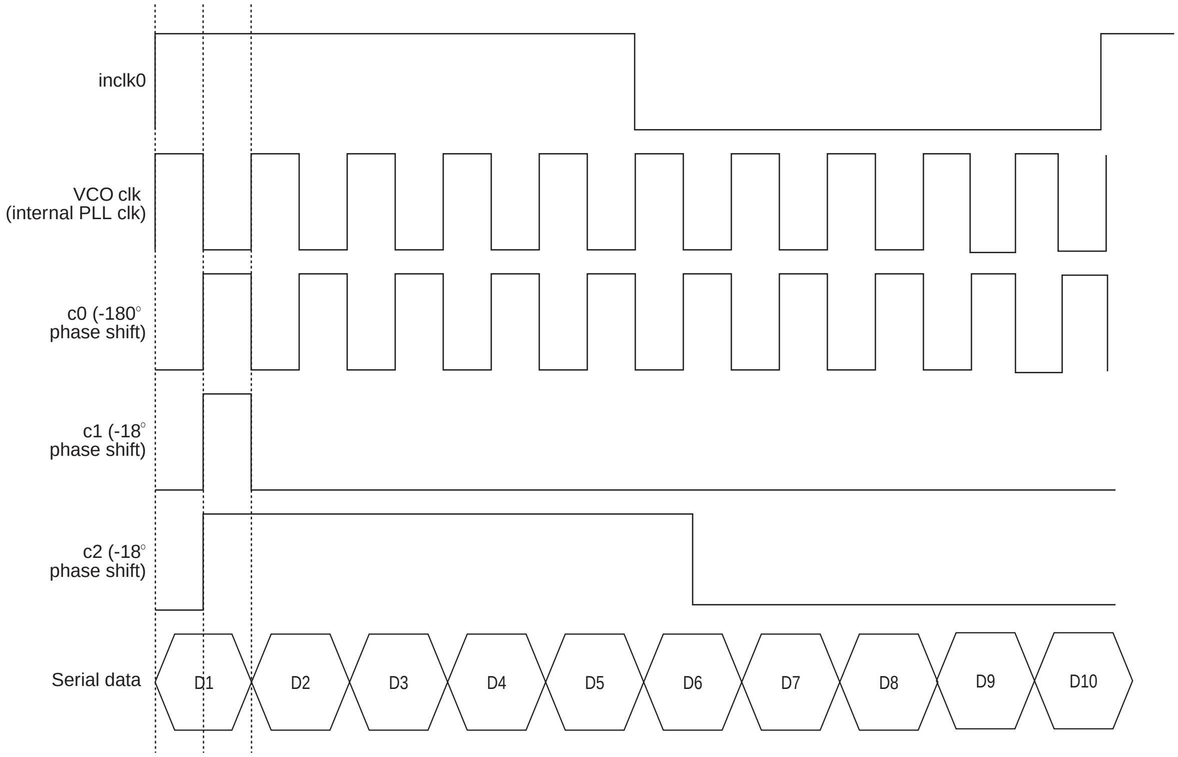
Example: LVDS data rate = 1 Gbps; serialization factor = 10; input reference clock = 100 MHz

The following settings are used while generating the three output clocks using an ALTPLL megafunction. The serial clock must be 1000 MHz and the parallel clock must be 100 MHz (serial clock divided by the serialization factor):

- c0
  - Frequency = 1000 MHz (multiplication factor = 10 and division factor = 1)
  - Phase shift = -180° with respect to the VCO clock
  - Duty cycle = 50%
- c1
  - Frequency = (1000/10) = 100 MHz (multiplication factor = 1 and division factor = 1)
  - Phase shift = (-180/10) = -18° (c0 phase shift divided by the serialization factor)
  - Duty cycle = (100/10) = 10% (100 divided by the serialization factor)
- c2
  - Frequency = (1000/10) = 100 MHz (multiplication factor = 1 and division factor = 1)
  - Phase shift = (-180/10) = -18° (c0 phase shift divided by the serialization factor)
  - Duty cycle = 50%

The above calculations for phase shift assume that the input clock and serial data are edge aligned. Introducing a phase shift of -180° to c0 ensures that the input data is center-aligned with respect to the sampling clock (c0), as shown in Figure 8–22.

Figure 8–22. Phase Relationship for External PLL Interface Signals



Stratix IV Left/Right PLLs (PLL_Lx/ PLL_Rx)

The Stratix IV family contains up to eight left/right PLLs with up to four PLLs located on the left side and four on the right side of the device. The left PLLs can support high-speed differential I/O banks on the left side and the right PLLs can support banks on the right side of the device. The high-speed differential I/O receiver and transmitter channels use these left/right PLLs to generate the parallel clocks (`rx_outclock` and `tx_outclock`) and high-speed clocks (`diffioclk`). Figure 8–2 and Figure 8–3 show the locations of the left/right PLLs for Stratix IV E and Stratix IV GX, respectively. The PLL V_{CO} operates at the clock frequency of the data rate. Clock switchover and dynamic reconfiguration are allowed using the left/right PLL in high-speed differential I/O support mode.

For more details, refer to the *Clock Network and PLLs in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook*.

Stratix IV Clocking

The left/right PLLs feed into the differential transmitter and receive channels through the LVDS and DPA clock network. The center left/right PLLs can clock the transmitter and receive channels above and below them. The corner left/right PLLs can drive I/Os in the banks adjacent to them. Figure 8–23 and Figure 8–24 show center and corner PLL clocking in the Stratix IV family. More information about PLL clocking restrictions can be found in “Differential Pin Placement Guidelines” on page 8–34.

Figure 8–23. LVDS/DPA Clocks in the Stratix IV Family with Center PLLs

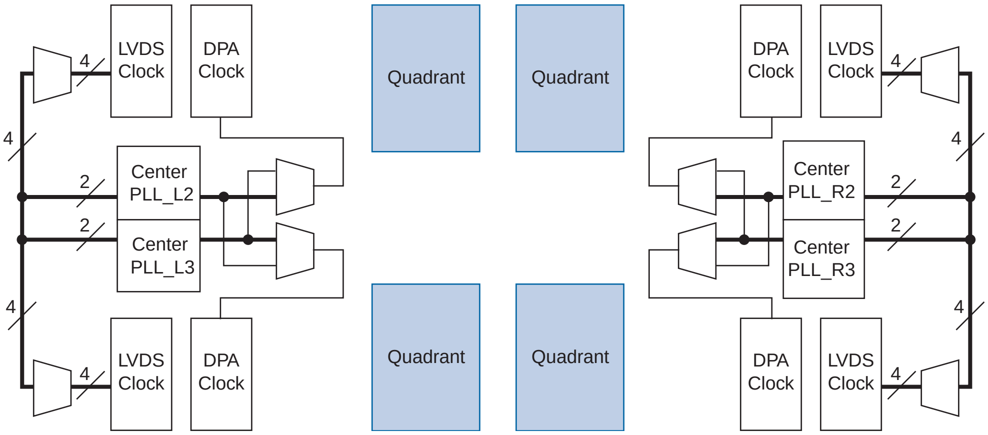
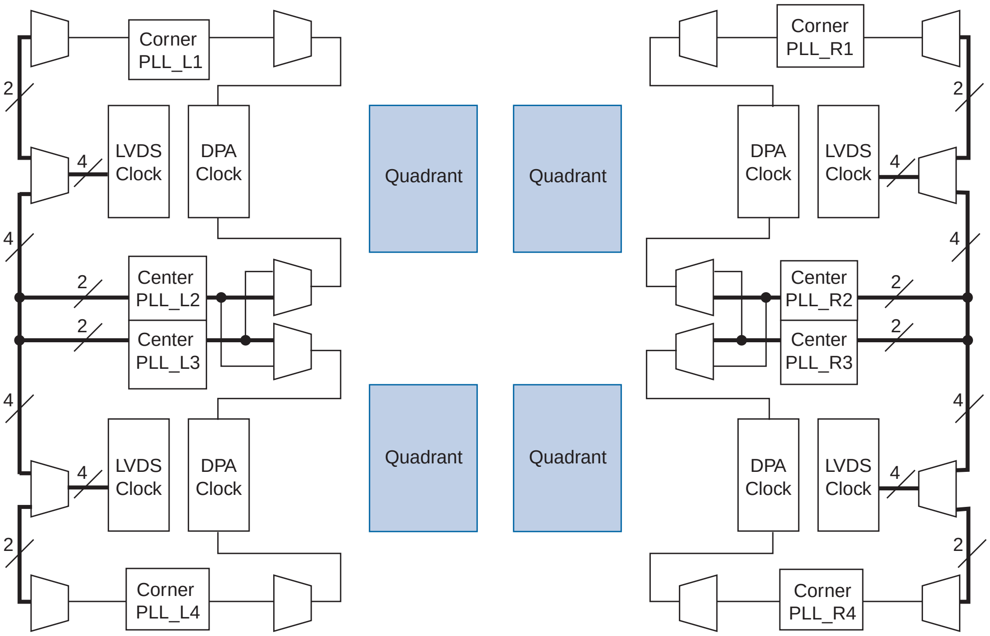


Figure 8–24. LVDS/DPA Clocks in the Stratix IV Family with Center and Corner PLLs



Stratix IV Source-Synchronous Timing Budget

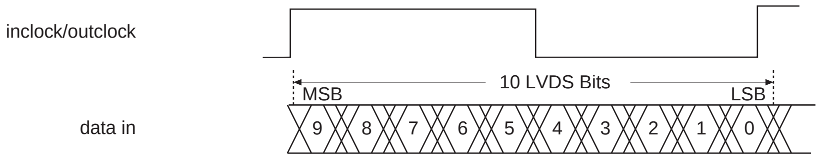
This section describes the timing budget, waveforms, and specifications for source-synchronous signaling in the Stratix IV family. LVDS I/O standards enable high-speed data transmission. This high data transmission rate results in better overall system performance. To take advantage of fast system performance, it is important to understand how to analyze timing for these high-speed signals. Timing analysis for the differential block is different from traditional synchronous timing analysis techniques.

Rather than focusing on clock-to-output and setup times, source synchronous timing analysis is based on the skew between the data and the clock signals. High-speed differential data transmission requires the use of timing parameters provided by IC vendors and is strongly influenced by board skew, cable skew, and clock jitter. This section defines the source-synchronous differential data orientation timing parameters, the timing budget definitions for the Stratix IV family, and how to use these timing parameters to determine a design's maximum performance.

Differential Data Orientation

There is a set relationship between an external clock and the incoming data. For operation at 1 Gbps and serialization factor of 10, the external clock is multiplied by 10, and phase-alignment can be set in the PLL to coincide with the sampling window of each data bit. The data is sampled on the falling edge of the multiplied clock. Figure 8–25 shows the data bit orientation of the ×10 mode.

Figure 8–25. Bit Orientation in Quartus II Software



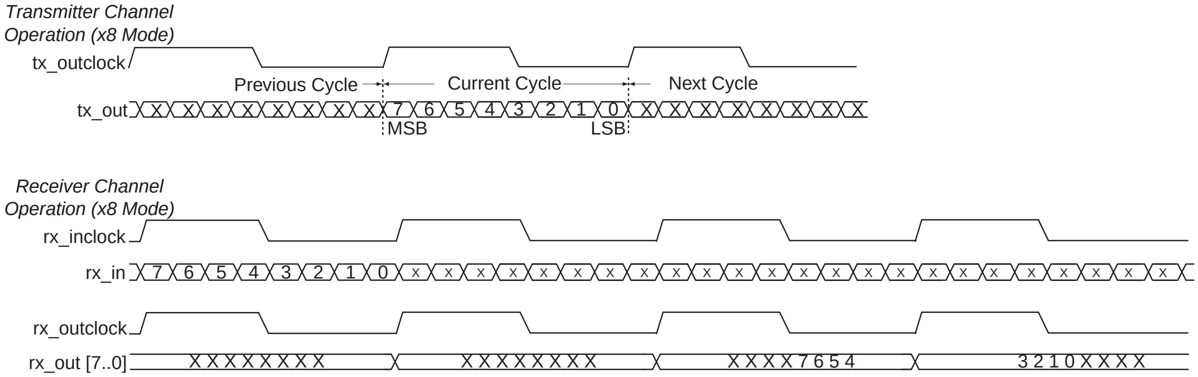
Differential I/O Bit Position

Data synchronization is necessary for successful data transmission at high frequencies. Figure 8–26 shows the data bit orientation for a channel operation. These figures are based on the following:

- Serialization factor equals clock multiplication factor
- Edge alignment is selected for phase alignment
- Implemented in hard SERDES

For other serialization factors, use the Quartus II software tools and find the bit position within the word. The bit positions after deserialization are listed in Table 8–11.

Figure 8–26. Bit-Order and Word Boundary for One Differential Channel *(Note 1)*


Transmitter Channel Operation (x8 Mode)
tx_outclock
Previous Cycle
Current Cycle
Next Cycle
tx_out
X X X X X X X X X X X X X X X X X X X X X X X X
7 6 5 4 3 2 1 0
MSB
LSB
Receiver Channel Operation (x8 Mode)
rx_inclock
rx_in
7 6 5 4 3 2 1 0 x x x x x x x x x x x x x x x x x x x x x x x x
rx_outclock
rx_out [7..0]
XXXXXXXX
XXXXXXXX
XXXX7654
3210XXXX

Note to Figure 8–26:

(1) These are only functional waveforms and are not intended to convey timing information.

Table 8–11 shows the conventions for differential bit naming for 18 differential channels. The MSB and LSB positions increase with the number of channels used in a system.

Table 8–11. Differential Bit Naming

Receiver Channel Data Number	Internal 8-Bit Parallel Data	
	MSB Position	**LSB Position**
1	7	0
2	15	8
3	23	16
4	31	24
5	39	32
6	47	40
7	55	48
8	63	56
9	71	64
10	79	72
11	87	80
12	95	88
13	103	96
14	111	104
15	119	112
16	127	120
17	135	128
18	143	136

Transmitter Channel to Channel Skew for Non-DPA Mode

Transmitter channel-to-channel skew (TCCS) is an important parameter based on the Stratix IV transmitter in a source synchronous differential interface. This parameter is used in receiver skew margin calculation. For more details, refer to the "Receiver Skew Margin for Non-DPA Mode" section.

TCCS is the difference between the fastest and slowest data output transitions, including the TCO variation and clock skew. For LVDS transmitters, TimeQuest Timing Analyzer provides a TCCS report, which shows TCCS values for serial output ports.

You can get the TCCS value from the TCCS report (`report_TCCS`) in the Quartus II compilation report under TimeQuest Timing Analyzer, or from the *DC and Switching Characteristics of Stratix IV Device Family* chapter in volume 4 of the *Stratix IV Device Handbook*.

Receiver Skew Margin for Non-DPA Mode

Changes in system environment, such as temperature, media (cable, connector, or PCB), and loading effect the receiver's setup and hold times; internal skew affects the sampling ability of the receiver.

Different modes of LVDS receivers use different specifications, which can help in deciding the ability to sample the received serial data correctly. In DPA mode, you should use DPA jitter tolerance instead of receiver skew margin (RSKM).

In non-DPA mode, RCCS, RSKM, and SW specifications are used for high-speed source-synchronous differential signals in the receiver data path. The relationship between RSKM, RCCS and SW can be expressed by following RSKM equation:

$$RSKM = TUI - SW - RCCS$$

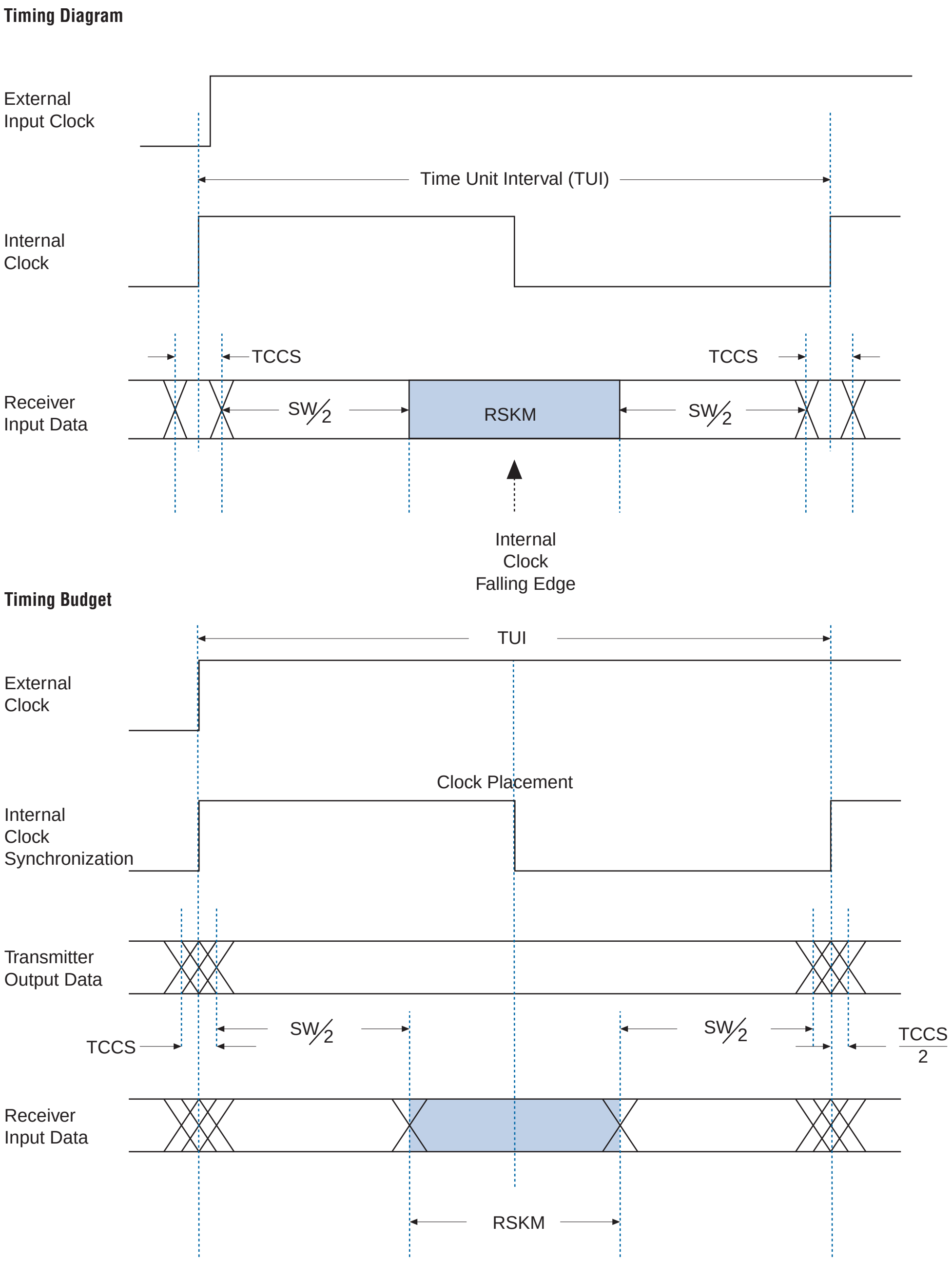
Where:

- TUI — Time period of the serial data.
- RSKM (Receiver Skew Margin) — The timing margin between the receiver's clock input and the data input sampling window.
- SW (Sampling Window) — The period of time that the input data must be stable to ensure that data is successfully sampled by the LVDS receiver. The SW is a device property and varies with device speed grade.
- RCCS (Receiver channel-to-channel Skew) — This involves the TCCS value of the upstream transmitter and the skew introduced by the system channel paths. For a non-Altera source synchronous transmitter, the TCCS value will be replaced with the corresponding parameter value for channel-to-channel skew.

Figure 8–27 shows the relationship between the RSKM, RCCS and the receiver's SW.

You must calculate the RSKM value to decide whether data can be sampled properly by the LVDS receiver or not with the given data rate, and device. A positive RSKM value indicates that LVDS receiver can sample the data properly, whereas a negative RSKM indicates that it cannot.

Figure 8–27. Differential High-Speed Timing Diagram and Timing Budget for Non-DPA



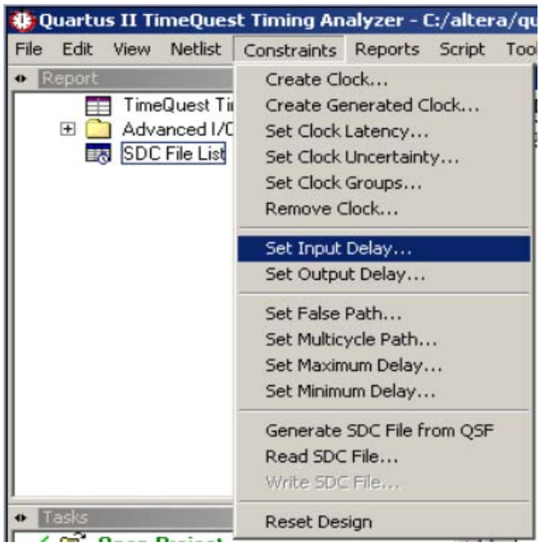
For LVDS receivers, the Quartus II software provides an RSKM report showing SW, TUI, and RSKM values for the non-DPA mode. The RSKM report can be generated by executing the `report_RSKM` command in the TimeQuest Timing Analyzer. You can find the RSKM report in the Quartus II compilation report under the TimeQuest Timing Analyzer section.

☞ In order to obtain the RSKM value, you should assign an appropriate Input Delay to the LVDS receiver through the TimeQuest Timing Analyzer constraints menu.

RCCS is equal to the difference between maximum input delay and minimum input delay. For assigning **Input Delay**, follow these steps:

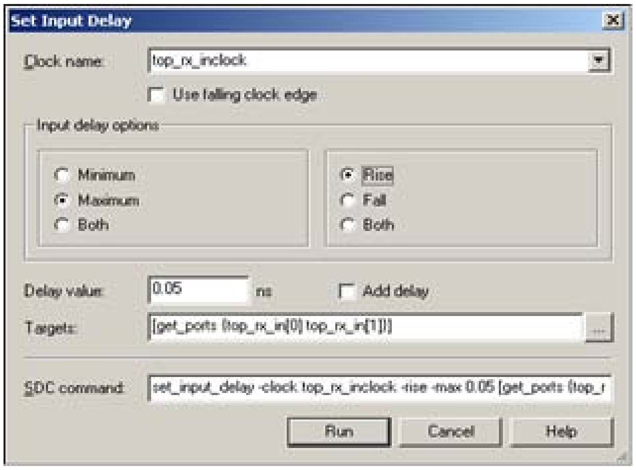
- The Quartus II TimeQuest Timing Analyzer GUI has many options for setting the constraints and analyzing the design. Figure 8–28 shows various commands on the Constraints menu. For setting input delay, you need to select **Set Input Delay** option.

Figure 8–28. Selection of Constraint Menu in TimeQuest Timing Analyzer



- Figure 8–29 shows the window for setting different parameters for **Set Input Delay**. The clock name should reference the source synchronous clock that feeds the LVDS receiver. You can select the desired clock using the pull down menu.

Figure 8–29. Input Time Delay Assignment Through TimeQuest Timing Analyzer


Set Input Delay

Clock name: top_rx_inclock

☐ Use falling clock edge

Input delay options

○ Minimum
● Maximum
○ Both

● Rise
○ Fall
○ Both

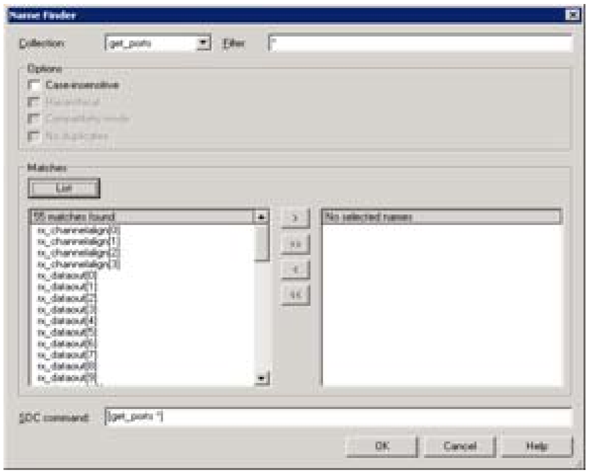
Delay value: 0.05 ns ☐ Add delay

Targets: [get_ports {top_rx_in[0] top_rx_in[1]}]

SDC command: set_input_delay -clock top_rx_inclock -rise -max 0.05 [get_ports {top_r

- Figure 8–30 shows the window that appears when you are selecting the **Targets** option. Using the **List** option in the node finder window, you can get a list of all available ports.

Figure 8–30. Node Finder Window in Set Input Delay Option



Name Finder

Collection: get_ports

Filter: *

Options

- ☐ Case-insensitive
- ☐ Hierarchical
- ☐ Compatibility mode
- ☐ No duplicates

Matches

55 matches found

rx_channelaligns[0]
rx_channelaligns[1]
rx_channelaligns[2]
rx_channelaligns[3]
rx_dataout[0]
rx_dataout[1]
rx_dataout[2]
rx_dataout[3]
rx_dataout[4]
rx_dataout[5]
rx_dataout[6]
rx_dataout[7]
rx_dataout[8]
rx_dataout[9]

No selected names

SDC command: [get_ports *]

- Select the LVDS receiver serial input ports (from the list) according to which input delay is set. Press **OK** to close this window.
- In the **Set Input Delay** window, provide the appropriate values in the **Input Delay Options** section and **Delay** value.
- Press **Run** to incorporate these values in TimeQuest Timing.
- Assign the appropriate delay for all the LVDS receiver input ports following the same steps as mentioned above. If you have already assigned **Input Delay** once and you need to add more delay to that input port, use the **Add Delay** option in the **Set Input Delay** window.

☞ If no input delay is set in the TimeQuest Timing Analyzer, the RCCS will default to zero. The input delay can also be set directly in a **.sdc** file using the `set_input_delay` command.

For more information on sdc file commands and TimeQuest Timing Analyzer, refer to the *Quartus II TimeQuest Timing Analyzer* chapter in volume 3 of the *Quartus II Development Software Handbook*.

Example:

Data Rate: 1 GbPs, Board channel-to-channel skew = 200 ps

For Stratix IV devices:

Transmitter channel-to-channel skew (TCCS) = 100ps (Pending characterization)
Sampling Window (SW) = 300ps (Pending characterization)

Time period of the serial data (TUI) = 1000ps
Total receivers channel-to-channel skew (RCCS) = TCCS + Board channel-to-channel skew = 100 ps + 200 ps
= 300 ps

RSKM = TUI - SW - RCCS
= 1000ps - 300ps - 300ps

= 400ps > 0

Since the receiver skew margin (RSKM) > 0ps, the receiver non-DPA mode should work correctly.

☞ You can also calculate RSKM using the steps described in the section "Guidelines for DPA-Enabled Differential Channels" on page 8–34.

Differential Pin Placement Guidelines

To ensure proper high-speed operation, differential pin placement guidelines have been established. The Quartus II compiler automatically checks that these guidelines are followed, and issues an error message if they are not met.

Since DPA usage adds some constraints on the placement of high-speed differential channels, this section is divided into pin placement guidelines with and without DPA usage.

☞ DPA enabled differential channels refer to DPA mode or soft-CDR mode; DPA disabled channels refer to non-DPA mode.

Guidelines for DPA-Enabled Differential Channels

The Stratix IV family devices have differential receivers and transmitters in I/O banks on the left and right sides of the device. Each receiver has a dedicated DPA circuit to align the phase of the clock to the data phase of its associated channel. When DPA-enabled channels are used in differential banks, you must adhere to the guidelines listed in the following sections.

DPA-Enabled Channels and Single-Ended I/Os

When there is a DPA channel enabled in a bank, both single-ended I/Os and differential I/O standards are allowed in the bank.

- Single-ended I/Os are allowed in the same I/O bank, as long as the single-ended I/O standard uses the same V_{CCIO} as the DPA-enabled differential I/O bank.
- Single-ended inputs can be in the same LAB row as a differential channel using the SERDES circuitry.
- IOE input registers are not available for the single-ended I/Os placed in the same LAB row as differential I/Os.
- DDIO can be placed within the same LAB row as a SERDES differential channel but half rate DDIO (single data rate) output pins cannot be placed within the same LAB row as a receiver SERDES differential channel. The input register must be implemented within the FPGA fabric logic.

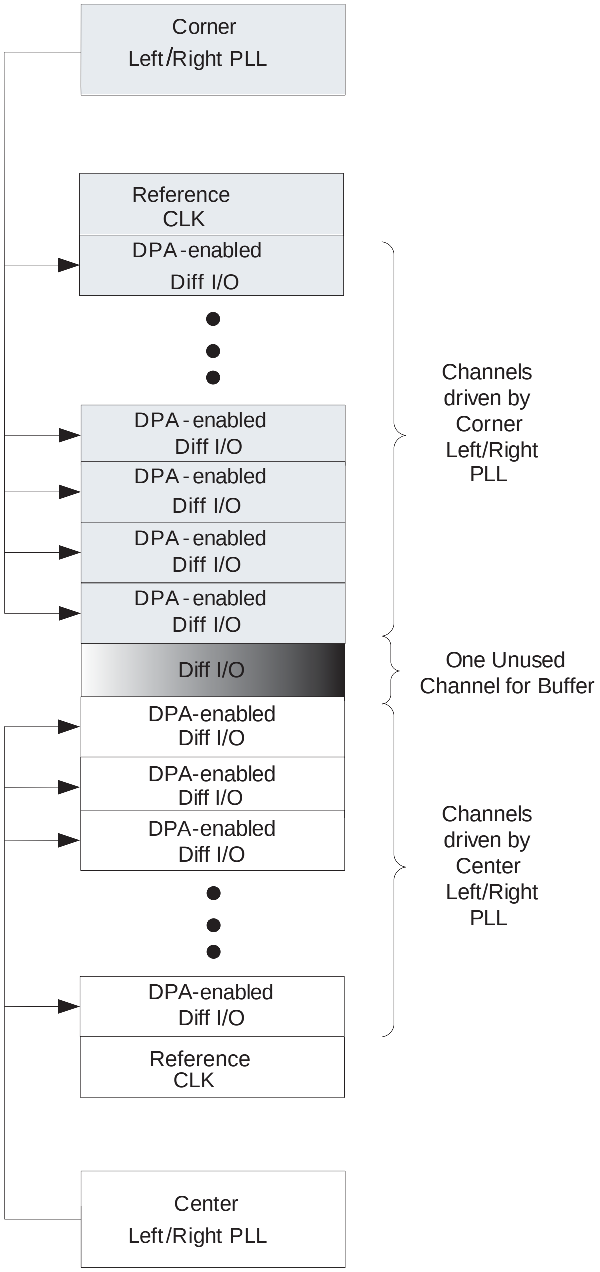
DPA-Enabled Channel Driving Distance

If the number of DPA channels driven by each left/right PLL exceeds 25 LAB rows, Altera recommends implementing data realignment (bit slip) circuitry for all the DPA channels.

Using Corner and Center Left/Right PLLs

- If a differential bank is being driven by two left/right PLLs, where the corner left/right PLL is driving one group and the center left/right PLL is driving another group, there must be at least one row of separation between the two groups of DPA-enabled channels (refer to Figure 8–31). The two groups can operate at independent frequencies.
- No separation is necessary if a single left/right PLL is driving DPA-enabled channels, as well as DPA-disabled channels.

Figure 8–31. Corner and Center Left/Right PLLs Driving DPA-Enabled Differential I/Os in the Same Bank



Using Both Center Left/Right PLLs

- Both center left/right PLLs can be used to drive DPA-enabled channels simultaneously, as long as they drive these channels in their adjacent banks only, as shown in Figure 8–32.
- If one of the center left/right PLLs drives the top and bottom banks, the other center left/right PLL cannot be used to drive differential channels, as shown in Figure 8–32.
- If the top PLL_L2/PLL_R2 drives DPA-enabled channels in the lower differential bank, the PLL_L3/PLL_R3 cannot drive DPA-enabled channels in the upper differential banks, and vice versa. In other words, the center left/right PLLs cannot drive cross-banks simultaneously, as shown in Figure 8–33.

Figure 8–32. Center Left/Right PLLs Driving DPA-Enabled Differential I/Os

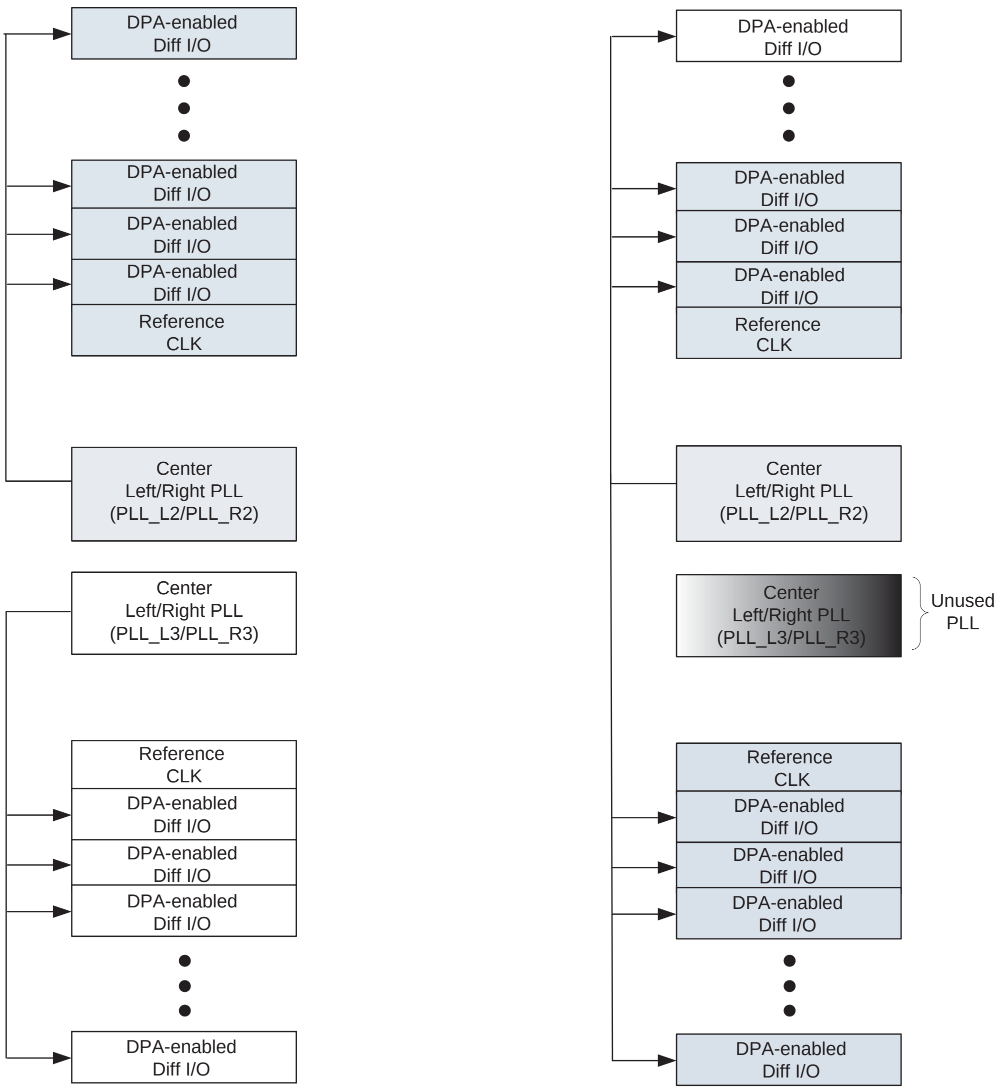
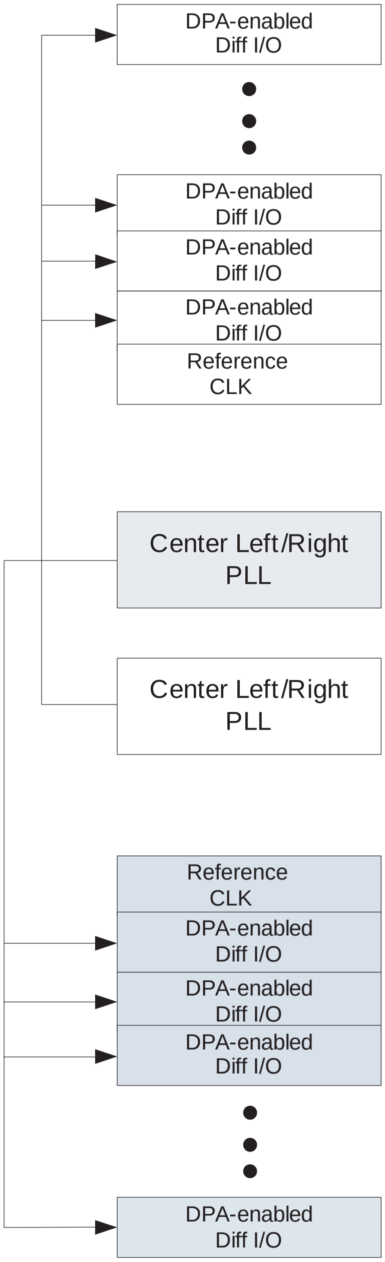


Figure 8–33. Invalid Placement of DPA-Enabled Differential I/Os Driven by Both Center Left/Right PLLs



Guidelines for DPA-Disabled Differential Channels

When DPA-disabled channels are used in the left and right banks of a Stratix IV family device, you must adhere to the guidelines in the following sections.

DPA-Disabled Channels and Single-Ended I/Os

The placement rules for DPA-disabled channels and single-ended I/Os are the same as those for DPA-enabled channels and single-ended I/Os.

DPA-Disabled Channel Driving Distance

Each left/right PLL can drive all the DPA-disabled channels in the entire bank.

Using Corner and Center Left/Right PLLs

- A corner left/right PLL can be used to drive all transmitter channels and a center left/right PLL can be used to drive all DPA-disabled receiver channels within the same differential bank. In other words, a transmitter channel and a receiver channel in the same LAB row can be driven by two different PLLs, as shown in Figure 8–34.
- A corner left/right PLL and a center left/right PLL can drive duplex channels in the same differential bank, as long as the channels driven by each PLL are not interleaved. No separation is necessary between the group of channels driven by the corner and center left/right PLLs. See Figure 8–34 and Figure 8–35.

Figure 8–34. Corner and Center Left/Right PLLs Driving DPA-Disabled Differential I/Os in the Same Bank

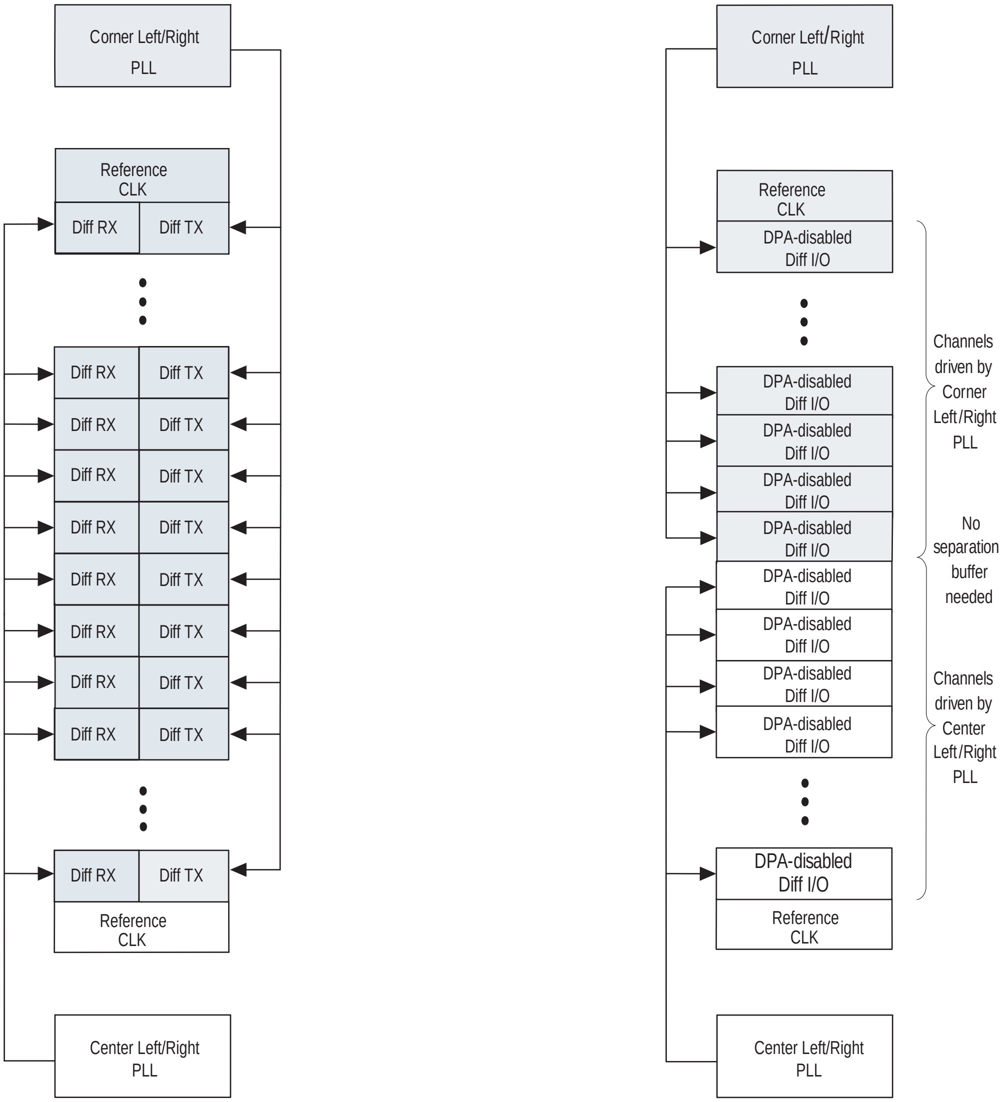
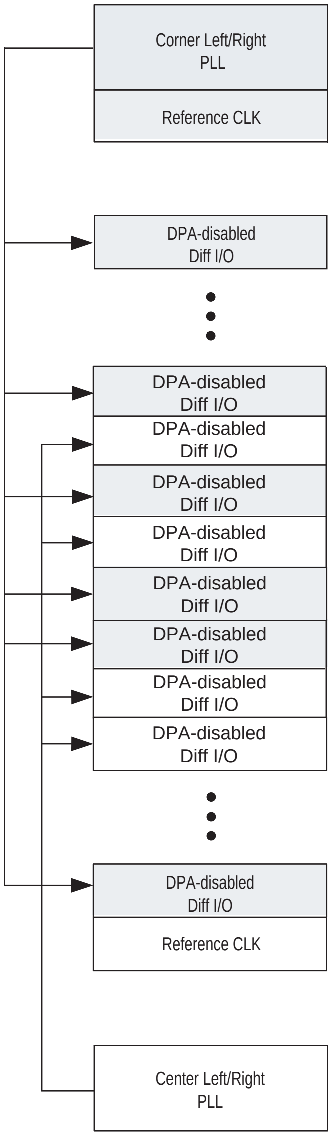


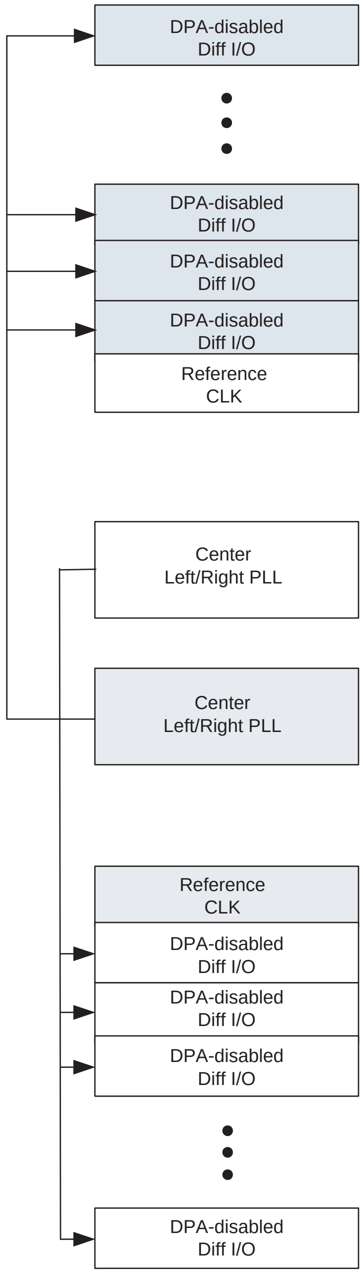
Figure 8–35. Invalid Placement of DPA-Disabled Differential I/Os Due to Interleaving of Channels Driven by the Corner and Center Left/Right PLLs



Using Both Center Left/Right PLLs

Both center left/right PLLs can be used simultaneously to drive DPA-disabled channels on upper and lower differential banks. Unlike DPA-enabled channels, the center left/right PLLs can drive cross-banks. For example, the upper-center left/right PLL can drive the lower differential bank at the same time the lower center left/right PLL is driving the upper differential bank, and vice versa, as shown in Figure 8–36.

Figure 8–36. Both Center Left/Right PLLs Driving Cross-Bank DPA-Disabled Channels Simultaneously



Referenced Documents

This chapter references the following documents:

- *Clock Networks and PLLs in Stratix IV Devices* chapter in the *Stratix IV Device Handbook*
- *DC and Switching Characteristics of Stratix IV Device Family* chapter in volume 4 of the *Stratix IV Device Handbook*

- *I/O Features in Stratix IV Devices* chapter in the *Stratix IV Device Handbook*
- *Quartus II TimeQuest Timing Analyzer* chapter in volume 3 of the *Quartus II Development Software Handbook*
- *Signal Integrity Analysis with Third-Party Tools* chapter in volume 3 of the *Quartus II Handbook*

Document Revision History

Table 8–12 shows the revision history for this document.

Table 8–12. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	Changes to figures including Figure 8–2, Figure 8–3, Figure 8–21, Figure 8–34. Removed Figure 8–31. Changes to tables including Table 8–1, Table 8–10. Changes to section "Differential Pin Placement Guidelines".	TBA Please disregard this section at this time -- it will be completed after stakeholder's review to ensure all changes are listed.
May 2008 v1.0	Initial Release.	—

Section III. System Integration

This section includes the following chapters:

- Chapter 9, Hot Socketing and Power-On Reset in Stratix IV Devices
- Chapter 10, Configuration, Design Security, and Remote System Upgrades in Stratix IV Devices
- Chapter 11, SEU Mitigation in Stratix IV Devices
- Chapter 12, JTAG Boundary-Scan Testing in Stratix IV Devices
- Chapter 13, Power Management in Stratix IV Devices

Revision History

Refer to each chapter for its own specific revision history. For information on when each chapter was updated, refer to the Chapter Revision Dates section, which appears in the full handbook.

9. Hot Socketing and Power-On Reset in Stratix IV Devices

SIV51009-2.0

Introduction

This chapter contains information about hot-socketing specifications, power-on reset (POR) requirements, and their implementation in Stratix® IV devices.

Stratix IV E devices offer hot socketing, also known as hot plug-in or hot swap, and power sequencing support without the use of any external devices. You can insert or remove a Stratix IV E device or a board in a system during system operation without causing undesirable effects to the running system bus or the board that was inserted into the system.

The hot-socketing feature also removes some of the difficulty when you use Stratix IV E devices on PCBs that contain a mixture of 3.0-, 2.5-, 1.8-, 1.5- and 1.2-V devices. With the Stratix IV E hot-socketing feature, you no longer have to ensure a proper power-up sequence for each device on the board.

The Stratix IV E hot-socketing feature provides:

- Board or device insertion and removal without external components or board manipulation
- Support for any power-up sequence
- I/O buffers non-intrusive to system buses during hot insertion

This section also discusses the POR circuitry in Stratix IV devices. The POR circuitry keeps the devices in the reset state until the power supply outputs are within operating range.

This chapter contains the following sections:

- "Stratix IV E Hot-Socketing Specifications"
- "Hot-Socketing Feature Implementation in Stratix IV E Devices" on page 9–3
- "Power-On Reset Circuitry" on page 9–4
- "Power-On Reset Specifications" on page 9–5

Stratix IV E Hot-Socketing Specifications

Stratix IV E devices are hot-socketing compliant without the need for any external components or special design requirements. Hot-socketing support in Stratix IV E devices has the following advantages:

- You can drive the device before power-up without damaging it.
- I/O pins remain tri-stated during power-up. The device does not drive out before or during power-up, thereby not affecting other buses in operation.

- You can insert a Stratix IV E device into or remove it from a powered-up system board without damaging the system board or interfering with its operation.

Altera uses GND as reference for the hot-socketing and I/O buffer circuitry design. You must connect the GND between boards before connecting the V_{CCINT} and the V_{CCIO} power supplies to ensure device reliability and compliance to the hot-socketing specifications.

Stratix IV E Devices Can Be Driven Before Power Up

You can drive signals into I/O pins, dedicated input pins, and dedicated clock pins of Stratix IV E devices before or during power up or power down without damaging the device. Stratix IV E devices support power up or power down of the V_{CCIO}, V_{CC}, V_{CCPGM}, and V_{CCPD} power supplies in any sequence in order to simplify system-level design.

I/O Pins Remain Tri-Stated During Power Up

A device that does not support hot socketing can interrupt system operation or cause contention by driving out before or during power up. In a hot-socketing situation, the Stratix IV E device’s output buffers are turned off during system power up or power down. Also, the Stratix IV device does not drive out until the device is configured and working within recommended operating conditions.

Insertion or Removal of a Stratix IV E Device from a Powered-Up System

Devices that do not support hot socketing can short power supplies when powered-up through the device signal pins. This irregular power up can damage both the driving and driven devices and can disrupt card power up.

You can insert a Stratix IV E device into or remove it from a powered-up system board without damaging the system board or interfering with its operation.

You can power up or power down the V_{CCIO}, V_{CC}, V_{CCPGM}, and V_{CCPD} supplies in any sequence. Individual power supply ramp-up and ramp-down rates can range from 50 µs to 100 ms. During hot socketing, the I/O pin capacitance is less than 15 pF and the clock pin capacitance is less than 20 pF.

For more information about the hot-socketing specification, refer to the *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook* and the *Hot-Socketing and Power Sequencing Feature and Testing for Altera Devices* white paper.

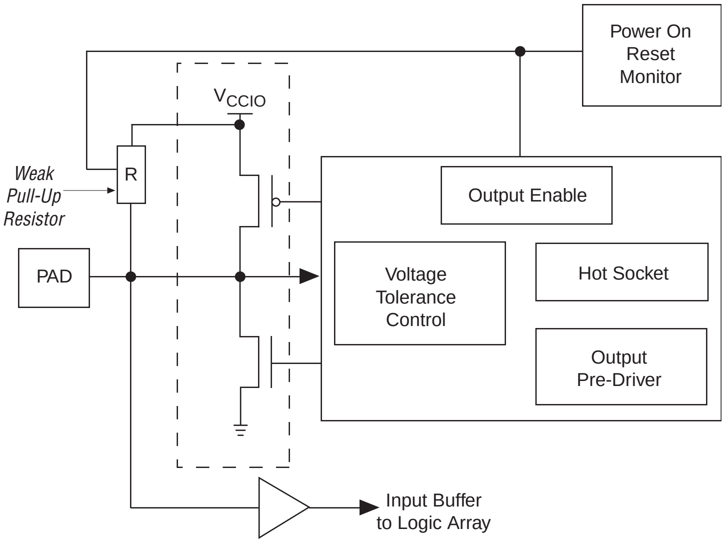
A possible concern regarding hot socketing is the potential for “latch-up.” Nevertheless, Stratix IV E devices are immune to latch-up when hot socketing. Latch-up can occur when electrical subsystems are hot socketed into an active system. During hot socketing, the signal pins can be connected and driven by the active system before the power supply can provide current to the device’s power and ground planes. This condition can lead to latch-up and cause a low-impedance path from power to ground within the device. As a result, the device draws a large amount of current, possibly causing electrical damage.

Hot-Socketing Feature Implementation in Stratix IV E Devices

The hot-socketing feature turns off the output buffer during power up and power down of V_{CC}, V_{CCIO}, V_{CCPGM}, or V_{CCPD} power supplies. The hot-socketing circuitry generates an internal `HOTSCKT` signal when the V_{CC}, V_{CCIO}, V_{CCPGM}, or V_{CCPD} power supplies are below the threshold voltage. The hot-socketing circuitry is designed to prevent excess I/O leakage during power up. When the voltage ramps up very slowly, it is still relatively low, even after the POR signal is released and the configuration is completed. The `CONF_DONE`, `nCEO`, and `nSTATUS` pins fail to respond, as the output buffer cannot flip from the state set by the hot-socketing circuit at this low voltage. Therefore, the hot-socketing circuitry has been removed from these configuration pins to make sure that they are able to operate during configuration. Thus, it is expected behavior for these pins to drive out during power-up and power-down sequences.

Figure 9–1 shows the Stratix IV E device's I/O pin circuitry.

Figure 9–1. Hot-Socketing Circuitry for Stratix IV E Devices



The POR circuit monitors the voltage level of the power supplies (V_{CC}, V_{CCPT}, V_{CCPD}, and V_{CCPGM}) and keeps the I/O pins tri-stated until the device is in user mode. The weak pull-up resistor (R) in the Stratix IV E input/output element (IOE) keeps the I/O pins from floating. The 3.0-V tolerance control circuit permits the I/O pins to be driven by 3.0 V before the V_{CCIO}, V_{CC}, V_{CCPD}, and/or V_{CCPGM} supplies are powered. It also prevents the I/O pins from driving out when the device is not in user mode.

☞ V_{CCPT}, V_{CCD_PLL}, V_{CCA_PLL}, V_{CC_AUX}, and V_{CC_CLKIN} are not monitored by POR, and thus have no affect on the device configuration.

Power-On Reset Circuitry

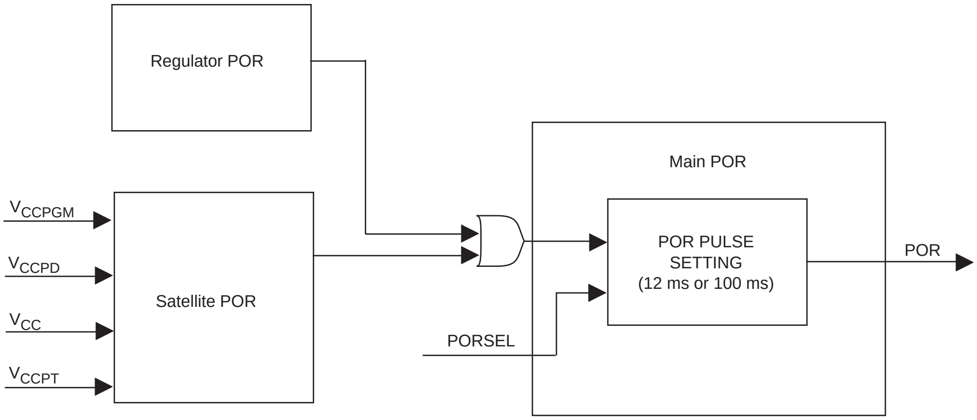
When power is applied to a Stratix IV device, a POR event occurs if the power supply reaches the recommended operating range within the maximum power supply ramp time (t_{RAMP}). If t_{RAMP} is not met, the device I/O pins and programming registers remain tri-stated, during which device configuration could fail. The maximum t_{RAMP} for Stratix IV devices is 100 ms; the minimum t_{RAMP} is 50 µs. Stratix IV devices provide a dedicated input pin (PORSEL) to select a POR delay time during power up. When the PORSEL pin is connected to GND, the POR delay time is 100 ms – 300 ms. When the PORSEL pin is set to high, the POR delay time is 4 ms – 12 ms.

The POR block consists of a regulator POR, satellite POR, and main POR to check the power supply levels for proper device configuration. The satellite POR monitors V_{CCPD} and V_{CCPGM} power supplies that are used in the IO buffers and for device programming. The satellite POR also monitors the V_{CC} and V_{CCPT} power supplies that are used in the device core. The POR block also checks for functionality of I/O level shifters powered by V_{CCPD} and V_{CCPGM} during power-up mode. The main POR waits for satellite POR and the regulator POR to release the POR signal. Until the release of the POR signal, the device configuration cannot start.

The internal configuration memory supply that is used during device configuration, is checked by the regulator POR block and is gated in the main POR block for the final POR trip. A simplified diagram of the POR block is shown in Figure 9–2.

☞ All configuration-related dedicated and dual function I/O pins must be powered by V_{CCPGM}.

Figure 9–2. Simplified POR Diagram for Stratix IV Devices



Power-On Reset Specifications

The POR circuit monitors the power supplies listed in Table 9–1.

Table 9–1. Power Supplies Monitored by the POR Circuitry

Power Supply	Description	Setting (V)
V_{CC}	Core and periphery power supply	0.9
V_{CCPT}	Programmable power technology power supply	1.5
V_{CCPD}	I/O pre-driver power supply	2.5, 3.0
V_{CCPGM}	Configuration pins power supply	1.8, 2.5, 3.0

The POR circuit does not monitor the power supplies listed in Table 9–2.

Table 9–2. Power Supplies Not Monitored by the POR Circuitry

Power Supply	Description	Setting (V)
V_{CCIO}	I/O power supply	1.2, 1.5, 1.8, 2.5, 3.0
V_{CCA}	PLL analog global power supply	2.5
V_{CCD_PLL}	PLL digital power supply	0.9
V_{CC_CLKIN}	PLL differential clock input power supply (top and bottom I/O banks only)	2.5
V_{CCBAT}	Battery back-up power supply for design security volatile key storage	2.5

The POR specification is designed to ensure that all the circuits in the Stratix IV device are at certain known states during power up.

The POR signal pulse width is programmable using the PORSEL input pin. When PORSEL is set to low, the POR signal pulse width is set to 100 ms. When the PORSEL is set to high, the POR signal pulse width is set to 12 ms.

For more information about the POR specification, refer to the *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook*.

Conclusion

Stratix IV E devices are hot-socketing compliant and allow successful device power-up without the need for power sequencing. The Stratix IV POR circuitry keeps the devices in the reset state until the power supply voltage levels are within operating range.

Referenced Documents

This chapter references the following documents:

- *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook*
- *Hot-Socketing and Power Sequencing Feature and Testing for Altera Devices* white paper

Revision History

Table 9–3 shows the revision history for this document.

Table 9–3. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	■ Updated "Hot-Socketing Feature Implementation in Stratix IV E Devices" on page 9–3 ■ Updated "Power-On Reset Circuitry" on page 9–4 ■ Updated Table 9–1 ■ Made minor editorial changes	—
May 2008 v1.0	Initial release.	—

10. Configuration, Design Security, and Remote System Upgrades in Stratix IV Devices

SIV51010-2.0

Introduction

This chapter contains complete information about Stratix® IV supported configuration schemes, instructions about how to execute the required configuration schemes, and the necessary option pin settings.

Stratix IV devices use SRAM cells to store configuration data. As SRAM memory is volatile, you must download configuration data to the Stratix IV device each time the device powers up. You can configure Stratix IV devices using one of four configuration schemes:

- Fast passive parallel (FPP)
- Fast active serial (AS)
- Passive serial (PS)
- Joint Test Action Group (JTAG)

All configuration schemes use either an external controller (for example, a MAX® II device or microprocessor), a configuration device, or a download cable. Refer to "Configuration Features" on page 10–3 for more information.

This chapter includes the following sections:

- "Configuration Schemes" on page 10–2
- "Configuration Features" on page 10–3
- "Fast Passive Parallel Configuration" on page 10–5
- "Fast Active Serial Configuration (Serial Configuration Devices)" on page 10–14
- "Passive Serial Configuration" on page 10–22
- "JTAG Configuration" on page 10–31
- "Device Configuration Pins" on page 10–37
- "Configuration Data Decompression" on page 10–43
- "Remote System Upgrades" on page 10–45
- "Remote System Upgrade Mode" on page 10–49
- "Dedicated Remote System Upgrade Circuitry" on page 10–51
- "Quartus II Software Support" on page 10–56
- "Design Security" on page 10–57

Configuration Devices

Altera® serial configuration devices support a single-device and multi-device configuration solution for Stratix IV devices and are used in the fast AS configuration scheme. Serial configuration devices offer a low-cost, low pin-count configuration solution.

For information about serial configuration devices, refer to the *Serial Configuration Devices (EPCS1, EPCS4, EPCS16, EPCS64, and EPCS128) Data Sheet* in volume 2 of the *Configuration Handbook*.

All minimum timing information in this chapter covers the entire Stratix IV family. Some devices may work at less than the minimum timing stated in this handbook due to process variation.

Configuration Schemes

Select the configuration scheme by driving the Stratix IV device MSEL pins either high or low, as shown in Table 10–1. The MSEL input buffers are powered by the V_{CC} power supply. Altera recommends you hardwire the MSEL[] pins to V_{CCPGM} and GND. The MSEL[2..0] pins have 5-kΩ internal pull-down resistors that are always active. During power-on reset (POR) and during reconfiguration, the MSEL pins have to be at LVTTL V_{IL} and V_{IH} levels to be considered logic low and logic high.

To avoid problems with detecting an incorrect configuration scheme, hardwire the MSEL[] pins to V_{CCPGM} and GND without pull-up or pull-down resistors. Do not drive the MSEL[] pins by a microprocessor or another device.

Table 10–1. Stratix IV Configuration Schemes

Configuration Scheme	MSEL2	MSEL1	MSEL0
Fast passive parallel	0	0	0
Passive serial	0	1	0
Fast AS (40 MHz) *(1)*	0	1	1
Remote system upgrade fast AS (40 MHz) *(1)*	0	1	1
FPP with design security feature and/or decompression enabled *(2)*	0	0	1
JTAG-based configuration *(4)*	*(3)*	*(3)*	*(3)*

Notes to Table 10–1:

(1) Stratix IV devices only support fast AS configuration. You must use either EPCS16, EPCS64, or EPCS128 devices to configure a Stratix IV device.

(2) These modes are only supported when using a MAX II device or a microprocessor with flash memory for configuration. In these modes, the host system must output a DCLK that is ×4 the data rate.

(3) Do not leave the MSEL pins floating, connect them to V_{CCPGM} or GND. These pins support the non-JTAG configuration scheme used in production. If you only use the JTAG configuration, connect the MSEL pins to GND.

(4) JTAG-based configuration takes precedence over other configuration schemes, which means MSEL pin settings are ignored. JTAG-based configuration does not support the design security or decompression features.

Table 10–2 shows the uncompressed raw binary file (**.rbf**) configuration file sizes for Stratix IV devices.

Table 10–2. Stratix IV Uncompressed Raw Binary File (.rbf) Sizes (Part 1 of 2) *(Note 1)*

Device	Data Size (Mbits)	Data Size (MBytes)
EP4SE110	53	6.625
EP4SE230	104	13.0

Table 10–2. Stratix IV Uncompressed Raw Binary File (.rbf) Sizes (Part 2 of 2) *(Note 1)*

Device	Data Size (Mbits)	Data Size (MBytes)
EP4SE290	141	17.625
EP4SE360	141	17.625
EP4SE530	188	23.5
EP4SE680	234	29.25
EP4SGX70	53	6.625
EP4SGX110	53	6.625
EP4SGX230	104	13
EP4SGX290	141	17.625
EP4SGX360	141	17.625
EP4SGX530	188	23.5

Note to Table 10–2:

(1) These values are preliminary.

Use the data in Table 10–2 to estimate the file size before design compilation. Different configuration file formats, such as a hexidecimal (**.hex**) or tabular text file (**.ttf**) format, have different file sizes. Refer to the Quartus® II software for the different types of configuration file and file sizes. However, for any specific version of the Quartus II software, any design targeted for the same device will have the same uncompressed configuration file size. If you are using compression, the file size can vary after each compilation because the compression ratio is dependent on the design.

For more information about setting device configuration options or creating configuration files, refer to the *Device Configuration Options* and *Configuration File Formats* chapters in volume 2 of the *Configuration Handbook*.

Configuration Features

Stratix IV devices offer design security, decompression, and remote system upgrade features. Design security using configuration bitstream encryption is available in Stratix IV devices, which protects your designs. Stratix IV devices can receive a compressed configuration bitstream and decompress this data in real-time, reducing storage requirements and configuration time. You can make real-time system upgrades from remote locations of your Stratix IV designs with the remote system upgrade feature.

Table 10–3 summarizes which configuration features you can use in each configuration scheme.

Table 10–3. Stratix IV Configuration Features (Part 1 of 2)

Configuration Scheme	Configuration Method	Decompression	Design Security	Remote System Upgrade
FPP	MAX II device or a microprocessor with flash memory	✓ *(1)*	✓ *(1)*	—
Fast AS	Serial configuration device	✓	✓	✓ *(2)*

Table 10–3. Stratix IV Configuration Features (Part 2 of 2)

Configuration Scheme	Configuration Method	Decompression	Design Security	Remote System Upgrade
PS	MAX II device or a microprocessor with flash memory	✓	✓	—
	Download cable	✓	✓	—
JTAG	MAX II device or a microprocessor with flash memory	—	—	—
	Download cable	—	—	—

Notes to Table 10–3:

(1) In these modes, the host system must send a DCLK that is ×4 the data rate.

(2) Remote system upgrade is only available in the fast AS configuration scheme. Only remote update mode is supported when using the fast AS configuration scheme. Local update mode is not supported.

You can also refer to the following:

- For more information about the configuration data decompression feature, refer to "Configuration Data Decompression" on page 10–43.
- For more information about the remote system upgrade feature, refer to "Remote System Upgrades" on page 10–45.
- For more information about the design security feature, refer to "Design Security" on page 10–57.

For more information about PFL, refer to *AN 386: Using the MAX II Parallel Flash Loader with the Quartus II Software*.

If your system already contains a common flash interface (CFI) flash memory, you can use it for Stratix IV device configuration storage as well. The MAX II parallel flash loader (PFL) feature in MAX II devices provides an efficient method to program CFI flash memory devices through the JTAG interface and the logic to control configuration from the flash memory device to the Stratix IV device. Both PS and FPP configuration modes are supported using this PFL feature.

For more information about programming Altera serial configuration devices, refer to "Programming Serial Configuration Devices" on page 10–20.

Power-On Reset Circuit

The POR circuit keeps the entire system in reset until the power supply voltage levels have stabilized on power-up. Upon power-up, the device does not release nSTATUS until V_{CCPT}, V_{CC}, V_{CCPD}, and V_{CCPGM} are above the device's POR trip point. On power down, brown-out occurs if the V_{CC}, or V_{CCPT} ramps down below the POR trip point and if V_{CC}, V_{CCPD}, or V_{CCPGM} drops below the threshold level of the hot socket circuitry.

In Stratix IV devices, a pin-selectable option (PORSEL) is provided that allows you to select between standard POR time or a fast POR time. When PORSEL is driven low, the standard POR time is 100 ms < T_{POR} < 300 ms, which has a lower power-ramp rate. When PORSEL is driven high, the fast POR time is 4 ms < T_{POR} < 12 ms. In both cases, you can extend the POR time by using an external component to assert the nSTATUS pin low.

V_{CCPGM} Pins

Stratix IV devices have a power supply, V_{CCPGM}, for all the dedicated configuration pins and dual function pins. The supported configuration voltage is 1.8 V, 2.5 V, and 3.0 V. Stratix IV devices do not support 1.5 V configuration.

Use the V_{CCPGM} pin to power all dedicated configuration inputs, dedicated configuration outputs, dedicated configuration bidirectional pins, and some of the dual functional pins that you use for configuration. With V_{CCPGM}, configuration input buffers do not have to share power lines with the regular I/O buffer in Stratix IV devices.

The operating voltage for the configuration input pin is independent of the I/O banks power supply V_{CCIO} during configuration. Therefore, no configuration voltage constraints on V_{CCIO} are needed in Stratix IV devices.

V_{CCPD} Pins

Stratix IV devices have a dedicated programming power supply, V_{CCPD}, which must be connected to 3.0 V/2.5 V to power the I/O pre-drivers, JTAG input and output pins (TCK, TMS, TDI, TDO and TRST), and design security circuitry.

☞ V_{CCPGM} and V_{CCPD} must ramp up from 0 V to the desired voltage level within 100 ms when PORSEL is low or 4 ms when PORSEL is high. If these supplies are not ramped up within this specified time, your Stratix IV device will not configure successfully. If your system cannot ramp up the power supplies within 100 ms or 4 ms, you must hold nCONFIG low until all power supplies are stable.

☞ V_{CCPD} must be greater than or equal to V_{CCIO}.

For more information about configuration pins power supply, refer to "Device Configuration Pins" on page 10–37.

Fast Passive Parallel Configuration

Fast passive parallel configuration in Stratix IV devices is designed to meet the continuously increasing demand for faster configuration times. Stratix IV devices are designed with the capability of receiving byte-wide configuration data per clock cycle. Table 10–4 shows the MSEL pin settings when using the FPP configuration scheme.

Table 10–4. Stratix IV MSEL Pin Settings for FPP Configuration Schemes

Configuration Scheme	MSEL2	MSEL1	MSEL0
FPP	0	0	0
FPP with the design security feature and/or decompression enabled *(1)*	0	0	1

Note to Table 10–4:

(1) These modes are only supported when using a MAX II device or a microprocessor with flash memory for configuration. In these modes, the host system must output a DCLK that is ×4 the data rate.

You can perform FPP configuration of Stratix IV devices using an intelligent host, such as a MAX II device or a microprocessor.

FPP Configuration Using a MAX II Device as an External Host

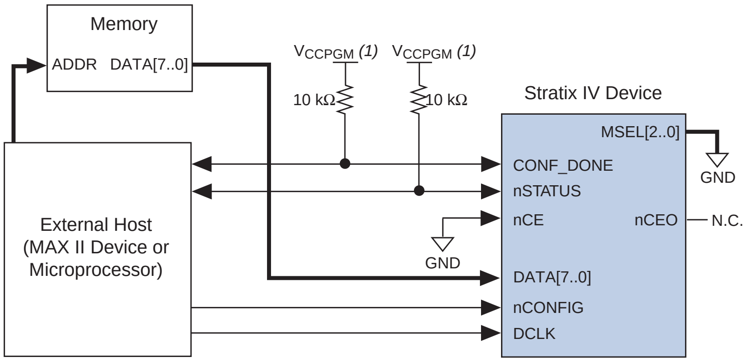
FPP configuration using compression and an external host provides the fastest method to configure Stratix IV devices. In this configuration scheme, you can use a MAX II device as an intelligent host that controls the transfer of configuration data from a storage device, such as flash memory, to the target Stratix IV device. You can store configuration data in **.rbf**, **.hex**, or **.ttf** format. When using the MAX II devices as an intelligent host, a design that controls the configuration process, such as fetching the data from flash memory and sending it to the device, must be stored in the MAX II device.

☞ If you are using the Stratix IV decompression and/or design security features, the external host must be able to send a DCLK frequency that is ×4 the data rate.

The ×4 DCLK signal does not require an additional pin and is sent on the DCLK pin. The maximum DCLK frequency is 125 MHz, which results in a maximum data rate of 250 Mbps. If you are not using the Stratix IV decompression or design security features, the data rate is ×8 the DCLK frequency.

Figure 10–1 shows the configuration interface connections between the Stratix IV device and a MAX II device for single device configuration.

Figure 10–1. Single Device FPP Configuration Using an External Host



Note to Figure 10–1:

(1) Connect the resistor to a supply that provides an acceptable input signal for the Stratix IV device. V_{CCPGM} needs to be high enough to meet the V_{IH} specification of the I/O on the device and the external host. Altera recommends that you power up all configuration system I/Os with V_{CCPGM}.

Upon power-up, the Stratix IV device goes through a POR. The POR delay is dependent on the PORSEL pin setting. When PORSEL is driven low, the standard POR time is 100 ms < T_{POR} < 300 ms. When PORSEL is driven high, the fast POR time is 4 ms < T_{POR} < 12 ms. During POR, the device resets, holds nSTATUS low, and tri-states all user I/O pins. Once the device successfully exits POR, all user I/O pins continue to be tri-stated. If nIO_pullup is driven low during power up and configuration, the user I/O pins and dual-purpose I/O pins have weak pull-up resistors, which are on (after POR) before and during configuration. If nIO_pullup is driven high, the weak pull-up resistors are disabled.

The configuration cycle consists of three stages: reset, configuration, and initialization. While nCONFIG or nSTATUS are low, the device is in the reset stage. To initiate configuration, the MAX II device must drive the nCONFIG pin from low to high.

☞ To begin the configuration process, you must fully power V_{CC}, V_{CCIO}, V_{CCPGM}, and V_{CCPD} of the banks where the configuration and JTAG pins reside to the appropriate voltage levels.

When nCONFIG goes high, the device comes out of reset and releases the open-drain nSTATUS pin, which is then pulled high by an external 10-kΩ pull-up resistor. Once nSTATUS is released, the device is ready to receive configuration data and the configuration stage begins. When nSTATUS is pulled high, the MAX II device places the configuration data one byte at a time on the DATA[7..0] pins.

☞ Stratix IV devices receive configuration data on the DATA[7..0] pins and the clock is received on the DCLK pin. Data is latched into the device on the rising edge of DCLK. If you are using the Stratix IV decompression and/or design security features, configuration data is latched on the rising edge of every fourth DCLK cycle. After the configuration data is latched in, it is processed during the following three DCLK cycles. Therefore, you can only stop DCLK after three clock cycles after the last data is latched into the Stratix IV Devices.

Data is continuously clocked into the target device until CONF_DONE goes high. The CONF_DONE pin goes high one byte early in parallel configuration (FPP) modes. The last byte is required for serial configuration (AS and PS) modes. After the device has received the next-to-last byte of the configuration data successfully, it releases the open-drain CONF_DONE pin, which is pulled high by an external 10-kΩ pull-up resistor. A low-to-high transition on CONF_DONE indicates configuration is complete and initialization of the device can begin. The CONF_DONE pin must have an external 10-kΩ pull-up resistor for the device to initialize.

In Stratix IV devices, the initialization clock source is either the internal oscillator or the optional CLKUSR pin. By default, the internal oscillator is the clock source for initialization. If you use the internal oscillator, the Stratix IV device provides itself with enough clock cycles for proper initialization. Therefore, if the internal oscillator is the initialization clock source, sending the entire configuration file to the device is sufficient to configure and initialize the device. Driving DCLK to the device after configuration is complete does not affect device operation.

You can also synchronize initialization of multiple devices or delay initialization with the CLKUSR option. You can turn on the **Enable user-supplied start-up clock** (**CLKUSR**) option in the Quartus II software from the **General** tab of the **Device and Pin Options** dialog box. Supplying a clock on CLKUSR does not affect the configuration process. The CONF_DONE pin goes high one byte early in parallel configuration (FPP) modes. The last byte is required for serial configuration (AS and PS) modes. After the CONF_DONE pin transitions high, CLKUSR is enabled after the time specified as t_{CD2CU}. After this time period elapses, Stratix IV devices require 8532 clock cycles to initialize properly and enter user mode. Stratix IV devices support a CLKUSR f_{MAX} of 125 MHz.

An optional INIT_DONE pin is available, which signals the end of initialization and the start of user-mode with a low-to-high transition. This **Enable INIT_DONE Output** option is available in the Quartus II software from the **General** tab of the **Device and Pin Options** dialog box. If you use the INIT_DONE pin, it is high because of an external 10-kΩ pull-up resistor when nCONFIG is low and during the beginning of configuration. Once the option bit to enable INIT_DONE is programmed into the device (during the first frame of configuration data), the INIT_DONE pin goes low.

When initialization is complete, the INIT_DONE pin is released and pulled high. The MAX II device must be able to detect this low-to-high transition, which signals the device has entered user mode. When initialization is complete, the device enters user mode. In user-mode, the user I/O pins no longer have weak pull-up resistors and function as assigned in your design.

To ensure DCLK and DATA[7..0] are not left floating at the end of configuration, the MAX II device must drive them either high or low, whichever is convenient on your board. The DATA[7..0] pins are available as user I/O pins after configuration. When you select the FPP scheme as a default in the Quartus II software, these I/O pins are tri-stated in user mode. To change this default option in the Quartus II software, select the **Dual-Purpose Pins** tab of the **Device & Pin Options** dialog box.

The configuration clock (DCLK) speed must be below the specified frequency to ensure correct configuration. No maximum DCLK period exists, which means you can pause configuration by halting DCLK for an indefinite amount of time.

☞ If you are using the Stratix IV decompression and/or design security features and need to stop DCLK, it can only be stopped three clock cycles after the last data byte was latched into the Stratix IV device.

By stopping DCLK, the configuration circuit allows enough clock cycles to process the last byte of latched configuration data. When the clock restarts, the MAX II device must provide data on the DATA[7..0] pins prior to sending the first DCLK rising edge.

If an error occurs during configuration, the device drives its nSTATUS pin low, resetting itself internally. The low signal on the nSTATUS pin also alerts the MAX II device that there is an error. If the **Auto-restart configuration after error** option (available in the Quartus II software from the **General** tab of the **Device and Pin Options** dialog box) is turned on, the device releases nSTATUS after a reset time-out period (maximum of 500 µs). After nSTATUS is released and pulled high by a pull-up resistor, the MAX II device can try to reconfigure the target device without needing to pulse nCONFIG low. If this option is turned off, the MAX II device must generate a low-to-high transition (with a low pulse of at least 2 µs) on nCONFIG to restart the configuration process.

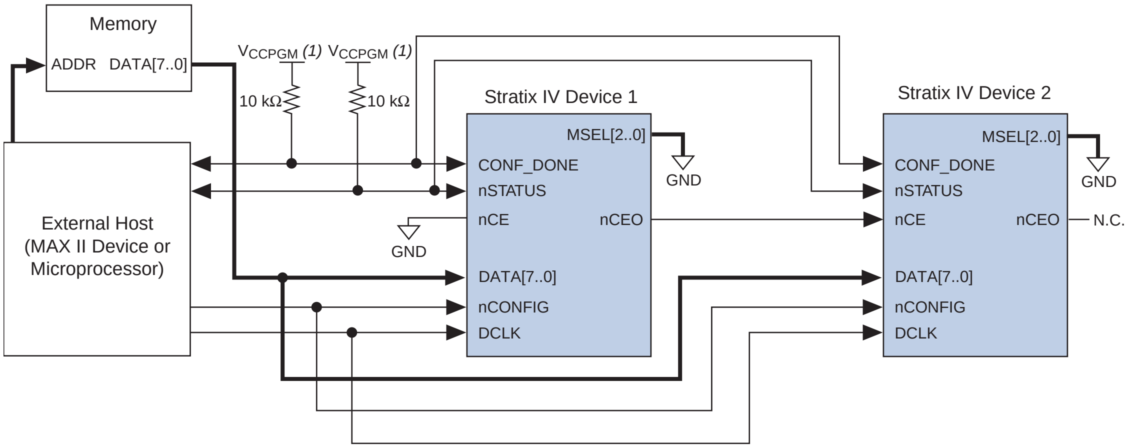
The MAX II device can also monitor the CONF_DONE and INIT_DONE pins to ensure successful configuration. The MAX II device must monitor the CONF_DONE pin to detect errors and determine when programming completes. If all configuration data is sent, but the CONF_DONE or INIT_DONE signals have not gone high, the MAX II device reconfigures the target device.

☞ If you use the optional CLKUSR pin and nCONFIG is pulled low to restart the configuration during device initialization, ensure CLKUSR continues toggling during the time nSTATUS is low (maximum of 500 µs).

When the device is in user mode, initiating a reconfiguration is done by transitioning the nCONFIG pin low-to-high. The nCONFIG pin needs to be low for at least 2 µs. When nCONFIG is pulled low, the device also pulls nSTATUS and CONF_DONE low and all I/O pins are tri-stated. Once nCONFIG returns to a logic high level and nSTATUS is released by the device, reconfiguration begins.

Figure 10–2 shows how to configure multiple devices using a MAX II device. This circuit is similar to the FPP configuration circuit for a single device, except the Stratix IV devices are cascaded for multi-device configuration.

Figure 10–2. Multi-Device FPP Configuration Using an External Host



Note to Figure 10–2:

(1) Connect the pull-up resistor to a supply that provides an acceptable input signal for all Stratix IV devices in the chain. V_{CCPGM} needs to be high enough to meet the V_{IH} specification of the I/O standard on the device and the external host. Altera recommends you power up all configuration system's I/Os with $V_{CCPGM.}$

In a multi-device FPP configuration, the first device's nCE pin is connected to GND while its nCEO pin is connected to nCE of the next device in the chain. The last device's nCE input comes from the previous device, while its nCEO pin is left floating. After the first device completes configuration in a multi-device configuration chain, its nCEO pin drives low to activate the second device's nCE pin, which prompts the second device to begin configuration. The second device in the chain begins configuration within one clock cycle; therefore, the transfer of data destinations is transparent to the MAX II device. All other configuration pins (nCONFIG, nSTATUS, DCLK, DATA[7..0], and CONF_DONE) are connected to every device in the chain. The configuration signals may require buffering to ensure signal integrity and prevent clock skew problems. Ensure that the DCLK and DATA lines are buffered for every fourth device. Because all device CONF_DONE pins are tied together, all devices initialize and enter user mode at the same time.

All nSTATUS and CONF_DONE pins are tied together; if any device detects an error, configuration stops for the entire chain and you must reconfigure the entire chain. For example, if the first device flags an error on nSTATUS, it resets the chain by pulling its nSTATUS pin low. This behavior is similar to a single device detecting an error.

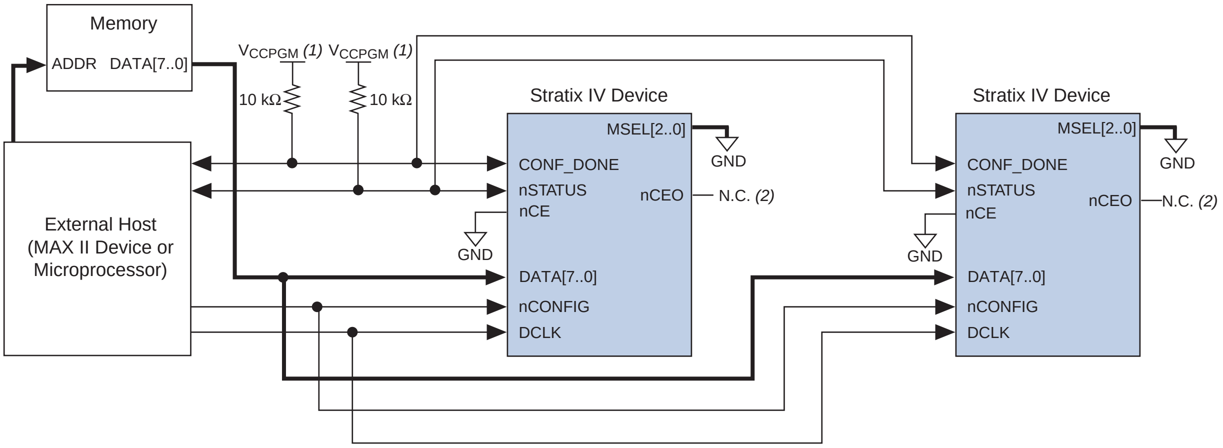
If the **Auto-restart configuration after error** option is turned on, the devices release their nSTATUS pins after a reset time-out period (maximum of 500 µs). After all nSTATUS pins are released and pulled high, the MAX II device tries to reconfigure the chain without pulsing nCONFIG low. If this option is turned off, the MAX II device must generate a low-to-high transition (with a low pulse of at least 2 µs) on nCONFIG to restart the configuration process.

In a multi-device FPP configuration chain, all Stratix IV devices in the chain must either enable or disable the decompression and/or design security features. You cannot selectively enable the decompression and/or design security features for each device in the chain because of the DATA and DCLK relationship. If the chain contains devices that do not support design security, use a serial configuration scheme.

If a system has multiple devices that contain the same configuration data, tie all device nCE inputs to GND and leave the nCEO pins floating. All other configuration pins (nCONFIG, nSTATUS, DCLK, DATA[7..0], and CONF_DONE) are connected to every device in the chain. Configuration signals may require buffering to ensure signal integrity and prevent clock skew problems. Ensure that the DCLK and DATA lines are buffered for every fourth device. Devices must be the same density and package. All devices start and complete configuration at the same time.

Figure 10–3 shows a multi-device FPP configuration when both Stratix IV devices are receiving the same configuration data.

Figure 10–3. Multiple-Device FPP Configuration Using an External Host When Both Devices Receive the Same Data



Notes to Figure 10–3:

(1) Connect the resistor to a supply that provides an acceptable input signal for all Stratix IV devices in the chain. V_{CCPGM} needs to be high enough to meet the V_{IH} specification of the I/O on the device and the external host. Altera recommends you power up all configuration system's I/Os with V_{CCPGM}.

(2) The nCEO pins of both Stratix IV devices are left unconnected when configuring the same configuration data into multiple devices.

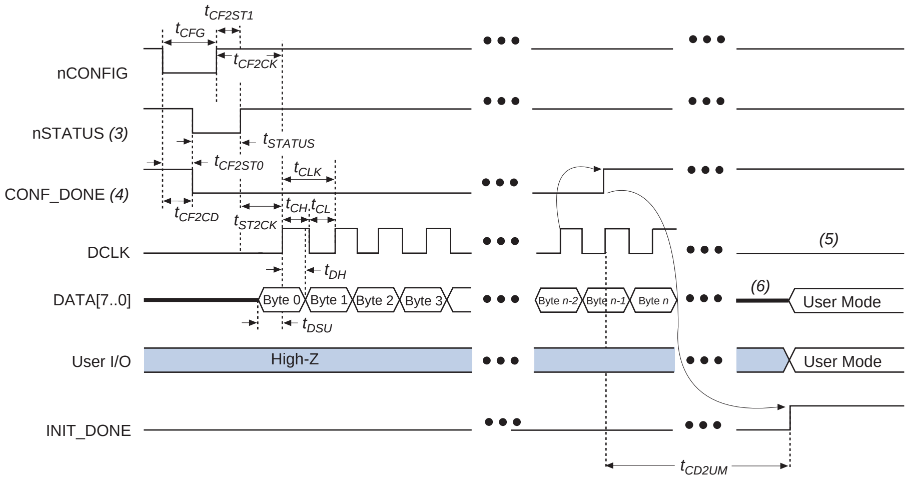
You can use a single configuration chain to configure Stratix IV devices with other Altera devices that support FPP configuration, such as other types of Stratix devices. To ensure that all devices in the chain complete configuration at the same time, or that an error flagged by one device initiates reconfiguration in all devices, tie all of the device CONF_DONE and nSTATUS pins together.

For more information about configuring multiple Altera devices in the same configuration chain, refer to the *Configuring Mixed Altera FPGA Chains* in volume 2 of the *Configuration Handbook*.

FPP Configuration Timing

Figure 10–4 shows the timing waveform for FPP configuration when using a MAX II device as an external host. This waveform shows the timing when the decompression and design security features are not enabled.

Figure 10–4. FPP Configuration Timing Waveform *(Note 1)*, *(2)*



Notes to Figure 10–4:

(1) Use this timing waveform when decompression and design security features are not used.

(2) The beginning of this waveform shows the device in user mode. In user mode, `nCONFIG`, `nSTATUS`, and `CONF_DONE` are at logic high levels. When `nCONFIG` is pulled low, a reconfiguration cycle begins.

(3) Upon power-up, the Stratix IV device holds `nSTATUS` low for the time of the POR delay.

(4) Upon power-up, before and during configuration, `CONF_DONE` is low.

(5) Do not leave `DCLK` floating after configuration. You can drive it high or low, whichever is more convenient.

(6) `DATA[7..0]` are available as user I/O pins after configuration. The state of these pins depends on the dual-purpose pin settings.

Table 10–5 defines the timing parameters for Stratix IV devices for FPP configuration when the decompression and design security features are not enabled.

Table 10–5. FPP Timing Parameters for Stratix IV Devices ***(Note 1), (2)*** ***(Part 1 of 2)***

Symbol	Parameter	Minimum	Maximum	Units
t_{CF2CD}	`nCONFIG` low to `CONF_DONE` low	—	800	ns
t_{CF2ST0}	`nCONFIG` low to `nSTATUS` low	—	800	ns
t_{CFG}	`nCONFIG` low pulse width	2	—	µs
t_{STATUS}	`nSTATUS` low pulse width	10	500 *(3)*	µs
t_{CF2ST1}	`nCONFIG` high to `nSTATUS` high	—	500 *(3)*	µs
t_{CF2CK}	`nCONFIG` high to first rising edge on `DCLK`	500	—	µs
t_{ST2CK}	`nSTATUS` high to first rising edge of `DCLK`	2	—	µs
t_{DSU}	Data setup time before rising edge on `DCLK`	4	—	ns
t_{DH}	Data hold time after rising edge on `DCLK`	0	—	ns
t_{CH}	`DCLK` high time	3.2	—	ns
t_{CL}	`DCLK` low time	3.2	—	ns
t_{CLK}	`DCLK` period	8	—	ns
f_{MAX}	`DCLK` frequency	—	125	MHz
t_R	Input rise time	—	40	ns
t	Input fall time	—	40	ns

Table 10–5. FPP Timing Parameters for Stratix IV Devices *(Note 1), (2)* *(Part 2 of 2)*

Symbol	Parameter	Minimum	Maximum	Units
t_{CD2UM}	CONF_DONE high to user mode *(4)*	55	150	µs
t_{CD2CU}	CONF_DONE high to CLKUSR enabled	4 × maximum DCLK period	—	—
t_{CD2UMC}	CONF_DONE high to user mode with CLKUSR option on	t_{CD2CU} + (8532 × CLKUSR period)	—	—

Notes to Table 10–5:

(1) This information is preliminary.

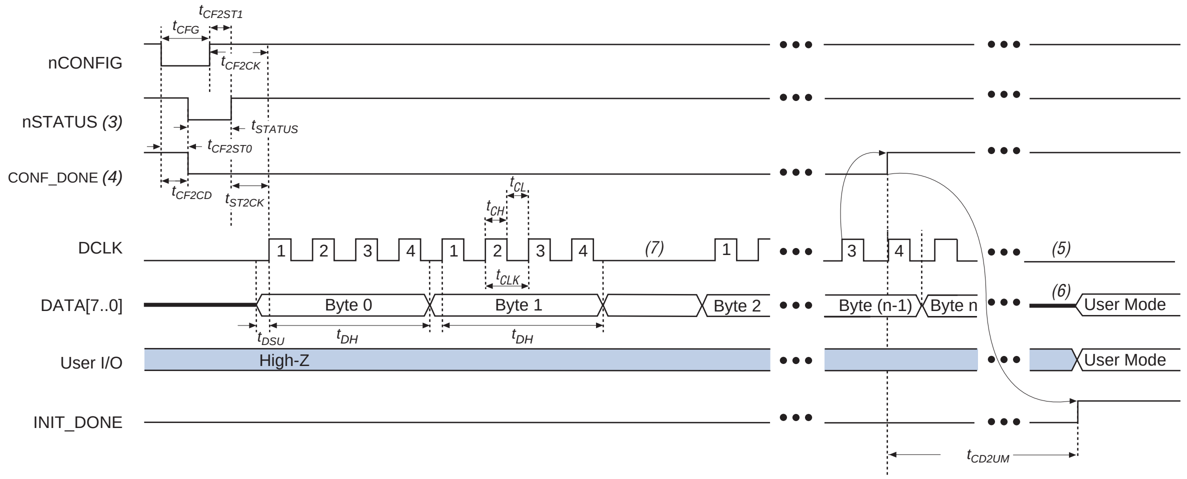
(2) Use these timing parameters when the decompression and design security features are not used.

(3) This value is obtainable if you do not delay configuration by extending the nCONFIG or nSTATUS low pulse width.

(4) The minimum and maximum numbers apply only if you chose the internal oscillator as the clock source for starting up the device.

Figure 10–5 shows the timing waveform for FPP configuration when using a MAX II device as an external host. This waveform shows the timing when the decompression and/or design security features are enabled.

Figure 10–5. FPP Configuration Timing Waveform with Decompression or Design Security Feature Enabled *(Note 1), (2)*



Notes to Figure 10–5:

(1) You need to use this timing waveform when the decompression and/or design security features are used.

(2) The beginning of this waveform shows the device in user-mode. In user-mode, nCONFIG, nSTATUS, and CONF_DONE are at logic high levels. When nCONFIG is pulled low, a reconfiguration cycle begins.

(3) Upon power-up, the Stratix IV device holds nSTATUS low for the time of the POR delay.

(4) Upon power-up, before and during configuration, CONF_DONE is low.

(5) Do not leave DCLK floating after configuration. You can drive it high or low, whichever is more convenient.

(6) DATA[7..0] are available as user I/O pins after configuration. The state of these pins depends on the dual-purpose pin settings.

(7) If needed, you can pause DCLK by holding it low. When DCLK restarts, the external host must provide data on the DATA[7..0] pins prior to sending the first DCLK rising edge.

Table 10–6 defines the timing parameters for Stratix IV devices for FPP configuration when the decompression and/or the design security features are enabled.

Table 10–6. FPP Timing Parameters for Stratix IV Devices with the Decompression or Design Security Features Enabled *(Note 1)*, *(2)*

Symbol	Parameter	Minimum	Maximum	Units
t_{CF2CD}	nCONFIG low to CONF_DONE low	—	800	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low	—	800	ns
t_{CFG}	nCONFIG low pulse width	2	—	µs
t_{STATUS}	nSTATUS low pulse width	10	500 *(3)*	µs
t_{CF2ST1}	nCONFIG high to nSTATUS high	—	500 *(3)*	µs
t_{CF2CK}	nCONFIG high to first rising edge on DCLK	500	—	µs
t_{ST2CK}	nSTATUS high to first rising edge of DCLK	2	—	µs
t_{DSU}	Data setup time before rising edge on DCLK	4	—	ns
t_{DH}	Data hold time after rising edge on DCLK	24	—	ns
t_{CH}	DCLK high time	3.2	—	ns
t_{CL}	DCLK low time	3.2	—	ns
t_{CLK}	DCLK period	8	—	ns
f_{MAX}	DCLK frequency	—	125	MHz
t_{DATA}	Data rate	—	250	Mbps
t_R	Input rise time	—	40	ns
t	Input fall time	—	40	ns
t_{CD2UM}	CONF_DONE high to user mode *(4)*	55	150	µs
t_{CD2CU}	CONF_DONE high to CLKUSR enabled	4 × maximum DCLK period	—	—
t_{CD2UMC}	CONF_DONE high to user mode with CLKUSR option on *(4)*	t_{CD2CU} + (8532 × CLKUSR period)	—	—

Notes to Table 10–6:

(1) This information is preliminary.

(2) Use these timing parameters when the decompression and design security features are used.

(3) This value is obtainable if you do not delay configuration by extending the nCONFIG or nSTATUS low pulse width.

(4) The minimum and maximum numbers apply only if you chose the internal oscillator as the clock source for starting up the device.

Device configuration options and how to create configuration files are discussed further in the *Device Configuration Options* and *Configuration File Formats* chapters in volume 2 of the *Configuration Handbook*.

FPP Configuration Using a Microprocessor

In this configuration scheme, a microprocessor can control the transfer of configuration data from a storage device, such as flash memory, to the target Stratix IV device.

All information in "FPP Configuration Using a MAX II Device as an External Host" on page 10–6 is also applicable when using a microprocessor as an external host. Refer to this section for all configuration and timing information.

Fast Active Serial Configuration (Serial Configuration Devices)

In the fast AS configuration scheme, Stratix IV devices are configured using a serial configuration device. These configuration devices are low-cost devices with non-volatile memory that feature a simple four-pin interface and a small form factor. These features make serial configuration devices an ideal low-cost configuration solution.

The largest serial configuration device currently supports 128 MBits of configuration bitstream. Use Stratix IV decompression features or select an FPP or PS configuration scheme for larger Stratix IV devices such as EP4SGX290 and EP4SGX360.

For more information about serial configuration devices, refer to the *Serial Configuration Devices (EPCS1, EPCS4, EPCS16, EPCS64, and EPCS128) Data Sheet* chapter in volume 2 of the *Configuration Handbook*.

Serial configuration devices provide a serial interface to access configuration data. During device configuration, Stratix IV devices read configuration data using the serial interface, decompress data if necessary, and configure their SRAM cells. This scheme is referred to as the AS configuration scheme because the Stratix IV device controls the configuration interface. This scheme contrasts with the PS configuration scheme, where the configuration device controls the interface.

The Stratix IV decompression and design security features are fully available when configuring your Stratix IV device using fast AS mode.

Table 10–7 shows the MSEL pin settings when using the AS configuration scheme.

Table 10–7. Stratix IV MSEL Pin Settings for AS Configuration Schemes *(Note 1)*

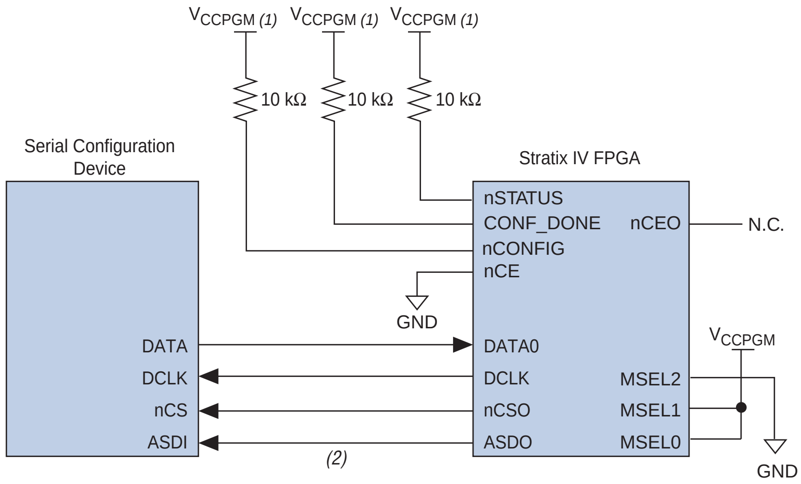
Configuration Scheme	MSEL2	MSEL1	MSEL0
Fast AS (40 MHz)	0	1	1
Remote system upgrade fast AS (40 MHz)	0	1	1

Note to Table 10–7:

(1) Use EPCS16, EPCS64, or EPCS128 devices.

Serial configuration devices have a four-pin interface: serial clock input (DCLK), serial data output (DATA), AS data input (ASDI), and an active-low chip select (nCS). This four-pin interface connects to Stratix IV device pins, as shown in Figure 10–6.

Figure 10–6. Single Device Fast AS Configuration



Notes to Figure 10–6:

(1) Connect the pull-up resistors to V_{CCPGM} at a 3.0-V supply.

(2) Stratix IV devices use the ASDO-to-ASDI path to control the configuration device.

You can power the EPCS serial configuration device with 3.0 V when you configure the Stratix IV FPGA using Active Serial (AS) Configuration mode. This is feasible because the power supply to the EPCS device ranges between 2.7 V and 3.6 V. You do not need a dedicated 3.3 V power supply to power the EPCS device. The EPCS device and the VCCPGM pins on the Stratix IV device may share the same 3.0 V power supply.

Upon power-up, the Stratix IV devices go through a POR. The POR delay is dependent on the PORSEL pin setting. When PORSEL is driven low, the standard POR time is 100 ms < T_{POR} < 300 ms. When PORSEL is driven high, the fast POR time is 4 ms < T_{POR} < 12 ms. During POR, the device resets, holds nSTATUS and CONF_DONE low, and tri-states all user I/O pins. Once the device successfully exits POR, all user I/O pins continue to be tri-stated. If nIO_pullup is driven low during power-up and configuration, the user I/O pins and dual-purpose I/O pins will have weak pull-up resistors, which are on (after POR) before and during configuration. If nIO_pullup is driven high, the weak pull-up resistors are disabled.

The configuration cycle consists of three stages: reset, configuration, and initialization. While nCONFIG or nSTATUS are low, the device is in reset. After POR, the Stratix IV device releases nSTATUS, which is pulled high by an external 10-kΩ pull-up resistor and enters configuration mode.

☞ To begin configuration, power the V_{CC}, V_{CCIO}, V_{CCPGM}, and V_{CCPD} voltages (for the banks where the configuration and JTAG pins reside) to the appropriate voltage levels.

The serial clock (DCLK) generated by the Stratix IV device controls the entire configuration cycle and provides timing for the serial interface. Stratix IV devices use an internal oscillator to generate DCLK. Using the MSEL[] pins, you can select to use a 40 MHz oscillator.

In fast AS configuration schemes, Stratix IV devices drive out control signals on the falling edge of DCLK. The serial configuration device responds to the instructions by driving out configuration data on the falling edge of DCLK. Then the data is latched into the Stratix IV device on the following falling edge of DCLK.

In configuration mode, Stratix IV devices enable the serial configuration device by driving the nCSO output pin low, which connects to the chip select (nCS) pin of the configuration device. The Stratix IV device uses the serial clock (DCLK) and serial data output (ASDO) pins to send operation commands and/or read address signals to the serial configuration device. The configuration device provides data on its serial data output (DATA) pin, which connects to the DATA0 input of the Stratix IV devices.

After all configuration bits are received by the Stratix IV device, it releases the open-drain CONF_DONE pin, which is pulled high by an external 10-kΩ resistor. Initialization begins only after the CONF_DONE signal reaches a logic high level. All AS configuration pins (DATA0, DCLK, nCSO, and ASDO) have weak internal pull-up resistors that are always active. After configuration, these pins are set as input tri-stated and are driven high by the weak internal pull-up resistors. The CONF_DONE pin must have an external 10-kΩ pull-up resistor in order for the device to initialize.

In Stratix IV devices, the initialization clock source is either the internal oscillator or the optional CLKUSR pin. By default, the internal oscillator is the clock source for initialization. If you use the internal oscillator, the Stratix IV device provides itself with enough clock cycles for proper initialization. You also have the flexibility to synchronize initialization of multiple devices or to delay initialization with the CLKUSR option. You can turn on the **Enable user-supplied start-up clock** (**CLKUSR**) option in the Quartus II software from the **General** tab of the **Device and Pin Options** dialog box. When you select the **Enable user supplied start-up clock** option, the CLKUSR pin is the initialization clock source. Supplying a clock on CLKUSR does not affect the configuration process. After all configuration data is accepted and CONF_DONE goes high, CLKUSR is enabled after four clock cycles of DCLK. After this time period elapses, Stratix IV devices require 8532 clock cycles to initialize properly and enter user mode. Stratix IV devices support a CLKUSR f_{MAX} of 125 MHz.

An optional INIT_DONE pin is available, which signals the end of initialization and the start of user-mode with a low-to-high transition. The **Enable INIT_DONE Output** option is available in the Quartus II software from the **General** tab of the **Device and Pin Options** dialog box. If you use the INIT_DONE pin, it is high due to an external 10-kΩ pull-up resistor when nCONFIG is low and during the beginning of configuration. Once the option bit to enable INIT_DONE is programmed into the device (during the first frame of configuration data), the INIT_DONE pin goes low. When initialization is complete, the INIT_DONE pin is released and pulled high. This low-to-high transition signals that the device has entered user mode. When initialization is complete, the device enters user mode. In user mode, the user I/O pins no longer have weak pull-up resistors and function as assigned in your design.

If an error occurs during configuration, Stratix IV devices assert the nSTATUS signal low, indicating a data frame error, and the CONF_DONE signal stays low. If the **Auto-restart configuration after error** option (available in the Quartus II software from the **General** tab of the **Device and Pin Options** dialog box) is turned on, the Stratix IV device resets the configuration device by pulsing nCSO, releases nSTATUS after a reset time-out period (maximum of 500 µs), and retries configuration. If this option is turned off, the system must monitor nSTATUS for errors and then pulse nCONFIG low for at least 2 µs to restart configuration.

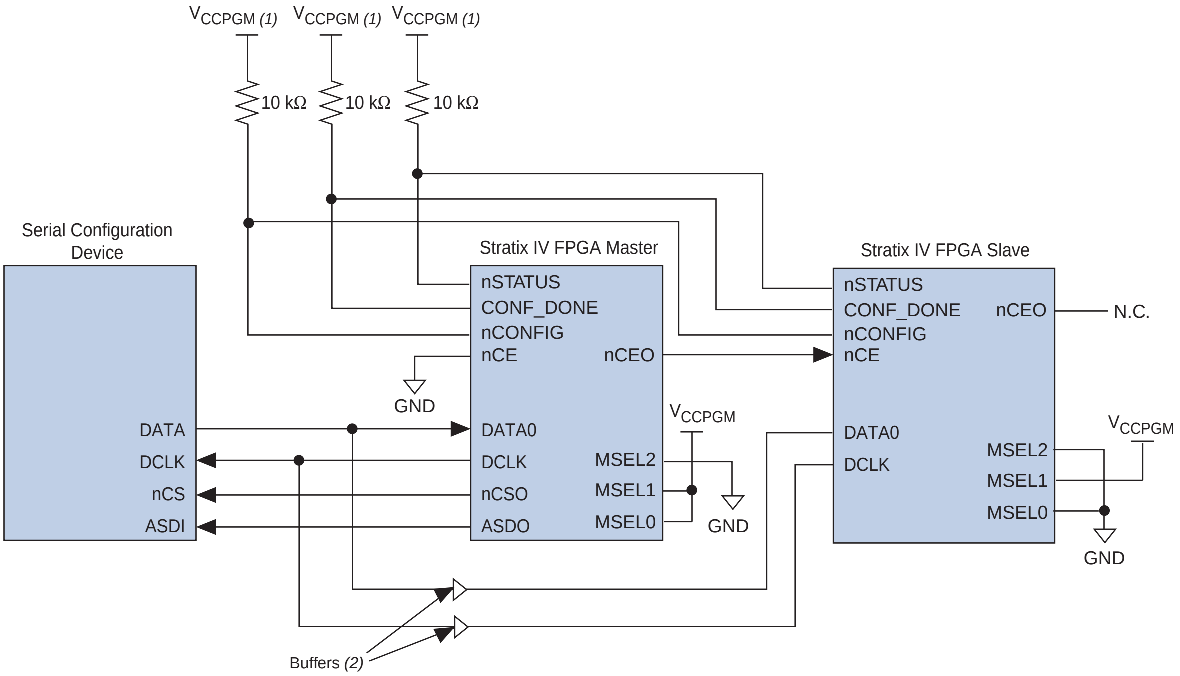
When the Stratix IV device is in user mode, you can initiate reconfiguration by pulling the nCONFIG pin low. The nCONFIG pin should be low for at least 2 µs. When nCONFIG is pulled low, the device also pulls nSTATUS and CONF_DONE low and all I/O pins are tri-stated. Once nCONFIG returns to a logic high level and nSTATUS is released by the Stratix IV device, reconfiguration begins.

You can configure multiple Stratix IV devices using a single serial configuration device. You can cascade multiple Stratix IV devices using the chip-enable (nCE) and chip-enable-out (nCEO) pins. The first device in the chain must have its nCE pin connected to GND. You must connect its nCEO pin to the nCE pin of the next device in the chain. When the first device captures all of its configuration data from the bitstream, it drives the nCEO pin low, enabling the next device in the chain. You must leave the nCEO pin of the last device unconnected. The nCONFIG, nSTATUS, CONF_DONE, DCLK, and DATA0 pins of each device in the chain are connected (refer to Figure 10–7).

The first Stratix IV device in the chain is the configuration master and controls configuration of the entire chain. You must connect its MSEL pins to select the AS configuration scheme. The remaining Stratix IV devices are configuration slaves. You must connect their MSEL pins to select the PS configuration scheme. Any other Altera device that supports PS configuration can also be part of the chain as a configuration slave.

Figure 10–7 shows the pin connections for the multi-device fast AS configuration.

Figure 10–7. Multi-Device Fast AS Configuration



Notes to Figure 10–7:

(1) Connect the pull-up resistors to a V_{CCPGM} at a 3.0-V supply.

(2) Connect the repeater buffers between the Stratix IV master and slave device(s) for DATA[0] and DCLK. This is to prevent any potential signal integrity and clock skew problems.

As shown in Figure 10–7, the nSTATUS and CONF_DONE pins on all target devices are connected together with external pull-up resistors. These pins are open-drain bidirectional pins on the devices. When the first device asserts nCEO (after receiving all of its configuration data), it releases its CONF_DONE pin. But the subsequent devices in the chain keep this shared CONF_DONE line low until they have received their configuration data. When all target devices in the chain have received their configuration data and have released CONF_DONE, the pull-up resistor drives a high level on this line and all devices simultaneously enter initialization mode.

If an error occurs at any point during configuration, the nSTATUS line is driven low by the failing device. If you enable the **Auto-restart configuration after error** option, reconfiguration of the entire chain begins after a reset time-out period (maximum of 500 µs). If the **Auto-restart configuration after error** option is turned off, the external system must monitor nSTATUS for errors and then pulse nCONFIG low to restart configuration. The external system can pulse nCONFIG if it is under system control rather than tied to V_{CCGPM}.

☞ While you can cascade Stratix IV devices, you cannot cascade or chain together serial configuration devices.

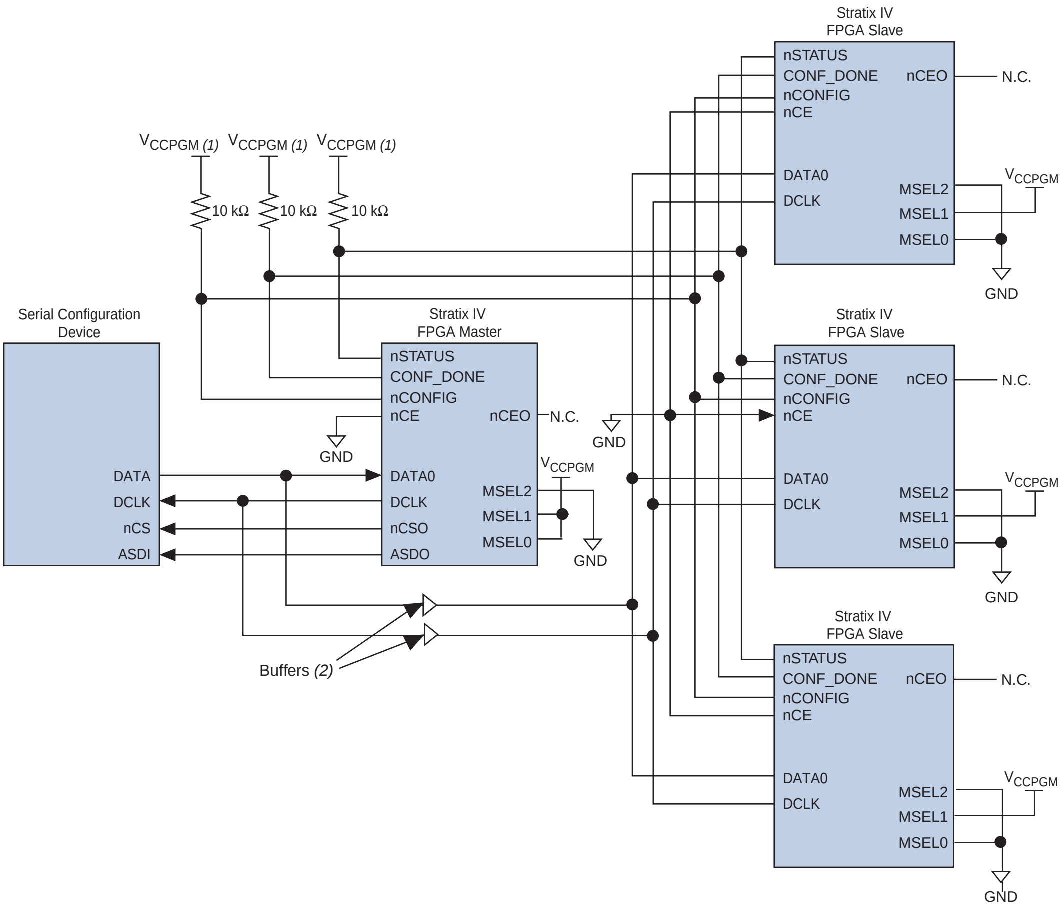
If the configuration bitstream size exceeds the capacity of a serial configuration device, you must select a larger configuration device and/or enable the compression feature. When configuring multiple devices, the size of the bitstream is the sum of the individual device configuration bitstreams.

A system may have multiple devices that contain the same configuration data. In active serial chains, you can implement this by storing one copy of the **.sof** in the serial configuration device. The same copy of the **.sof** configures the master Stratix IV device and all remaining slave devices concurrently. All Stratix IV devices must be the same density and package.

To configure four identical Stratix IV devices with the same **.sof**, you can set up the chain as shown in Figure 10–8. The first device is the master device and its MSEL pins need to be set to select AS configuration. The other three slave devices are set up for concurrent configuration and their MSEL pins need to be set to select PS configuration. The nCE input pins from the master and slave are connected to GND, and the DATA and DCLK pins connect in parallel to all four devices. During the configuration cycle, the master device reads its configuration data from the serial configuration device and transmits the configuration data to all three slave devices, configuring all of them simultaneously.

Figure 10–8 shows the multi-device fast AS configuration when the devices receive the same data using single **.sof**.

Figure 10–8. Multi-Device Fast AS Configuration When the Devices Receive the Same Data Using Single **.sof** Files



Notes to Figure 10–8:

(1) Connect the pull-up resistors to a V_{CCPGM} at 3.0-V supply.

(2) Connect the repeater buffers between the Stratix IV master and slave device(s) for DATA[0] and DCLK. This is to prevent any potential signal integrity and clock skew problems.

Estimating Active Serial Configuration Time

Active serial configuration time is dominated by the time it takes to transfer data from the serial configuration device to the Stratix IV device. This serial interface is clocked by the Stratix IV DCLK output (generated from an internal oscillator). Because the Stratix IV device only supports fast AS configuration, the DCLK frequency needs to be set to **40 MHz (25 ns)**.

Therefore, the minimum configuration time estimate for an EP4SE110 device (53.0 MBits of uncompressed data) is:

RBF Size × (minimum DCLK period / 1 bit per DCLK cycle) = estimated minimum configuration time

53 Mbits × (25 ns / 1 bit) = 1325 ms

☞ The calculation above is based on preliminary uncompressed **.rbf** size. The final **.rbf** size will be available after the Quartus II software is able to generate the **.rbf** file.

Enabling compression reduces the amount of configuration data that is transmitted to the Stratix IV device, which also reduces configuration time. On average, compression reduces configuration time, depending on the design.

Programming Serial Configuration Devices

Serial configuration devices are non-volatile, flash-memory-based devices. You can program these devices in-system using the USB-Blaster™, EthernetBlaster™, or ByteBlaster™ II download cable. Alternatively, you can program them using the Altera programming unit (APU), supported third-party programmers, or a microprocessor with the SRunner software driver.

You can perform in-system programming of serial configuration devices using the conventional AS programming interface or JTAG interface solution.

Because serial configuration devices do not support the JTAG interface, the conventional method to program them is using the AS programming interface. The configuration data used to program serial configuration devices is downloaded using programming hardware.

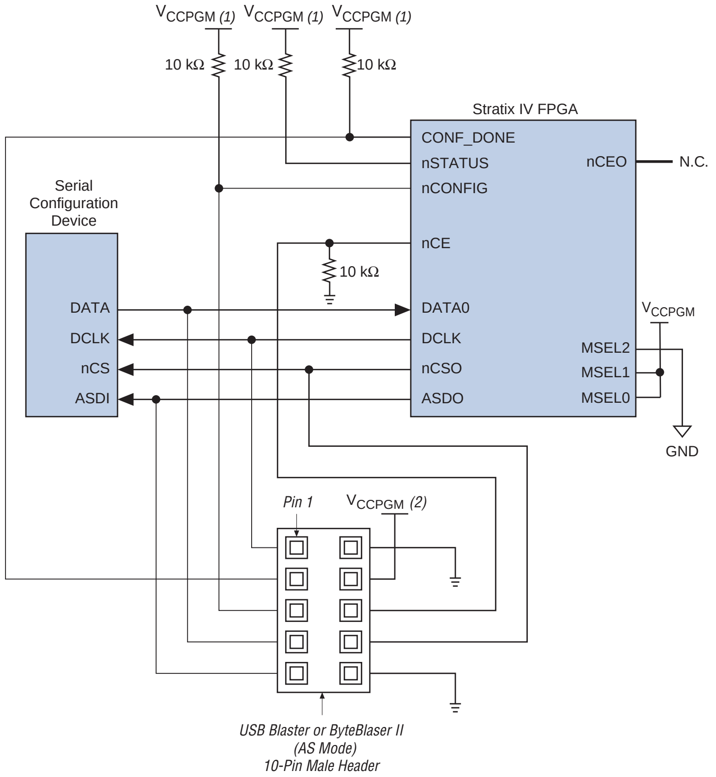
During in-system programming, the download cable disables device access to the AS interface by driving the `nCE` pin high. Stratix IV devices are also held in reset by a low level on `nCONFIG`. After programming is complete, the download cable releases `nCE` and `nCONFIG`, allowing the pull-down and pull-up resistors to drive GND and V_{CCPGM}, respectively. Figure 10–9 shows the download cable connections for the serial configuration device.

Altera has developed Serial FlashLoader (SFL); an in-system programming solution for serial configuration devices using the JTAG interface. This solution requires the Stratix IV device to be a bridge between the JTAG interface and the serial configuration device.

For more information about SFL, refer to *AN 370: Using the Serial FlashLoader with Quartus II Software*.

For more information about the USB Blaster download cable, refer to the *USB-Blaster Download Cable User Guide*. For more information about the ByteBlaster II cable, refer to the *ByteBlaster II Download Cable User Guide*. For more information about the EthernetBlaster download cable, refer to the *EthernetBlaster Communications Cable User Guide*.

Figure 10–9. In-System Programming of Serial Configuration Devices



Notes to Figure 10–9:

(1) Connect these pull-up resistors to V_{CCPGM} at a 3.0-V supply.

(2) Power up the USB-ByteBlaster, ByteBlaster II, or EthernetBlaster cable's $V_{CC(TRGT)}$ with V_{CCPGM}.

You can program serial configuration devices with the Quartus II software using the Altera programming hardware and the appropriate configuration device programming adapter.

In production environments, you can program serial configuration devices using multiple methods. You can use Altera programming hardware or other third-party programming hardware to program blank serial configuration devices before they are mounted on PCBs. Alternatively, you can use an on-board microprocessor to program the serial configuration device in-system using C-based software drivers provided by Altera.

You can program a serial configuration device in-system by an external microprocessor using SRunner. SRunner is a software driver developed for embedded serial configuration device programming, which can be easily customized to fit in different embedded systems. SRunner is able to read a raw programming data (**.rpd**) file and write to serial configuration devices. The serial configuration device programming time using SRunner is comparable to the programming time with the Quartus II software.

For more information about SRunner, refer to *AN 418: SRunner: An Embedded Solution for EPCS Programming* and the source code on the Altera website at www.altera.com.

For more information about programming serial configuration devices, refer to the *Serial Configuration Devices (EPCS1, EPCS4, EPCS16, EPCS64, and EPCS128) Data Sheet* chapter in volume 2 of the *Configuration Handbook*.

Passive Serial Configuration

You can program PS configuration of Stratix IV devices using an intelligent host, such as a MAX II device or microprocessor with flash memory, or a download cable. In the PS scheme, an external host (a MAX II device, embedded processor, or host PC) controls configuration. Configuration data is clocked into the target Stratix IV device using the `DATA0` pin at each rising edge of `DCLK`.

☞ The Stratix IV decompression and design security features are fully available when configuring your Stratix IV device using PS mode.

Table 10–8 shows the `MSEL` pin settings when using the PS configuration scheme.

Table 10–8. Stratix IV MSEL Pin Settings for PS Configuration Schemes

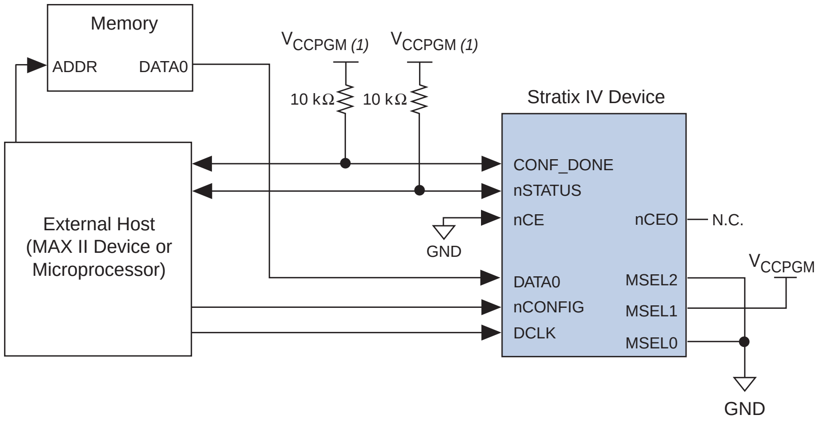
Configuration Scheme	MSEL2	MSEL1	MSEL0
Passive Serial (PS)	0	1	0

PS Configuration Using a MAX II Device as an External Host

In this configuration scheme, you can use a MAX II device as an intelligent host that controls the transfer of configuration data from a storage device, such as flash memory, to the target Stratix IV device. You can store configuration data in **.rbf**, **.hex**, or **.ttf** format.

Figure 10–10 shows the configuration interface connections between a Stratix IV device and a MAX II device for single device configuration.

Figure 10–10. Single Device PS Configuration Using an External Host



Note to Figure 10–10:

(1) Connect the resistor to a supply that provides an acceptable input signal for the Stratix IV device. V_{CCPGM} needs to be high enough to meet the V_{IH} specification of the I/O on the device and the external host. Altera recommends you power up all configuration systems' I/Os with V_{CCPGM}.

Upon power-up, Stratix IV devices go through a POR. The POR delay is dependent on the PORSEL pin setting. When PORSEL is driven low, the standard POR time is 100 ms < T_{POR} < 300 ms. When PORSEL is driven high, the fast POR time is 4 ms < T_{POR} < 12 ms. During POR, the device resets, holds nSTATUS low, and tri-states all user I/O pins. Once the device successfully exits POR, all user I/O pins continue to be tri-stated. If nIO_pullup is driven low during power-up and configuration, the user I/O pins and dual-purpose I/O pins will have weak pull-up resistors which are on (after POR) before and during configuration. If nIO_pullup is driven high, the weak pull-up resistors are disabled.

The configuration cycle consists of three stages: reset, configuration, and initialization. While nCONFIG or nSTATUS are low, the device is in reset. To initiate configuration, the MAX II device must generate a low-to-high transition on the nCONFIG pin.

☞ V_{CC}, V_{CCIO}, V_{CCPGM}, and V_{CCPD} of the banks where the configuration and JTAG pins reside need to be fully powered to the appropriate voltage levels to begin the configuration process.

When nCONFIG goes high, the device comes out of reset and releases the open-drain nSTATUS pin, which is then pulled high by an external 10-kΩ pull-up resistor. Once nSTATUS is released, the device is ready to receive configuration data and the configuration stage begins. When nSTATUS is pulled high, the MAX II device places the configuration data one bit at a time on the DATA0 pin. If you are using configuration data in **.rbf**, **.hex**, or **.ttf** format, you must send the LSB of each data byte first. For example, if the **.rbf** contains the byte sequence `02 1B EE 01 FA`, the serial bitstream you should transmit to the device is `0100-0000 1101-1000 0111-0111 1000-0000 0101-1111`.

The Stratix IV device receives configuration data on the DATA0 pin and the clock is received on the DCLK pin. Data is latched into the device on the rising edge of DCLK. Data is continuously clocked into the target device until CONF_DONE goes high. After the device has received all configuration data successfully, it releases the open-drain CONF_DONE pin, which is pulled high by an external 10-kΩ pull-up resistor. A low-to-high transition on CONF_DONE indicates configuration is complete and initialization of the device can begin. The CONF_DONE pin must have an external 10-kΩ pull-up resistor for the device to initialize.

In Stratix IV devices, the initialization clock source is either the internal oscillator or the optional CLKUSR pin. By default, the internal oscillator is the clock source for initialization. If you use the internal oscillator, the Stratix IV device provides itself with enough clock cycles for proper initialization. Therefore, if the internal oscillator is the initialization clock source, sending the entire configuration file to the device is sufficient to configure and initialize the device. Driving DCLK to the device after configuration is complete does not affect device operation.

You also have the flexibility to synchronize initialization of multiple devices or to delay initialization with the CLKUSR option. You can turn on the **Enable user-supplied start-up clock** (**CLKUSR**) option in the Quartus II software from the **General** tab of the **Device and Pin Options** dialog box. If you supply a clock on CLKUSR, it will not affect the configuration process. After all configuration data has been accepted and CONF_DONE goes high, CLKUSR is enabled after the time specified as t_{CD2CU}. After this time period elapses, Stratix IV devices require 8,532 clock cycles to initialize properly and enter user mode. Stratix IV devices support a CLKUSR f_{MAX} of 125 MHz.

An optional INIT_DONE pin is available, which signals the end of initialization and the start of user-mode with a low-to-high transition. The **Enable INIT_DONE Output** option is available in the Quartus II software from the **General** tab of the **Device and Pin Options** dialog box. If you use the INIT_DONE pin, it is high due to an external 10-kΩ pull-up resistor when nCONFIG is low and during the beginning of configuration. Once the option bit to enable INIT_DONE is programmed into the device (during the first frame of configuration data), the INIT_DONE pin goes low. When initialization is complete, the INIT_DONE pin is released and pulled high. The MAX II device must be able to detect this low-to-high transition which signals the device has entered user mode. When initialization is complete, the device enters user mode. In user-mode, the user I/O pins will no longer have weak pull-up resistors and will function as assigned in your design.

To ensure DCLK and DATA0 are not left floating at the end of configuration, the MAX II device must drive them either high or low, whichever is convenient on your board. The DATA[0] pin is available as a user I/O pin after configuration. When you chose the PS scheme as a default in the Quartus II software, this I/O pin is tri-stated in user mode and should be driven by the MAX II device. To change this default option in the Quartus II software, select the **Dual-Purpose Pins** tab of the **Device and Pin Options** dialog box.

The configuration clock (DCLK) speed must be below the specified frequency to ensure correct configuration. No maximum DCLK period exists, which means you can pause configuration by halting DCLK for an indefinite amount of time.

If an error occurs during configuration, the device drives its nSTATUS pin low, resetting itself internally. The low signal on the nSTATUS pin also alerts the MAX II device that there is an error. If the **Auto-restart configuration after error** option (available in the Quartus II software from the **General** tab of the **Device and Pin Options** dialog box) is turned on, the Stratix IV device releases nSTATUS after a reset time-out period (maximum of 500 µs). After nSTATUS is released and pulled high by a pull-up resistor, the MAX II device can try to reconfigure the target device without needing to pulse nCONFIG low. If this option is turned off, the MAX II device must generate a low-to-high transition (with a low pulse of at least 2 µs) on nCONFIG to restart the configuration process.

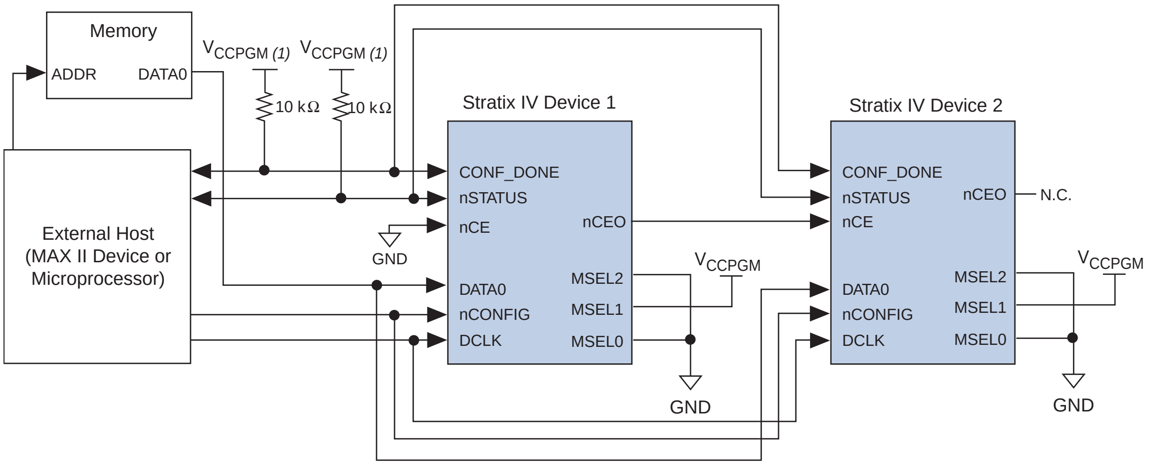
The MAX II device can also monitor the CONF_DONE and INIT_DONE pins to ensure successful configuration. The CONF_DONE pin must be monitored by the MAX II device to detect errors and determine when programming completes. If all configuration data is sent, but CONF_DONE or INIT_DONE have not gone high, the MAX II device must reconfigure the target device.

☞ If you use the optional CLKUSR pin and nCONFIG is pulled low to restart configuration during device initialization, you need to ensure that CLKUSR continues toggling during the time nSTATUS is low (maximum of 500 µs).

When the device is in user-mode, you can initiate a reconfiguration by transitioning the nCONFIG pin low-to-high. The nCONFIG pin must be low for at least 2 µs. When nCONFIG is pulled low, the device also pulls nSTATUS and CONF_DONE low and all I/O pins are tri-stated. Once nCONFIG returns to a logic high level and nSTATUS is released by the device, reconfiguration begins.

Figure 10–11 shows how to configure multiple devices using a MAX II device. This circuit is similar to the PS configuration circuit for a single device, except Stratix IV devices are cascaded for multi-device configuration.

Figure 10–11. Multi-Device PS Configuration Using an External Host



Note to Figure 10–11:

(1) Connect the resistor to a supply that provides an acceptable input signal for all Stratix IV devices in the chain. V_{CCPGM} needs to be high enough to meet the V_{IH} specification of the I/O on the device and the external host. Altera recommends you power up all configuration systems' I/Os with V_{CCPGM}.

In multi-device PS configuration, the first device's nCE pin is connected to GND, while its nCEO pin is connected to nCE of the next device in the chain. The last device's nCE input comes from the previous device, while its nCEO pin is left floating. After the first device completes configuration in a multi-device configuration chain, its nCEO pin drives low to activate the second device's nCE pin, which prompts the second device to begin configuration. The second device in the chain begins configuration within one clock cycle. Therefore, the transfer of data destinations is transparent to the MAX II device. All other configuration pins (nCONFIG, nSTATUS, DCLK, DATA0, and CONF_DONE) are connected to every device in the chain. Configuration signals can require buffering to ensure signal integrity and prevent clock skew problems. Ensure that the DCLK and DATA lines are buffered for every fourth device. Because all device CONF_DONE pins are tied together, all devices initialize and enter user mode at the same time.

Because all nSTATUS and CONF_DONE pins are tied together, if any device detects an error, configuration stops for the entire chain and you must reconfigure the entire chain. For example, if the first device flags an error on nSTATUS, it resets the chain by pulling its nSTATUS pin low. This behavior is similar to a single device detecting an error.

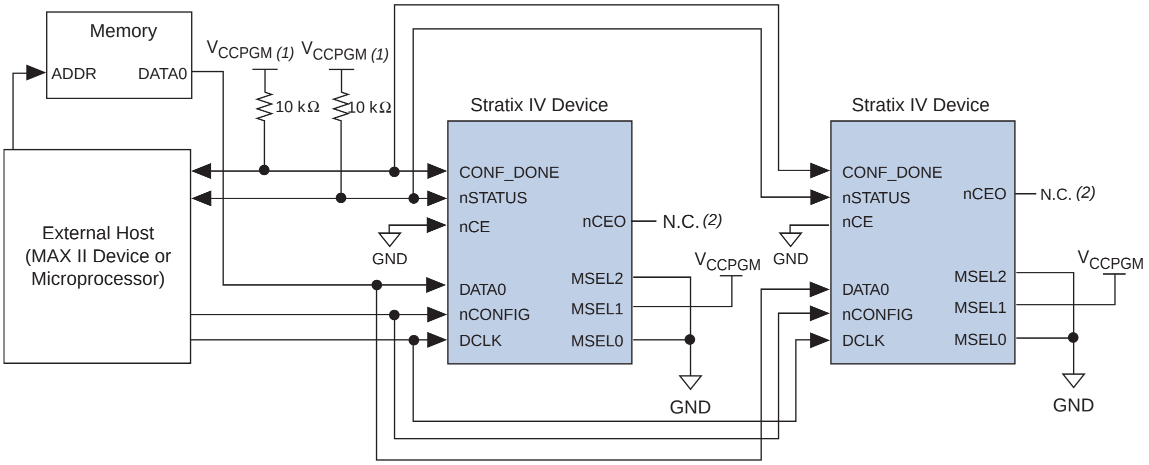
If the **Auto-restart configuration after error** option is turned on, the devices release their nSTATUS pins after a reset time-out period (maximum of 500 µs). After all nSTATUS pins are released and pulled high, the MAX II device can try to reconfigure the chain without needing to pulse nCONFIG low. If this option is turned off, the MAX II device must generate a low-to-high transition (with a low pulse of at least 2 µs) on nCONFIG to restart the configuration process.

In your system, you can have multiple devices that contain the same configuration data. To support this configuration scheme, all device nCE inputs are tied to GND, while nCEO pins are left floating. All other configuration pins (nCONFIG, nSTATUS, DCLK, DATA0, and CONF_DONE) are connected to every device in the chain.

Configuration signals can require buffering to ensure signal integrity and prevent clock skew problems. Ensure that the DCLK and DATA lines are buffered for every fourth device. Devices must be the same density and package. All devices will start and complete configuration at the same time.

Figure 10–12 shows multi-device PS configuration when both Stratix IV devices are receiving the same configuration data.

Figure 10–12. Multiple-Device PS Configuration When Both Devices Receive the Same Data



Notes to Figure 10–12:

(1) Connect the resistor to a supply that provides an acceptable input signal for all Stratix IV devices in the chain. V_{CCPGM} needs to be high enough to meet the V_{IH} specification of the I/O on the device and the external host. Altera recommends you power up all configuration system's I/Os with V_{CCPGM}.

(2) The nCEO pins of both devices are left unconnected when configuring the same configuration data into multiple devices.

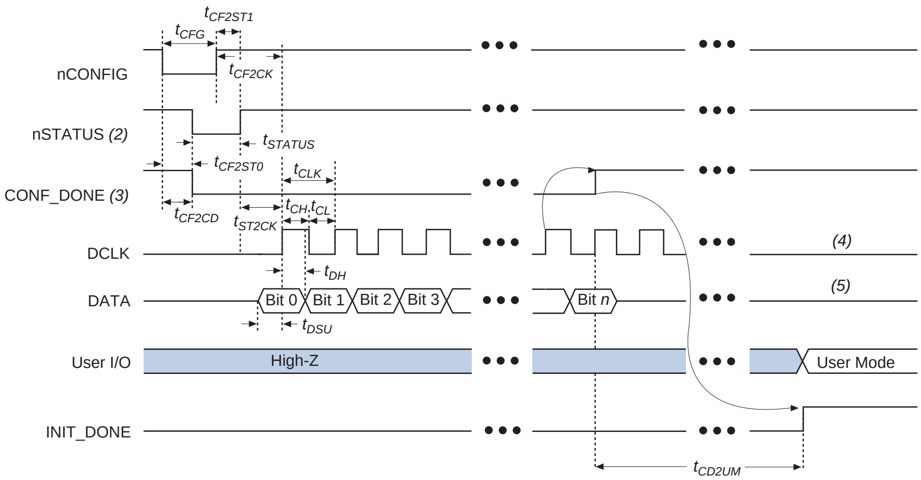
You can use a single configuration chain to configure Stratix IV devices with other Altera devices. To ensure that all devices in the chain complete configuration at the same time, or that an error flagged by one device initiates reconfiguration in all devices, all of the device CONF_DONE and nSTATUS pins must be tied together.

For more information about configuring multiple Altera devices in the same configuration chain, refer to the *Configuring Mixed Altera FPGA Chains* chapter in volume 2 of the *Configuration Handbook*.

PS Configuration Timing

Figure 10–13 shows the timing waveform for PS configuration when using a MAX II device as an external host.

Figure 10–13. PS Configuration Timing Waveform *(Note 1)*



Notes to Figure 10–13:

(1) The beginning of this waveform shows the device in user mode. In user mode, nCONFIG, nSTATUS, and CONF_DONE are at logic high levels. When nCONFIG is pulled low, a reconfiguration cycle begins.

(2) Upon power-up, the Stratix IV device holds nSTATUS low for the time of the POR delay.

(3) Upon power-up, before and during configuration, CONF_DONE is low.

(4) Do not leave DCLK floating after configuration. You can drive it high or low, whichever is more convenient.

(5) DATA[0] is available as a user I/O pin after configuration. The state of this pin depends on the dual-purpose pin settings.

Table 10–9 defines the timing parameters for Stratix IV devices for PS configuration.

Table 10–9. ***PS Timing Parameters for Stratix IV Devices*** ***(Note 1)*** ***(Part 1 of 2)***

Symbol	Parameter	Minimum	Maximum	Units
t_{CF2CD}	nCONFIG low to CONF_DONE low	—	800	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low	—	800	ns
t_{CFG}	nCONFIG low pulse width	2	—	µs
t_{STATUS}	nSTATUS low pulse width	10	500 *(2)*	µs
t_{CF2ST1}	nCONFIG high to nSTATUS high	—	500 *(2)*	µs
t_{CF2CK}	nCONFIG high to first rising edge on DCLK	500	—	µs
t_{ST2CK}	nSTATUS high to first rising edge of DCLK	2	—	µs
t_{DSU}	Data setup time before rising edge on DCLK	4	—	ns
t_{DH}	Data hold time after rising edge on DCLK	0	—	ns
t_{CH}	DCLK high time	3.2	—	ns
t_{CL}	DCLK low time	3.2	—	ns
t_{CLK}	DCLK period	8	—	ns
f_{MAX}	DCLK frequency	—	125	MHz
t_R	Input rise time	—	40	ns

Table 10–9. ***PS Timing Parameters for Stratix IV Devices*** ***(Note 1)*** ***(Part 2 of 2)***

Symbol	Parameter	Minimum	Maximum	Units
t	Input fall time	—	40	ns
t_{CD2UM}	CONF_DONE high to user mode *(3)*	55	150	s
t_{CD2CU}	CONF_DONE high to CLKUSR enabled	4 × maximum DCLK period	—	—
t_{CD2UMC}	CONF_DONE high to user mode with CLKUSR option on	t_{CD2CU} + (8532 CLKUSR period)	—	—

Notes to Table 10–9:

(1) This information is preliminary.

(2) This value is applicable if you do not delay configuration by extending the nCONFIG or nSTATUS low pulse width.

(3) The minimum and maximum numbers apply only if you choose the internal oscillator as the clock source for starting the device.

Device configuration options and how to create configuration files are discussed further in the *Device Configuration Options* and *Configuration File Formats* chapters in volume 2 of the *Configuration Handbook*.

PS Configuration Using a Microprocessor

In this PS configuration scheme, a microprocessor controls the transfer of configuration data from a storage device, such as flash memory, to the target Stratix IV device.

Refer to "PS Configuration Using a MAX II Device as an External Host" on page 10–22 for all configuration and timing information. This section is also applicable when using a microprocessor as an external host.

PS Configuration Using a Download Cable

In this section, the generic term "download cable" includes the Altera USB-Blaster universal serial bus (USB) port download cable, MasterBlaster serial/USB communications cable, ByteBlaster II parallel port download cable, ByteBlasterMV parallel port download cable, and EthernetBlaster download cable.

In a PS configuration with a download cable, an intelligent host (such as a PC) transfers data from a storage device to the device using the USB Blaster, MasterBlaster, ByteBlaster II, EthernetBlaster, or ByteBlasterMV cable.

Upon power-up, the Stratix IV devices go through a POR. The POR delay is dependent on the PORSEL pin setting. When PORSEL is driven low, the standard POR time is 100 ms < T_{POR} < 300 ms. When PORSEL is driven high, the fast POR time is 4 ms < T_{POR} < 12 ms. During POR, the device resets, holds nSTATUS low, and tri-states all user I/O pins. Once the device successfully exits POR, all user I/O pins continue to be tri-stated. If nIO_pullup is driven low during power-up and configuration, the user I/O pins and dual-purpose I/O pins will have weak pull-up resistors, which are on (after POR) before and during configuration. If nIO_pullup is driven high, the weak pull-up resistors are disabled.

The configuration cycle consists of three stages: reset, configuration, and initialization. While `nCONFIG` or `nSTATUS` are low, the device is in reset. To initiate configuration in this scheme, the download cable generates a low-to-high transition on the `nCONFIG` pin.

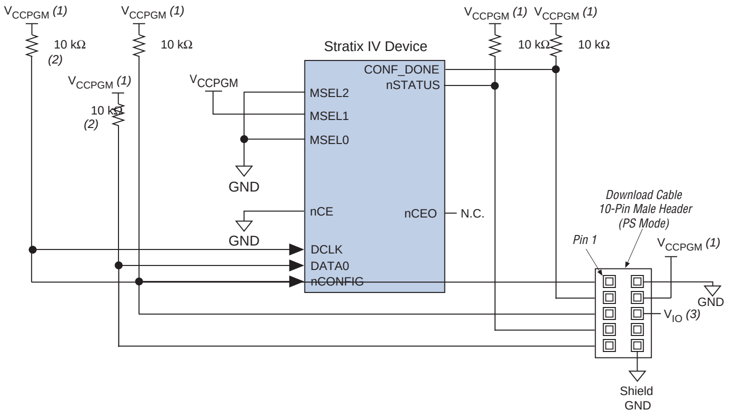
☞ To begin configuration, power the V_{CC}, V_{CCIO}, V_{CCPGM}, and V_{CCPD} voltages (for the banks where the configuration and JTAG pins reside) to the appropriate voltage levels.

When `nCONFIG` goes high, the device comes out of reset and releases the open-drain `nSTATUS` pin, which is then pulled high by an external 10-kΩ pull-up resistor. Once `nSTATUS` is released, the device is ready to receive configuration data and the configuration stage begins. The programming hardware or download cable then places the configuration data one bit at a time on the device's `DATA0` pin. The configuration data is clocked into the target device until `CONF_DONE` goes high. The `CONF_DONE` pin must have an external 10-kΩ pull-up resistor for the device to initialize.

When using a download cable, setting the **Auto-restart configuration after error** option does not affect the configuration cycle because you must manually restart configuration in the Quartus II software when an error occurs. Additionally, the **Enable user-supplied start-up clock** (**CLKUSR**) option has no affect on the device initialization because this option is disabled in the **.sof** when programming the device using the Quartus II programmer and download cable. Therefore, if you turn on the `CLKUSR` option, you do not need to provide a clock on `CLKUSR` when you are configuring the device with the Quartus II programmer and a download cable.

Figure 10–14 shows PS configuration for Stratix IV devices using a USB Blaster, EthernetBlaster, MasterBlaster, ByteBlaster II, or ByteBlasterMV cable.

Figure 10–14. PS Configuration Using a USB Blaster, EthernetBlaster, MasterBlaster, ByteBlaster II, or ByteBlasterMV Cable



Notes to Figure 10–14:

(1) Connect the pull-up resistor to the same supply voltage (V_{CCPGM}) as the USB Blaster, MasterBlaster (`VIO` pin), ByteBlaster II, ByteBlasterMV, or EthernetBlaster cable.

(2) You only need the pull-up resistors on `DATA0` and `DCLK` if the download cable is the only configuration scheme used on your board. This ensures that `DATA0` and `DCLK` are not left floating after configuration. For example, if you are also using a configuration device, you do not need the pull-up resistors on `DATA0` and `DCLK`.

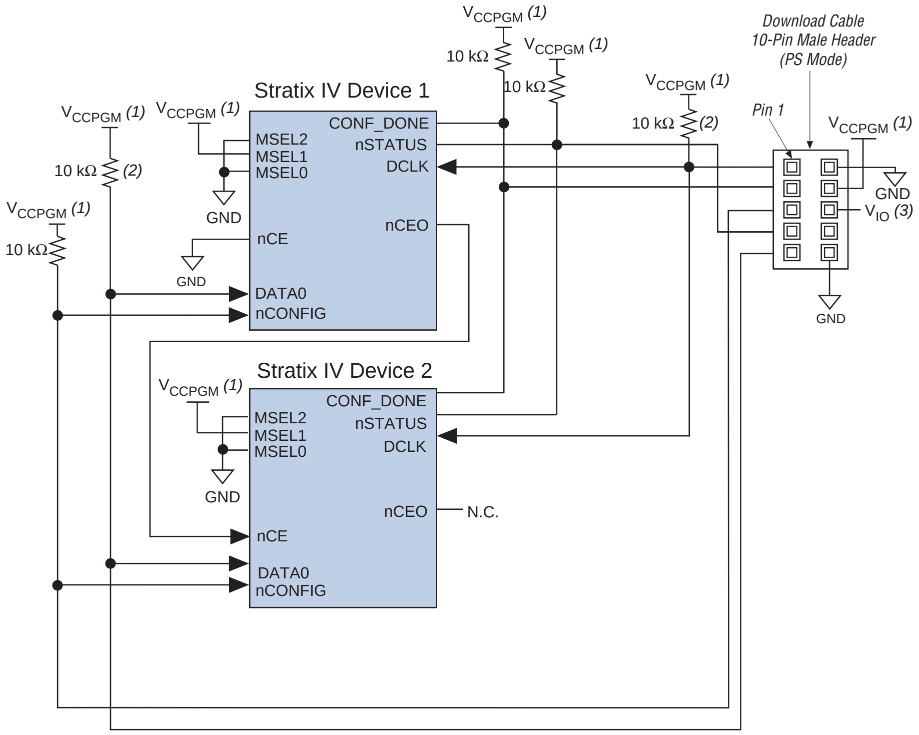
(3) Pin 6 of the header is a V_{IO} reference voltage for the MasterBlaster output driver. V_{IO} needs to match the device's V_{CCPGM}. Refer to the *MasterBlaster Serial/USB Communications Cable User Guide* for this value. In the USB-Blaster, ByteBlaster II, and ByteBlasterMV cable, this pin is a no connect.

You can use a download cable to configure multiple Stratix IV devices by connecting each device's nCEO pin to the subsequent device's nCE pin. The first device's nCE pin is connected to GND while its nCEO pin is connected to the nCE of the next device in the chain. The last device's nCE input comes from the previous device, while its nCEO pin is left floating. All other configuration pins (nCONFIG, nSTATUS, DCLK, DATA0, and CONF_DONE) are connected to every device in the chain. Because all CONF_DONE pins are tied together, all devices in the chain initialize and enter user mode at the same time.

In addition, because the nSTATUS pins are tied together, the entire chain halts configuration if any device detects an error. The **Auto-restart configuration after error** option does not affect the configuration cycle because you must manually restart configuration in the Quartus II software when an error occurs.

Figure 10–15 shows how to configure multiple Stratix IV devices with a download cable.

Figure 10–15. Multi-Device PS Configuration Using a USB Blaster, EthernetBlaster, MasterBlaster, ByteBlaster II, or ByteBlasterMV Cable



Notes to Figure 10–15:

(1) Connect the pull-up resistor to the same supply voltage (V_{CCPGM}) as the USB Blaster, MasterBlaster (VIO pin), ByteBlaster II, ByteBlasterMV, or EthernetBlaster cable.

(2) You only need the pull-up resistors on DATA0 and DCLK if the download cable is the only configuration scheme used on your board. This is to ensure that DATA0 and DCLK are not left floating after configuration. For example, if you are also using a configuration device, you do not need the pull-up resistors on DATA0 and DCLK.

(3) Pin 6 of the header is a V_{IO} reference voltage for the MasterBlaster output driver. V_{IO} needs to match the device's V_{CCPGM}. Refer to the *MasterBlaster Serial/USB Communications Cable User Guide* for this value. In the ByteBlasterMV cable, this pin is a no connect. In the USB-Blaster, ByteBlaster II, and ByteBlasterMV cables, this pin is a no connect.

For more information about how to use the USB Blaster, MasterBlaster, ByteBlaster II, or ByteBlasterMV cables, refer to the following user guides:

- *USB Blaster USB Download Cable User Guide*
- *MasterBlaster Serial/USB Communications Cable User Guide*
- *ByteBlaster II Download Cable User Guide*
- *ByteBlasterMV Download Cable User Guide*
- *EthernetBlaster Communications Cable User Guide*

JTAG Configuration

JTAG has developed a specification for boundary-scan testing. This boundary-scan test (BST) architecture offers the capability to efficiently test components on PCBs with tight lead spacing. The BST architecture can test pin connections without using physical test probes and capture functional data while a device is operating normally. You can also use JTAG circuitry to shift configuration data into the device. The Quartus II software automatically generates **.sof** files that you can use for JTAG configuration with a download cable in the Quartus II software programmer.

For more information about JTAG boundary-scan testing and commands available using Stratix IV devices, refer to the following documents:

- *JTAG Boundary Scan Testing* chapter in volume 1 of the *Stratix IV Device Handbook*
- *Programming Support for Jam STAPL Language*

Stratix IV devices are designed such that JTAG instructions have precedence over any device configuration modes. Therefore, JTAG configuration can take place without waiting for other configuration modes to complete. For example, if you attempt JTAG configuration of Stratix IV devices during PS configuration, PS configuration is terminated and JTAG configuration begins.

You cannot use the Stratix IV decompression or design security features if you are configuring your Stratix IV device when using JTAG-based configuration.

A device operating in JTAG mode uses four required pins, `TDI`, `TDO`, `TMS`, and `TCK`, and one optional pin, `TRST`. The `TCK` pin has an internal weak pull-down resistor, while the `TDI`, `TMS`, and `TRST` pins have weak internal pull-up resistors (typically 25 kΩ). The JTAG output pin `TDO` and all JTAG input pins are powered by 2.5-V/3.0-V V_{CCPD}. All the JTAG pins support only LVTTL I/O standard.

All user I/O pins are tri-stated during JTAG configuration. Table 10–10 explains each JTAG pin's function.

The `TDO` output is powered by the V_{CCPD} power supply of I/O bank 1A. For recommendations about how to connect a JTAG chain with multiple voltages across the devices in the chain, refer to the *JTAG Boundary Scan Testing* chapter in volume 1 of the *Stratix IV Device Handbook*.

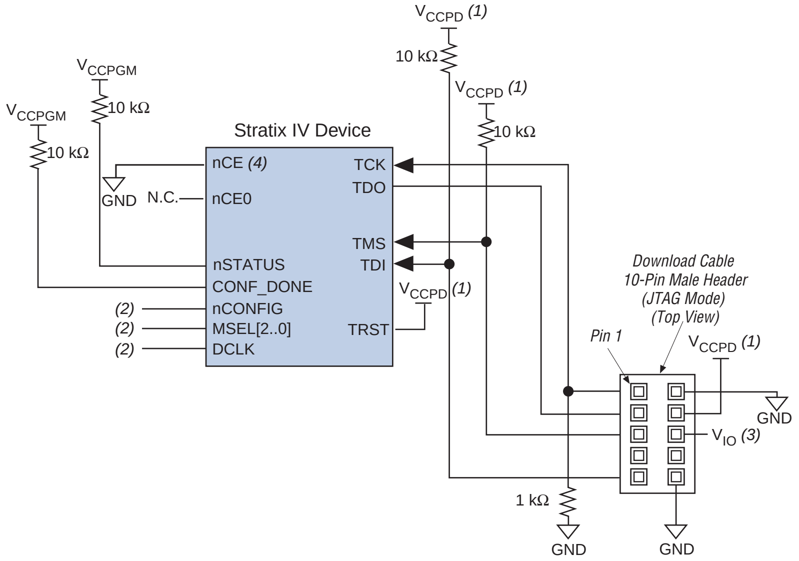
Table 10–10. Dedicated JTAG Pins

Pin Name	Pin Type	Description
`TDI`	Test data input	Serial input pin for instructions as well as test and programming data. Data is shifted in the rising edge of `TCK`. If the JTAG interface is not required on the board, you can disable the JTAG circuitry by connecting this pin to logic high.
`TDO`	Test data output	Serial data output pin for instructions as well as test and programming data. Data is shifted out on the falling edge of `TCK`. The pin is tri-stated if data is not being shifted out of the device. If the JTAG interface is not required on the board, you can disable the JTAG circuitry by leaving this pin unconnected.
`TMS`	Test mode select	Input pin that provides the control signal to determine the transitions of the TAP controller state machine. `TMS` is evaluated on the rising edge of `TCK`. Therefore, you must set up `TMS` before the rising edge of `TCK`. Transitions within the state machine occur on the falling edge of `TCK` after the signal is applied to `TMS`. If the JTAG interface is not required on the board, you can disable the JTAG circuitry by connecting this pin to logic high.
`TCK`	Test clock input	The clock input to the BST circuitry. Some operations occur at the rising edge while others occur at the falling edge. If the JTAG interface is not required on the board, you can disable the JTAG circuitry by connecting this pin to GND.
`TRST`	Test reset input (optional)	Active-low input to asynchronously reset the boundary-scan circuit. The `TRST` pin is optional according to IEEE Std. 1149.1. If the JTAG interface is not required on the board, you can disable the JTAG circuitry by connecting this pin to GND.

During JTAG configuration, you can download data to the device on the PCB through the USB Blaster, MasterBlaster, ByteBlaster II, EthernetBlaster, or ByteBlasterMV download cable. Configuring devices through a cable is similar to programming devices in-system, except you must connect the `TRST` pin to V_{CCPD}. This ensures that the TAP controller is not reset.

Figure 10–16 shows JTAG configuration of a single Stratix IV device.

Figure 10–16. JTAG Configuration of a Single Device Using a Download Cable



Notes to Figure 10–16:

(1) Connect the pull-up resistor to the same supply voltage as the USB Blaster, MasterBlaster (`VIO` pin), ByteBlaster II, ByteBlasterMV, or EthernetBlaster cable. The voltage supply can be connected to the V_{CCPD} of the device.

(2) Connect the `nCONFIG` and `MSEL[2..0]` pins to support a non-JTAG configuration scheme. If you only use the JTAG configuration, connect `nCONFIG` to V_{CCPGM}, and `MSEL[2..0]` to GND. Pull `DCLK` either high or low, whichever is convenient on your board.

(3) Pin 6 of the header is a V_{IO} reference voltage for the MasterBlaster output driver. V_{IO} needs to match the device's V_{CCPD}. Refer to the *MasterBlaster Serial/USB Communications Cable User Guide* for this value. In the USB-Blaster, ByteBlaster II, and ByteBlasterMV cable, this pin is a no connect.

(4) You must connect `nCE` to GND or driven low for successful JTAG configuration.

To configure a single device in a JTAG chain, the programming software places all other devices in bypass mode. In bypass mode, devices pass programming data from the `TDI` pin to the `TDO` pin through a single bypass register without being affected internally. This scheme enables the programming software to program or verify the target device. Configuration data driven into the device appears on the `TDO` pin one clock cycle later.

The Quartus II software verifies successful JTAG configuration upon completion. At the end of configuration, the software checks the state of `CONF_DONE` through the JTAG port. When the Quartus II software generates a JAM file (**.jam)** for a multi-device chain, it contains instructions so that all the devices in the chain are initialized at the same time. If `CONF_DONE` is not high, the Quartus II software indicates that configuration has failed. If `CONF_DONE` is high, the software indicates that configuration was successful. After the configuration bitstream is transmitted serially using the JTAG `TDI` port, the `TCK` port is clocked an additional 1,094 cycles to perform device initialization.

Stratix IV devices have dedicated JTAG pins that always function as JTAG pins. Not only can you perform JTAG testing on Stratix IV devices before and after, but also during configuration. While other device families do not support JTAG testing during configuration, Stratix IV devices support the bypass, ID code, and sample instructions during configuration without interrupting configuration. All other JTAG instructions may only be issued by first interrupting configuration and reprogramming I/O pins using the CONFIG_IO instruction.

The CONFIG_IO instruction allows I/O buffers to be configured using the JTAG port and when issued, interrupts configuration. This instruction allows you to perform board-level testing prior to configuring the Stratix IV device or waiting for a configuration device to complete configuration. Once configuration has been interrupted and JTAG testing is complete, you must reconfigure the part using JTAG (PULSE_CONFIG instruction) or by pulsing nCONFIG low.

The chip-wide reset (DEV_CLRn) and chip-wide output enable (DEV_OE) pins on Stratix IV devices do not affect JTAG boundary-scan or programming operations. Toggling these pins does not affect JTAG operations (other than the usual boundary-scan operation).

When designing a board for JTAG configuration of Stratix IV devices, consider the dedicated configuration pins. Table 10–11 shows how these pins are connected during JTAG configuration.

Table 10–11. Dedicated Configuration Pin Connections During JTAG Configuration

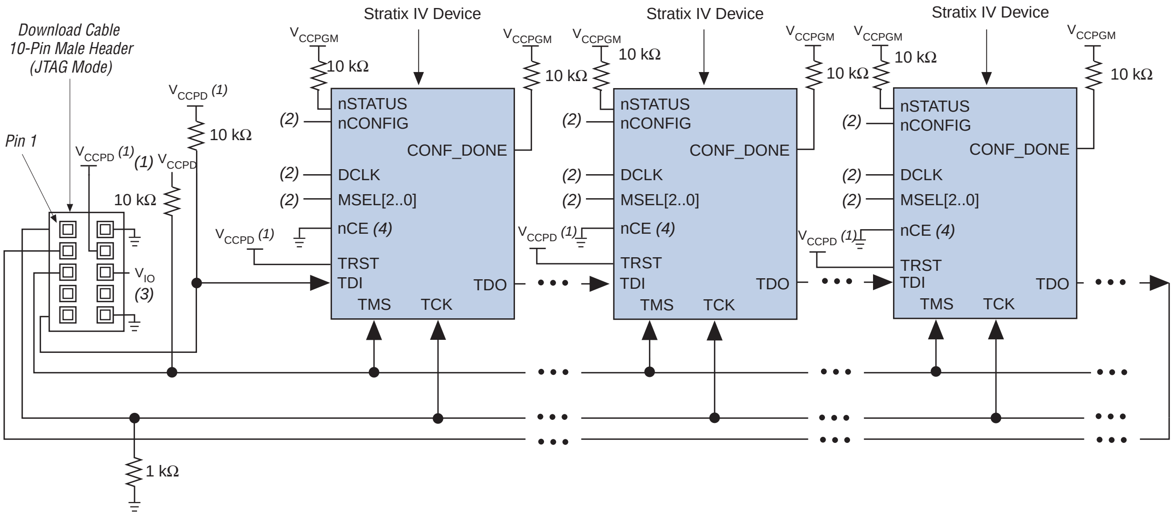
Signal	Description
nCE	On all Stratix IV devices in the chain, nCE should be driven low by connecting it to ground, pulling it low using a resistor, or driving it by some control circuitry. For devices that are also in multi-device FPP, AS, or PS configuration chains, the nCE pins must be connected to GND during JTAG configuration or JTAG must be configured in the same order as the configuration chain.
nCEO	On all Stratix IV devices in the chain, you can leave nCEO floating or connected to the nCE of the next device.
MSEL	These pins must not be left floating. These pins support whichever non-JTAG configuration is used in production. If you only use JTAG configuration, tie these pins to GND.
nCONFIG	Driven high by connecting to V_{CCPGM}, pulling up using a resistor, or driven high by some control circuitry.
nSTATUS	Pull to V_{CCPGM} using a 10-kΩ resistor. When configuring multiple devices in the same JTAG chain, each nSTATUS pin must be pulled up to V_{CCPGM} individually.
CONF_DONE	Pull to V_{CCPGM} using a 10-kΩ resistor. When configuring multiple devices in the same JTAG chain, each CONF_DONE pin must be pulled up to V_{CCPGM} individually. CONF_DONE going high at the end of JTAG configuration indicates successful configuration.
DCLK	Do not leave DCLK floating. Drive low or high, whichever is more convenient on your board.

When programming a JTAG device chain, one JTAG-compatible header is connected to several devices. The number of devices in the JTAG chain is limited only by the drive capability of the download cable. When four or more devices are connected in a JTAG chain, Altera recommends buffering the TCK, TDI, and TMS pins with an on-board buffer.

JTAG-chain device programming is ideal when the system contains multiple devices, or when testing your system using JTAG BST circuitry.

Figure 10–17 shows a multi-device JTAG configuration.

Figure 10–17. JTAG Configuration of Multiple Devices Using a Download Cable



Notes to Figure 10–17:

(1) Connect the pull-up resistor to the same supply voltage as the USB Blaster, MasterBlaster (V_{IO} pin), ByteBlaster II, ByteBlasterMV, or EthernetBlaster cable. Connect the voltage supply to V_{CCPD} of the device.

(2) Connect the nCONFIG, MSEL[2..0] pins to support a non-JTAG configuration scheme. If you only use JTAG configuration, connect nCONFIG to V_{CCPGM}, and MSEL[2..0] to GND. Pull DCLK either high or low, whichever is convenient on your board.

(3) Pin 6 of the header is a V_{IO} reference voltage for the MasterBlaster output driver. V_{IO} should match the device's V_{CCPD}. Refer to the *MasterBlaster Serial/USB Communications Cable User Guide* for this value. In the ByteBlasterMV cable, this pin is a no connect. In the USB-Blaster, ByteBlaster II, and ByteBlasterMV cables, this pin is a no connect.

(4) You must connect nCE to GND or drive it low for successful JTAG configuration.

You must connect the nCE pin to GND or drive it low during JTAG configuration. In multi-device FPP, AS, and PS configuration chains, the first device's nCE pin is connected to GND while its nCEO pin is connected to nCE of the next device in the chain. The last device's nCE input comes from the previous device, while its nCEO pin is left floating. In addition, the CONF_DONE and nSTATUS signals are all shared in multi-device FPP, AS, or PS configuration chains so the devices can enter user mode at the same time after configuration is complete. When the CONF_DONE and nSTATUS signals are shared among all the devices, you must configure every device when JTAG configuration is performed.

If you only use JTAG configuration, Altera recommends that you connect the circuitry as shown in Figure 10–17, where each of the CONF_DONE and nSTATUS signals are isolated, so that each device can enter user mode individually.

After the first device completes configuration in a multi-device configuration chain, its nCEO pin drives low to activate the second device's nCE pin, which prompts the second device to begin configuration. Therefore, if these devices are also in a JTAG chain, make sure the nCE pins are connected to GND during JTAG configuration or that the devices are JTAG configured in the same order as the configuration chain. As long as the devices are JTAG configured in the same order as the multi-device configuration chain, the nCEO of the previous device drives the nCE of the next device low when it has successfully been JTAG configured.

You can place other Altera devices that have JTAG support in the same JTAG chain for device programming and configuration.

JTAG configuration support is enhanced and allows more than 17 Stratix IV devices to be cascaded in a JTAG chain.

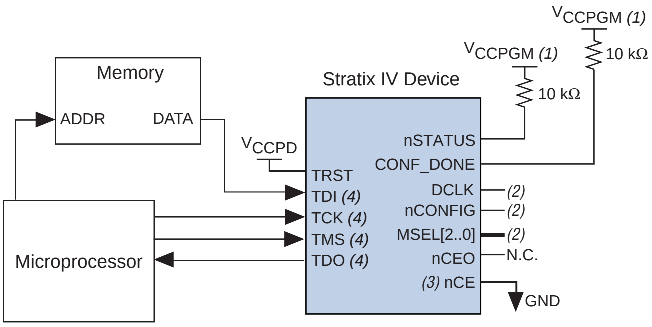
For more information about configuring multiple Altera devices in the same configuration chain, refer to the *Configuring Mixed Altera Device Chains* chapter in volume 2 of the *Configuration Handbook*.

You can configure Stratix IV devices using multiple configuration schemes on the same board. Combining JTAG configuration with passive serial (PS) or active serial (AS) configuration on your board is useful in the prototyping environment because it allows multiple methods to configure your FPGA.

For more information about combining JTAG configuration with other configuration schemes, refer to the *Combining Different Configuration Schemes* chapter in the *Configuration Handbook*.

Figure 10–18 shows JTAG configuration of a Single Device Using a Microprocessor.

Figure 10–18. JTAG Configuration of a Single Device Using a Microprocessor



Notes to Figure 10–18:

(1) Connect the pull-up resistor to a supply that provides an acceptable input signal for all Stratix IV devices in the chain. V_{CCPGM} needs to be high enough to meet the V_{IH} specification of the I/O on the device.

(2) Connect the nCONFIG and MSEL[2..0] pins to support a non-JTAG configuration scheme. If you use only the JTAG configuration, connect nCONFIG to V_{CCPGM} and MSEL[2..0] to GND. Pull DCLK either high or low, whichever is convenient on your board.

(3) Connect nCE to GND or drive it low for successful JTAG configuration.

(4) The microprocessor needs to use the same I/O standard as V_{CCPD} to drive the JTAG pins.

Jam STAPL

Jam STAPL, JEDEC standard JESD-71, is a standard file format for in-system programmability (ISP) purposes. Jam STAPL supports programming or configuration of programmable devices and testing of electronic systems, using the IEEE 1149.1 JTAG interface. Jam STAPL is a freely licensed open standard.

The Jam Player provides an interface for manipulating the IEEE Std. 1149.1 JTAG TAP state machine.

For more information about JTAG and Jam STAPL in embedded environments, refer to *AN 122: Using Jam STAPL for ISP and ICR via an Embedded Processor*. To download the jam player, visit the Altera website at www.altera.com.

Device Configuration Pins

The following tables describe the connections and functionality of all the configuration-related pins on the Stratix IV devices. Table 10–12 summarizes the Stratix IV configuration pins and their power supply.

Table 10–12. Stratix IV Configuration Pin Summary *(Note 1)* (Part 1 of 2)

Description	Input/Output	Dedicated	Powered By	Configuration Mode
TDI	Input	Yes	V_{CCPD}	JTAG
TMS	Input	Yes	V_{CCPD}	JTAG
TCK	Input	Yes	V_{CCPD}	JTAG
TRST	Input	Yes	V_{CCPD}	JTAG
TDO	Output	Yes	V_{CCPD}	JTAG
CRC_ERROR	Output	—	Pull-up	Optional, all modes
DATA0	Input	—	V_{CCPGM}/V_{CCIO}	All modes except JTAG
DATA[7..1]	Input	—	V_{CCPGM}/V_{CCIO}	FPP
INIT_DONE	Output	—	Pull-up	Optional, all modes
CLKUSR	Input	—	V_{CCPGM}/V_{CCIO}	Optional
nSTATUS	Bidirectional	Yes	V_{CCPGM}/Pull-up	All modes
nCE	Input	Yes	V_{CCPGM}	All modes
CONF_DONE	Bidirectional	Yes	V_{CCPGM}/Pull-up	All modes
nCONFIG	Input	Yes	V_{CCPGM}	All modes
PORSEL	Input	Yes	V_{CC} *(2)*	All modes
ASDO	Output	Yes	V_{CCPGM}	AS
nCSO	Output	Yes	V_{CCPGM}	AS
DCLK	Input	Yes	V_{CCPGM}	PS, FPP
	Output	Yes	V_{CCPGM}	AS
nIO_PULLUP	Input	Yes	V_{CC} *(2)*	All modes
nCEO	Output	Yes	V_{CCPGM}	All modes

Table 10–12. Stratix IV Configuration Pin Summary *(Note 1)* (Part 2 of 2)

Description	Input/Output	Dedicated	Powered By	Configuration Mode
MSEL[2..0]	Input	Yes	V_{CC} *(2)*	All modes

Notes to Table 10–12:

(1) The total number of pins is 29. The total number of dedicated pins is 18.

(2) Although MSEL[2..0] PORSEL and nIO_PULLUP are powered up by V_{CC}, Altera recommends you connect these pins to V_{CCPGM} or GND directly without using a pull-up or pull-down resistor.

Table 10–13 describes the dedicated configuration pins. You must connect these pins properly on your board for successful configuration. Some of these pins may not be required for your configuration schemes.

Table 10–13. *Dedicated Configuration Pins on the Stratix IV Device (Part 1 of 4)*

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
VCCPGM	N/A	All	Power	Dedicated power pin. Use this pin to power all dedicated configuration inputs, dedicated configuration outputs, dedicated configuration bidirectional pins, and some of the dual functional pins that are used for configuration. You must connect this pin to 1.8 V, 2.5 V, or 3.0 V. V_{CCPGM} must ramp-up from 0-V to V_{CCPGM} within 100 ms when PORSEL is low or 4 ms when PORSEL is high. If V_{CCPGM} is not ramped up within this specified time, your Stratix IV device will not configure successfully. If your system does not allow for a V_{CCPGM} ramp-up within 100 ms or 4 ms, you must hold nCONFIG low until all power supplies are stable.
VCCPD	N/A	All	Power	Dedicated power pin. Use this pin to power the I/O pre-drivers, JTAG input and output pins, and design security circuitry. You must connect this pin to 2.5 V or 3.0 V, depending on the I/O standards selected. V_{CCPD}= 3.0 V. For 2.5 V or below I/O standards, V_{CCPD} = 2.5 V. V_{CCPD} must ramp-up from 0 V to 2.5 V / 3.0 V within 100 ms when PORSEL is low or 4 ms when PORSEL is high. If V_{CCPD} is not ramped up within this specified time, your Stratix IV device will not configure successfully. If your system does not allow for a V_{CCPD} to ramp-up time within 100 ms or 4 ms, you must hold nCONFIG low until all power supplies are stable.
PORSEL	N/A	All	Input	Dedicated input which selects between a standard POR time or a fast POR time. A logic low selects a standard POR time setting of 100 ms < T_{POR} < 300 ms and a logic high selects a fast POR time setting of 4 ms < T_{POR} < 12 ms. The PORSEL input buffer is powered by V_{CC} and has an internal 5-kΩ pull-down resistor that is always active. Tie the PORSEL pin directly to V_{CCPGM} or GND.

Table 10–13. ***Dedicated Configuration Pins on the Stratix IV Device (Part 2 of 4)***

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
nIO_PULLUP	N/A	All	Input	Dedicated input that chooses whether the internal pull-up resistors on the user I/O pins and dual-purpose I/O pins (nCSO, nASDO, DATA[7..0], CLKUSR, and INIT_DONE) are on or off before and during configuration. A logic high turns off the weak internal pull-up resistors; while a logic low turns them on. The nIO-PULLUP input buffer is powered by V_{CC} and has an internal 5-kΩ pull-down resistor that is always active. Tie the nIO-PULLUP directly to V_{CCPGM} or GND.
MSEL[2..0]	N/A	All	Input	Three-bit configuration input that sets the Stratix IV device configuration scheme. Refer to Table 10–1 on page 10–2 for the appropriate connections. You must hardwire these pins to V_{CCPGM} or GND. The MSEL[2..0] pins have internal 5-kΩ pull-down resistors that are always active.
nCONFIG	N/A	All	Input	Configuration control input. Pulling this pin low during user-mode causes the device to lose its configuration data, enter a reset state, and tri-state all I/O pins. Returning this pin to a logic high level initiates a reconfiguration. Configuration is possible only if this pin is high, except in JTAG programming mode, when nCONFIG is ignored.
nSTATUS	N/A	All	Bidirectional open-drain	The device drives nSTATUS low immediately after power-up and releases it after the POR time. During user mode and regular configuration, this pin is pulled high by an external 10-kΩ resistor. This pin, when driven low by Stratix IV, indicates that the device has encountered an error during configuration. Status output — If an error occurs during configuration, nSTATUS is pulled low by the target device. Status input — If an external source drives the nSTATUS pin low during configuration or initialization, the target device enters an error state. Driving nSTATUS low after configuration and initialization does not affect the configured device. If you use a configuration device, driving nSTATUS low causes the configuration device to attempt to configure the device, but because the device ignores transitions on nSTATUS in user mode, the device does not reconfigure. To initiate a reconfiguration, nCONFIG must be pulled low.

Table 10–13. ***Dedicated Configuration Pins on the Stratix IV Device (Part 3 of 4)***

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
nSTATUS (continued)	—	—	—	If V_{CCPGM} is not fully powered up, the following could occur: ■ V_{CCPGM} is powered high enough for the nSTATUS buffer to function properly, and nSTATUS is driven low. When V_{CCPGM} is ramped up, POR trips and nSTATUS is released after POR expires. ■ V_{CCPGM} is not powered high enough for the nSTATUS buffer to function properly. In this situation, nSTATUS might appear logic high, triggering a configuration attempt that would fail because POR did not yet trip. When V_{CCPD} is powered up, nSTATUS is pulled low because POR did not yet trip. When POR trips after V_{CCPGM} is powered up, nSTATUS is released and pulled high. At that point, reconfiguration is triggered and the device is configured.
CONF_DONE	N/A	All	Bidirectional open-drain	Status output. The target device drives the CONF_DONE pin low before and during configuration. Once all configuration data is received without error and the initialization cycle starts, the target device releases CONF_DONE. Status input. After all data is received and CONF_DONE goes high, the target device initializes and enters user mode. The CONF_DONE pin must have an external 10-kΩpull-up resistor for the device to initialize. Driving CONF_DONE low after configuration and initialization does not affect the configured device.
nCE	N/A	All	Input	Active-low chip enable. The nCE pin activates the device with a low signal to allow configuration. The nCE pin must be held low during configuration, initialization, and user mode. In single device configuration, it must be tied low. In multi-device configuration, nCE of the first device is tied low while its nCEO pin is connected to nCE of the next device in the chain. The nCE pin must also be held low for successful JTAG programming of the device.
nCEO	N/A	All	Output	Output that drives low when device configuration is complete. In single device configuration, this pin is left floating. In multi-device configuration, this pin feeds the next device's nCE pin. The nCEO of the last device in the chain is left floating. The nCEO pin is powered by V_{CCPGM}.
ASDO	N/A	AS	Output	Control signal from the Stratix IV device to the serial configuration device in AS mode used to read out configuration data. In AS mode, ASDO has an internal pull-up resistor that is always active.
nCSO	N/A	AS	Output	Output control signal from the Stratix IV device to the serial configuration device in AS mode that enables the configuration device. In AS mode, nCSO has an internal pull-up resistor that is always active.

Table 10–13. *Dedicated Configuration Pins on the Stratix IV Device (Part 4 of 4)*

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
`DCLK`	N/A	Synchronous configuration schemes (PS, FPP, AS)	Input (PS, FPP) Output (AS)	In PS and FPP configuration, `DCLK` is the clock input used to clock data from an external source into the target device. Data is latched into the device on the rising edge of `DCLK`. In AS mode, `DCLK` is an output from the Stratix IV device that provides timing for the configuration interface. In AS mode, `DCLK` has an internal pull-up resistor (typically 25 kΩ) that is always active. After configuration, this pin is tri-stated. In schemes that use a configuration device, `DCLK` is driven low after configuration is done. In schemes that use a control host, `DCLK` must be driven either high or low, whichever is more convenient. Toggling this pin after configuration does not affect the configured device.
`DATA0`	N/A in AS mode. I/O in PS or FPP mode.	PS, FPP, AS	Input	Data input. In serial configuration modes, bit-wide configuration data is presented to the target device on the `DATA0` pin. In AS mode, `DATA0` has an internal pull-up resistor that is always active. After PS or FPP configuration, `DATA0` is available as a user I/O pin. The state of this pin depends on the **Dual-Purpose Pin** settings.
`DATA[7..1]`	I/O	Parallel configuration schemes (FPP)	Inputs	Data inputs. Byte-wide configuration data is presented to the target device on `DATA[7..0]`. In serial configuration schemes, they function as user I/O pins during configuration, which means they are tri-stated. After FPP configuration, `DATA[7..1]` are available as user I/O pins. The state of these pins depends on the **Dual-Purpose Pin** settings.

Table 10–14 describes the optional configuration pins. If these optional configuration pins are not enabled in the Quartus II software, they are available as general-purpose user I/O pins. Therefore, during configuration, these pins function as user I/O pins and are tri-stated with weak pull-up resistors.

Table 10–14. Optional Configuration Pins

Pin Name	User Mode	Pin Type	Description
CLKUSR	N/A if option is on. I/O if option is off.	Input	Optional user-supplied clock input synchronizes the initialization of one or more devices. Enable this pin by turning on the **Enable user-supplied start-up clock** (**CLKUSR**) option in the Quartus II software.
INIT_DONE	N/A if option is on. I/O if option is off.	Output open-drain	Use as Status pin to indicate when the device has initialized and is in user mode. When nCONFIG is low and during the beginning of configuration, the INIT_DONE pin is tri-stated and pulled high due to an external 10-kΩ pull-up resistor. Once the option bit to enable INIT_DONE is programmed into the device (during the first frame of configuration data), the INIT_DONE pin goes low. When initialization is complete, the INIT_DONE pin is released and pulled high and the device enters user mode. Thus, the monitoring circuitry must be able to detect a low-to-high transition. This pin is enabled by turning on the **Enable INIT_DONE output** option in the Quartus II software.
DEV_OE	N/A if option is on. I/O if option is off.	Input	Optional pin that allows you to override all tri-states on the device. When this pin is driven low, all I/O pins are tri-stated. When this pin is driven high, all I/O pins behave as programmed. Enable this pin by turning on the **Enable device-wide output enable** (**DEV_OE**) option in the Quartus II software.
DEV_CLRn	N/A if option is on. I/O if option is off.	Input	Optional pin that allows you to override all clears on all device registers. When this pin is driven low, all registers are cleared. When this pin is driven high, all registers behave as programmed. This pin is enabled by turning on the **Enable device-wide reset** (**DEV_CLRn**) option in the Quartus II software.

Table 10–15 describes the dedicated JTAG pins. JTAG pins must be kept stable before and during configuration to prevent accidental loading of JTAG instructions. The TDI, TMS, and TRST have weak internal pull-up resistors while TCK has a weak internal pull-down resistor (typically 25 kΩ). If you plan to use the SignalTap® embedded logic array analyzer, you must connect the JTAG pins of the Stratix IV device to a JTAG header on your board.

Table 10–15. Dedicated JTAG Pins (Part 1 of 2)

Pin Name	User Mode	Pin Type	Description
TDI	N/A	Input	Serial input pin for instructions as well as test and programming data. Data is shifted on the rising edge of TCK. The TDI pin is powered by the 2.5-V/3.0-V V_{CCPD} supply. If the JTAG interface is not required on the board, you can disable the JTAG circuitry by connecting this pin to logic high.
TDO	N/A	Output	Serial data output pin for instructions as well as test and programming data. Data is shifted out on the falling edge of TCK. The pin is tri-stated if data is not being shifted out of the device. The TDO pin is powered by V_{CCPD}. For recommendations about connecting a JTAG chain with multiple voltages across the devices in the chain, refer to the *JTAG Boundary Scan Testing* chapter in volume 1 of the *Stratix IV Device Handbook*. If the JTAG interface is not required on the board, you can disable the JTAG circuitry by leaving this pin unconnected.

Table 10–15. Dedicated JTAG Pins (Part 2 of 2)

Pin Name	User Mode	Pin Type	Description
TMS	N/A	Input	Input pin that provides the control signal to determine the transitions of the TAP controller state machine. TMS is evaluated on the rising edge of TCK. Therefore, you must set up TMS before the rising edge of TCK. Transitions within the state machine occur on the falling edge of TCK after the signal is applied to TMS. The TMS pin is powered by 2.5-V/3.0-V V_{CCPD}. If the JTAG interface is not required on the board, you can disable the JTAG circuitry by connecting this pin to logic high.
TCK	N/A	Input	Clock input to the BST circuitry. Some operations occur at the rising edge, while others occur at the falling edge. The TCK pin is powered by the 2.5-V/3.0-V V_{CCPD} supply. It is expected that the clock input waveform have a nominal 50% duty cycle. If the JTAG interface is not required on the board, you can disable the JTAG circuitry by connecting TCK to GND.
TRST	N/A	Input	Active-low input to asynchronously reset the boundary-scan circuit. The TRST pin is optional according to IEEE Std. 1149.1. The TRST pin is powered by the 2.5-V/3.0-V V_{CCPD} supply. Hold TMS at 1 or keep TCK static while TRST is changed from 0 to 1. If the JTAG interface is not required on the board, you can disable the JTAG circuitry by connecting the TRST pin to GND.

Configuration Data Decompression

Stratix IV devices support configuration data decompression, which saves configuration memory space and time. This feature allows you to store compressed configuration data in configuration devices or other memory and transmit this compressed bitstream to Stratix IV devices. During configuration, the Stratix IV device decompresses the bitstream in real time and programs its SRAM cells.

☞ Preliminary data indicates that compression typically reduces the configuration bitstream size by 35 to 55% based on the designs used.

Stratix IV devices support decompression in the FPP (when using a MAX II device or microprocessor + flash), fast AS, and PS configuration schemes. The Stratix IV decompression feature is not available in the JTAG configuration scheme.

☞ When using FPP mode, the intelligent host must provide a DCLK that is ×4 the data rate. Therefore, the configuration data must be valid for four DCLK cycles.

In PS mode, use the Stratix IV decompression feature, because sending compressed configuration data reduces configuration time.

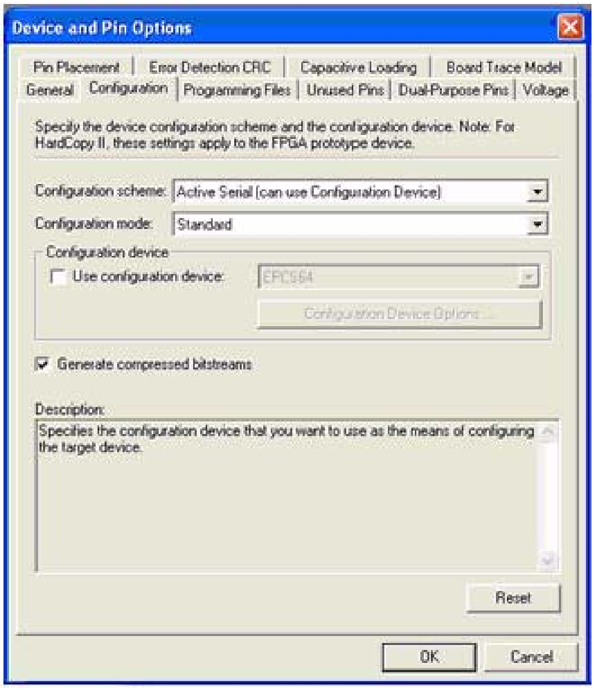
When you enable compression, the Quartus II software generates configuration files with compressed configuration data. This compressed file reduces the storage requirements in the configuration device or flash memory, and decreases the time needed to transmit the bitstream to the Stratix IV device. The time required by a Stratix IV device to decompress a configuration file is less than the time needed to transmit the configuration data to the device.

There are two ways to enable compression for Stratix IV bitstreams: before design compilation (in the Compiler Settings menu) and after design compilation (in the **Convert Programming Files** window).

To enable compression in the project's Compiler Settings menu, perform the following steps:

1. On the Assignments menu, click **Device** to bring up the **Settings** dialog box.
2. After selecting your Stratix IV device, open the **Device and Pin Options** window.
3. In the **Configuration** settings tab, turn on **Generate compressed bitstreams** (as shown in Figure 10–19).

Figure 10–19. Enabling Compression for Stratix IV Bitstreams in Compiler Settings



Specify the device configuration scheme and the configuration device. Note: For HardCopy II, these settings apply to the FPGA prototype device.

Configuration scheme: Active Serial (can use Configuration Device)

Configuration mode: Standard

Configuration device

□ Use configuration device: EPCS64

☑ Generate compressed bitstreams

Description:
Specifies the configuration device that you want to use as the means of configuring the target device.

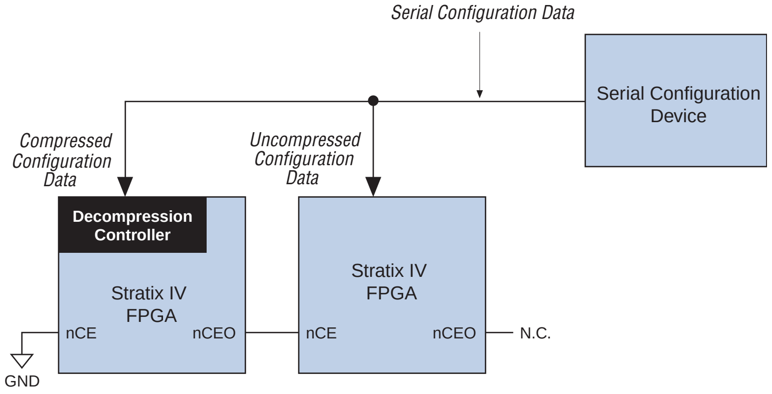
You can also enable compression when creating programming files from the **Convert Programming Files** window. To do this, perform the following steps:

1. On the File menu, click **Convert Programming Files**.
2. Select the programming file type (**.pof**, **.sram**, **.hex**, **.rbf**, or **.ttf**).
3. For **.pof** output files, select a configuration device.
4. In the **Input files to convert** box, select **SOF Data**.
5. Select **Add File** and add a Stratix IV device **.sof** file.
6. Select the name of the file you added to the **SOF Data** area and click **Properties**.
7. Check the **Compression** check box.

When multiple Stratix IV devices are cascaded, you can selectively enable the compression feature for each device in the chain if you are using a serial configuration scheme. Figure 10–20 depicts a chain of two Stratix IV devices. The first Stratix IV device has compression enabled and therefore receives a compressed bitstream from the configuration device. The second Stratix IV device has the compression feature disabled and receives uncompressed data.

In a multi-device FPP configuration chain (with a MAX II device or microprocessor + flash), all Stratix IV devices in the chain must either enable or disable the decompression feature. You cannot selectively enable the compression feature for each device in the chain because of the DATA and DCLK relationship.

Figure 10–20. Compressed and Uncompressed Configuration Data in the Same Configuration File


Serial Configuration Data
Serial Configuration Device
Compressed Configuration Data
Uncompressed Configuration Data
Decompression Controller
Stratix IV FPGA
nCE
nCEO
GND
Stratix IV FPGA
nCE
nCEO
N.C.

You can generate programming files for this setup by clicking **Convert Programming Files** on the File menu in the Quartus II software.

Remote System Upgrades

This section describes the functionality and implementation of the dedicated remote system upgrade circuitry. It also defines several concepts related to remote system upgrade, including factory configuration, application configuration, remote update mode, and user watchdog timer. Additionally, this section provides design guidelines for implementing remote system upgrades with the supported configuration schemes.

System designers sometimes face challenges such as shortened design cycles, evolving standards, and system deployments in remote locations. Stratix IV devices help overcome these challenges with their inherent reprogrammability and dedicated circuitry to perform remote system upgrades. Remote system upgrades help deliver feature enhancements and bug fixes without costly recalls, reduce time-to-market, extend product life, and help to avoid system downtime.

Stratix IV devices feature dedicated remote system upgrade circuitry. Soft logic (either the Nios® II embedded processor or user logic) implemented in a Stratix IV device can download a new configuration image from a remote location, store it in configuration memory, and direct the dedicated remote system upgrade circuitry to initiate a reconfiguration cycle. The dedicated circuitry performs error detection during and after the configuration process, recovers from any error condition by reverting back to a safe configuration image, and provides error status information.

Remote system upgrade is supported in fast active serial (AS) Stratix IV configuration schemes. You can also implement remote system upgrade in conjunction with advanced Stratix IV features such as real-time decompression of configuration data and design security using the advanced encryption standard (AES) for secure and efficient field upgrades. The largest serial configuration device currently supports 128 MBits of configuration bitstream.

Functional Description

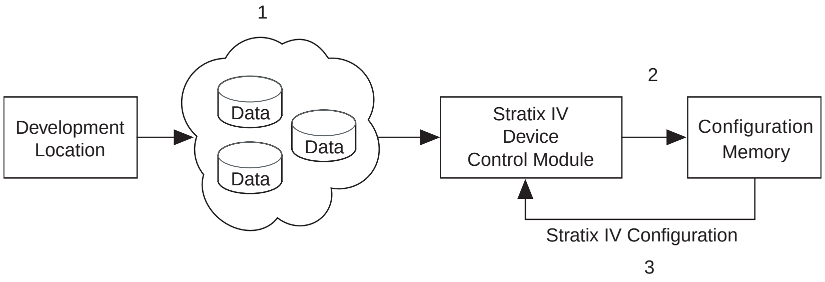
The dedicated remote system upgrade circuitry in Stratix IV devices manages remote configuration and provides error detection, recovery, and status information. User logic or a Nios II processor implemented in the Stratix IV device logic array provides access to the remote configuration data source and an interface to the system's configuration memory.

Stratix IV devices have remote system upgrade processes that involves the following steps:

1. A Nios II processor (or user logic) implemented in the Stratix IV device logic array receives new configuration data from a remote location. The connection to the remote source uses a communication protocol such as the transmission control protocol/Internet protocol (TCP/IP), peripheral component interconnect (PCI), user datagram protocol (UDP), universal asynchronous receiver/transmitter (UART), or a proprietary interface.
2. The Nios II processor (or user logic) stores this new configuration data in non-volatile configuration memory.
3. The Nios II processor (or user logic) initiates a reconfiguration cycle with the new or updated configuration data.
4. The dedicated remote system upgrade circuitry detects and recovers from any error(s) that might occur during or after the reconfiguration cycle and provides error status information to the user design.

Figure 10–21 shows the steps required for performing remote configuration updates. (The numbers in Figure 10–21 coincide with the steps just mentioned.)

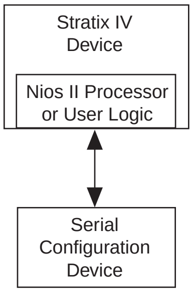
Figure 10–21. Functional Diagram of Stratix IV Remote System Upgrade



Stratix IV devices only support remote system upgrade in the single device fast AS configuration scheme.

Figure 10–22 shows a block diagram for implementing a remote system upgrade with the Stratix IV fast AS configuration scheme.

Figure 10–22. Remote System Upgrade Block Diagram for Stratix IV Fast AS Configuration Scheme



You must set the mode select pins (`MSEL[2..0]`) to fast AS mode to use the remote system upgrade in your system. Table 10–16 lists the `MSEL` pin settings for Stratix IV devices in standard configuration mode and remote system upgrade mode. The following sections describe the remote update of the remote system upgrade mode.

For more information about standard configuration schemes supported in Stratix IV devices, refer to "Configuration Schemes" on page 10–2.

Table 10–16. Stratix IV Remote System Upgrade Modes

Configuration Scheme	MSEL[2..0]	Remote System Upgrade Mode
Fast AS (40 MHz)	011	Standard
	011	Remote update *(1)*

Note to Table 10–16:

(1) EPCS16, EPCS64, and EPCS128 serial configuration devices support a `DCLK` up to 40 MHz. Refer to the *Serial Configuration Devices (EPCS1, EPCS4, EPCS16, EPCS64, and EPCS128) Data Sheet* chapter in volume 2 of the *Configuration Handbook* for more information.

☞ When using fast AS mode, you must select the remote update mode in the Quartus II software and insert the ALTREMOTE_UPDATE megafunction to access the circuitry. Refer to "ALTREMOTE_UPDATE Megafunction" on page 10–56 for more information.

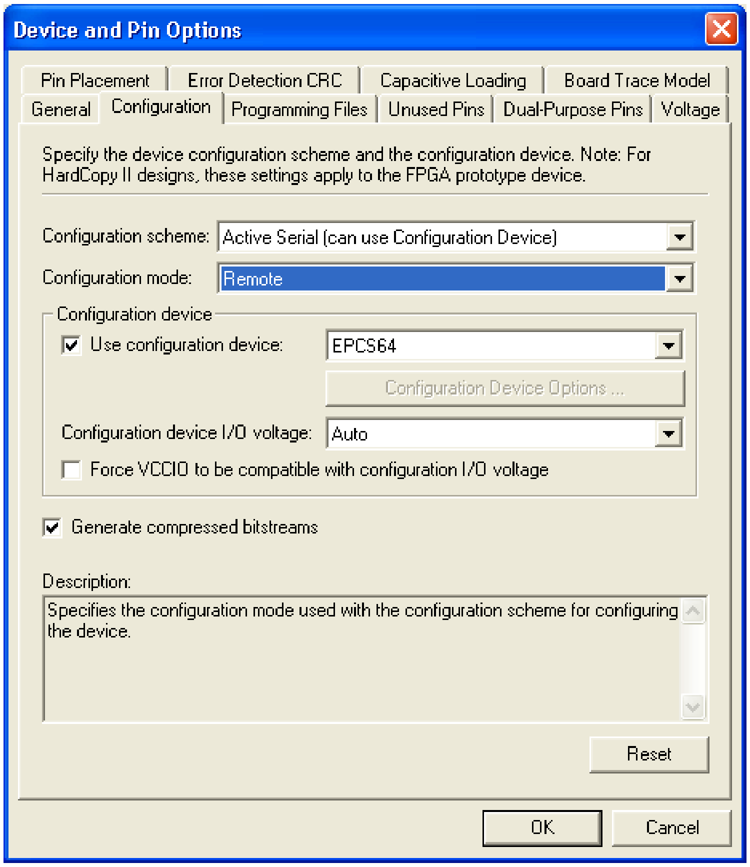
Enabling Remote Update

You can enable remote update for Stratix IV devices in the Quartus II software before design compilation (in the Compiler Settings menu). In remote update mode, the auto-restart configuration after error option is always enabled. To enable remote update in the project's compiler settings, perform the following steps in the Quartus II software:

1. On the Assignment menu, click **Device**. The **Settings** dialog box appears.
2. Click **Device and Pin Options**. The **Device and Pin Options** dialog box appears.
3. Click the **Configuration** tab.
4. From the **Configuration scheme** list, select **Active Serial** (you can also use **Configuration Device**) (Figure 10–23).
5. From the **Configuration Mode** list, select **Remote** (Figure 10–23).
6. Click **OK**.

7. In the **Settings** dialog box, click **OK**.

Figure 10–23. Enabling Remote Update for Stratix IV Devices in the Compiler Settings Menu



Specify the device configuration scheme and the configuration device. Note: For HardCopy II designs, these settings apply to the FPGA prototype device.

Configuration scheme: Active Serial (can use Configuration Device)

Configuration mode: Remote

Configuration device

☑ Use configuration device: EPCS64

Configuration device I/O voltage: Auto

□ Force VCCIO to be compatible with configuration I/O voltage

☑ Generate compressed bitstreams

Description:

Specifies the configuration mode used with the configuration scheme for configuring the device.

Configuration Image Types

When performing a remote system upgrade, Stratix IV device configuration bitstreams are classified as factory configuration images or application configuration images. An image, also referred to as a configuration, is a design loaded into the Stratix IV device that performs certain user-defined functions.

Each Stratix IV device in your system requires one factory image or the addition of one or more application images. The factory image is a user-defined fall-back, or safe configuration, and is responsible for administering remote updates in conjunction with the dedicated circuitry. Application images implement user-defined functionality in the target Stratix IV device. You may include the default application image functionality in the factory image.

A remote system upgrade involves storing a new application configuration image or updating an existing one using the remote communication interface. After an application configuration image is stored or updated remotely, the user design in the Stratix IV device initiates a reconfiguration cycle with the new image. Any errors during or after this cycle are detected by the dedicated remote system upgrade circuitry and cause the device to automatically revert to the factory image. The factory image then performs error processing and recovery. The factory configuration is written to the serial configuration device only once by the system manufacturer and should not be remotely updated. On the other hand, application configurations may be remotely updated in the system. Both images can initiate system reconfiguration.

Remote System Upgrade Mode

Remote system upgrade has one mode of operation: remote update mode. The remote update mode allows you to determine the functionality of your system upon power-up and offers different features.

Overview

In remote update mode, Stratix IV devices load the factory configuration image upon power up. The user-defined factory configuration determines which application configuration is to be loaded and triggers a reconfiguration cycle. The factory configuration may also contain application logic.

When used with serial configuration devices, remote update mode allows an application configuration to start at any flash sector boundary. For example, this translates to a maximum of 128 pages in the EPCS64 device and 32 pages in the EPCS16 device, where the minimum size of each page is 512 KBits. Altera recommends not using the same page in the serial configuration devices for two images. Additionally, remote update mode features a user watchdog timer that determines the validity of an application configuration.

Remote Update Mode

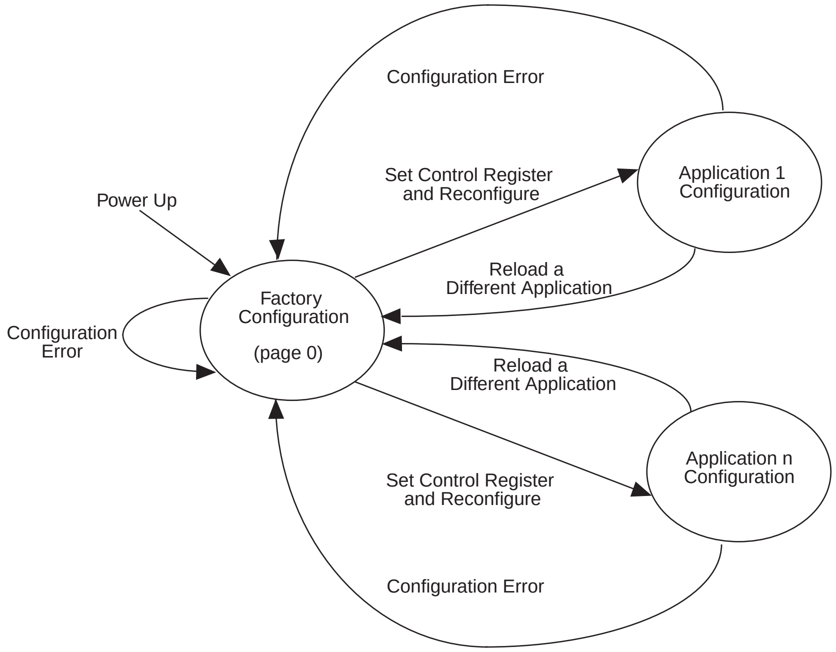
When a Stratix IV device is first powered up in remote update mode, it loads the factory configuration located at page zero (page registers `PGM[23:0] = 24'b0`). Always store the factory configuration image for your system at page address zero. This corresponds to the start address location 0×000000 in the serial configuration device.

The factory image is user-designed and contains soft logic to:

- Process any errors based on status information from the dedicated remote system upgrade circuitry
- Communicate with the remote host and receive new application configurations and store this new configuration data in the local non-volatile memory device
- Determine which application configuration is to be loaded into the Stratix IV device
- Enable or disable the user watchdog timer and load its time-out value (optional)
- Instruct the dedicated remote system upgrade circuitry to initiate a reconfiguration cycle

Figure 10–24 shows the transitions between the factory and application configurations in remote update mode.

Figure 10–24. Transitions between Configurations in Remote Update Mode


Configuration Error
Set Control Register and Reconfigure
Application 1 Configuration
Power Up
Reload a Different Application
Factory Configuration (page 0)
Configuration Error
Reload a Different Application
Application n Configuration
Set Control Register and Reconfigure
Configuration Error

After power up or a configuration error, the factory configuration logic is loaded automatically. The factory configuration also needs to specify whether to enable the user watchdog timer for the application configuration and if enabled, to include the timer setting information as well.

The user watchdog timer ensures that the application configuration is valid and functional. The timer must be continually reset within a specific amount of time during user mode operation of an application configuration. Only valid application configurations contain the logic to reset the timer in user mode. This timer reset logic should be part of a user-designed hardware and/or software health monitoring signal that indicates error-free system operation. If the timer is not reset in a specific amount of time; for example, the user application configuration detects a functional problem or if the system hangs, the dedicated circuitry will update the remote system upgrade status register, triggering the loading of the factory configuration.

☞ The user watchdog timer is automatically disabled for factory configurations. For more information about the user watchdog timer, refer to "User Watchdog Timer" on page 10–55.

If there is an error while loading the application configuration, the cause of the reconfiguration is written by the dedicated circuitry to the remote system upgrade status register. Actions that cause the remote system upgrade status register to be written are:

- nSTATUS driven low externally
- Internal CRC error
- User watchdog timer time-out

- A configuration reset (logic array nCONFIG signal or external nCONFIG pin assertion to low)

Stratix IV devices automatically load the factory configuration located at page address zero. This user-designed factory configuration can read the remote system upgrade status register to determine the reason for the reconfiguration. The factory configuration then takes appropriate error recovery steps and writes to the remote system upgrade control register to determine the next application configuration to be loaded.

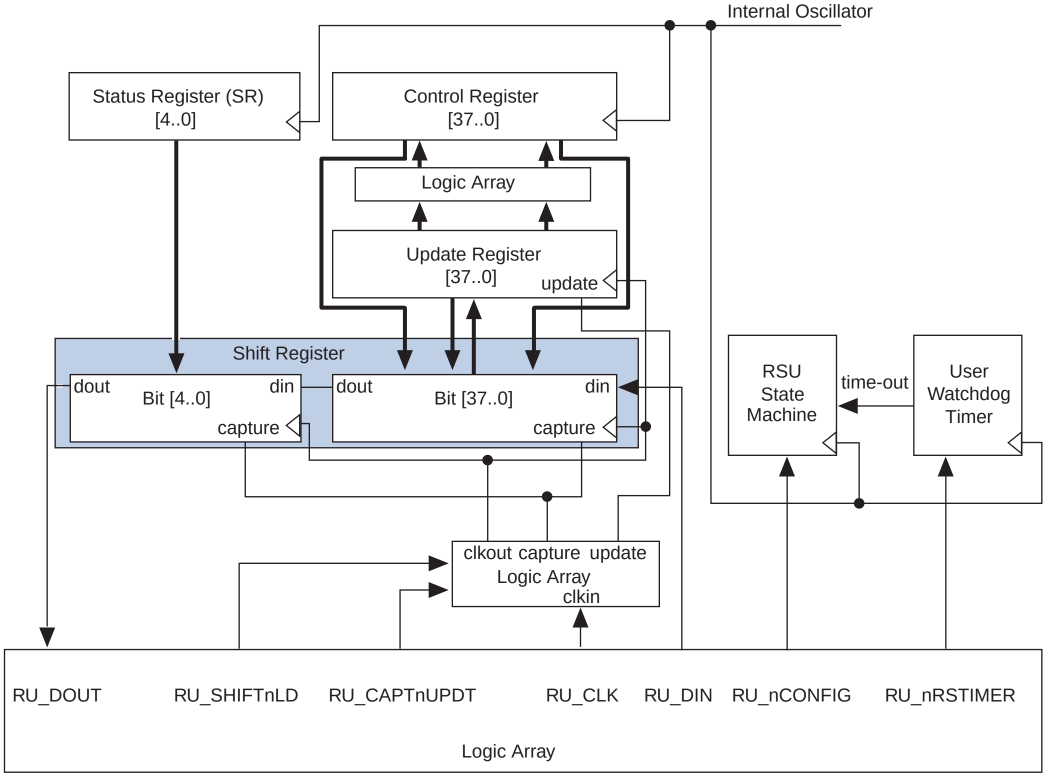
When Stratix IV devices successfully load the application configuration, they enter into user mode. In user mode, the soft logic (Nios II processor or state machine and the remote communication interface) assists the Stratix IV device in determining when a remote system update is arriving. When a remote system update arrives, the soft logic receives the incoming data, writes it to the configuration memory device, and triggers the device to load the factory configuration. The factory configuration reads the remote system upgrade status register and control register, determines the valid application configuration to load, writes the remote system upgrade control register accordingly, and initiates system reconfiguration.

Dedicated Remote System Upgrade Circuitry

This section describes the implementation of the Stratix IV remote system upgrade dedicated circuitry. The remote system upgrade circuitry is implemented in hard logic. This dedicated circuitry interfaces to the user-defined factory and application configurations implemented in the Stratix IV device logic array to provide the complete remote configuration solution. The remote system upgrade circuitry contains the remote system upgrade registers, a watchdog timer, and a state machine that controls those components.

Figure 10–25 shows the remote system upgrade block's data path.

Figure 10–25. Remote System Upgrade Circuit Data Path *(Note 1)*



Note to Figure 10–25:

(1) The `RU_DOUT`, `RU_SHIFTnLD`, `RU_CAPTnUPDT`, `RU_CLK`, `RU_DIN`, `RU_nCONFIG` and `RU_nRSTIMER` signals are internally controlled by the ALTREMOTE_UPDATE megafunction.

Remote System Upgrade Registers

The remote system upgrade block contains a series of registers that store the page addresses, watchdog timer settings, and status information. These registers are detailed in Table 10–17.

Table 10–17. ***Remote System Upgrade Registers (Part 1 of 2)***

Register	Description
Shift register	This register is accessible by the logic array and allows the update, status, and control registers to be written and sampled by user logic.
Control register	This register contains the current page address, user watchdog timer settings, and one bit specifying whether the current configuration is a factory configuration or an application configuration. During a read operation in an application configuration, this register is read into the shift register. When a reconfiguration cycle is initiated, the contents of the update register are written into the control register.

Table 10–17. ***Remote System Upgrade Registers (Part 2 of 2)***

Register	Description
Update register	This register contains data similar to that in the control register. However, it can only be updated by the factory configuration by shifting data into the shift register and issuing an update operation. When a reconfiguration cycle is triggered by the factory configuration, the control register is updated with the contents of the update register. During a capture in a factory configuration, this register is read into the shift register.
Status register	This register is written to by the remote system upgrade circuitry on every reconfiguration to record the cause of the reconfiguration. This information is used by the factory configuration to determine the appropriate action following a reconfiguration. During a capture cycle, this register is read into the shift register.

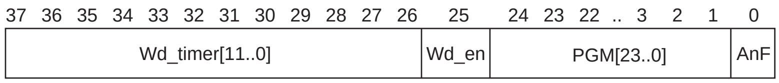
The remote system upgrade control and status registers are clocked by the 10-MHz internal oscillator (the same oscillator that controls the user watchdog timer). However, the remote system upgrade shift and update registers are clocked by the user clock input (`RU_CLK`).

Remote System Upgrade Control Register

The remote system upgrade control register stores the application configuration page address and user watchdog timer settings. The control register functionality depends on the remote system upgrade mode selection. In remote update mode, the control register page address bits are set to all zeros (`24'b0 = 0×000000`) at power up to load the factory configuration. A factory configuration in remote update mode has write access to this register.

The control register bit positions are shown in Figure 10–26 and defined in Table 10–18. In the figure, the numbers show the bit position of a setting within a register. For example, bit number 25 is the enable bit for the watchdog timer.

Figure 10–26. Remote System Upgrade Control Register



The application-not-factory (`AnF`) bit indicates whether the current configuration loaded in the Stratix IV device is the factory configuration or an application configuration. This bit is set low by the remote system upgrade circuitry when an error condition causes a fall-back to the factory configuration. When the `AnF` bit is high, the control register access is limited to read operations. When the `AnF` bit is low, the register allows write operations and disables the watchdog timer.

In remote update mode, the factory configuration design sets this bit high (1'b1) when updating the contents of the update register with the application page address and watchdog timer settings.

Table 10–18 shows the remote system upgrade control register contents.

Table 10–18. Remote System Upgrade Control Register Contents

Control Register Bit	Remote System Upgrade Mode	Value *(2)*	Definition
AnF *(1)*	Remote update	1'b0	Application not factory
PGM[23..0]	Remote update	24'b0×000000	AS configuration start address (StAdd[23..0])
Wd_en	Remote update	1'b0	User watchdog timer enable bit
Wd_timer[11..0]	Remote update	12'b000000000000	User watchdog time-out value (most significant 12 bits of 29-bit count value: {Wd_timer[11..0], 17'b0})

Notes to Table 10–18:

(1) In remote update mode, the remote configuration block does not update the AnF bit automatically (you can update it manually).

(2) This is the default value of the control register bit.

Remote System Upgrade Status Register

The remote system upgrade status register specifies the reconfiguration trigger condition. The various trigger and error conditions include:

- Cyclic redundancy check (CRC) error during application configuration
- nSTATUS assertion by an external device due to an error
- Stratix IV device logic array triggered a reconfiguration cycle, possibly after downloading a new application configuration image
- External configuration reset (nCONFIG) assertion
- User watchdog timer time-out

Figure 10–27 and Table 10–19 specify the contents of the status register. The numbers in the figure show the bit positions within a 5-bit register.

Figure 10–27. Remote System Upgrade Status Register

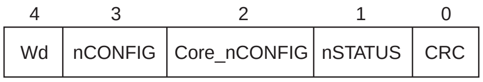


Table 10–19. Remote System Upgrade Status Register Contents

Status Register Bit	Definition	POR Reset Value
CRC (from configuration)	CRC error caused reconfiguration	1 bit '0'
nSTATUS	nSTATUS caused reconfiguration	1 bit '0'
CORE_nCONFIG *(1)*	Device logic array caused reconfiguration	1 bit '0'
nCONFIG	nCONFIG caused reconfiguration	1 bit '0'
Wd	Watchdog timer caused reconfiguration	1 bit '0'

Note to Table 10–19:

(1) Logic array reconfiguration forces the system to load the application configuration data into the Stratix IV device. This occurs after the factory configuration specifies the appropriate application configuration page address by updating the update register.

Remote System Upgrade State Machine

The remote system upgrade control and update registers have identical bit definitions, but serve different roles (refer to Table 10–17 on page 10–52). While both registers can only be updated when the device is loaded with a factory configuration image, the update register writes are controlled by the user logic; the control register writes are controlled by the remote system upgrade state machine.

In factory configurations, the user logic sends the AnF bit (set high), the page address, and the watchdog timer settings for the next application configuration bit to the update register. When the logic array configuration reset (RU_nCONFIG) goes low, the remote system upgrade state machine updates the control register with the contents of the update register and initiates system reconfiguration from the new application page.

In the event of an error or reconfiguration trigger condition, the remote system upgrade state machine directs the system to load a factory or application configuration (page zero or page one, based on the mode and error condition) by setting the control register accordingly. Table 10–20 lists the contents of the control register after such an event occurs for all possible error or trigger conditions.

The remote system upgrade status register is updated by the dedicated error monitoring circuitry after an error condition but before the factory configuration is loaded.

Table 10–20. Control Register Contents after an Error or Reconfiguration Trigger Condition

Reconfiguration Error/Trigger	Control Register Setting Remote Update
nCONFIG reset	All bits are 0
nSTATUS error	All bits are 0
CORE triggered reconfiguration	Update register
CRC error	All bits are 0
Wd time out	All bits are 0

Capture operations during factory configuration access the contents of the update register. This feature is used by the user logic to verify that the page address and watchdog timer settings were written correctly. Read operations in application configurations access the contents of the control register. This information is used by the user logic in the application configuration.

User Watchdog Timer

The user watchdog timer prevents a faulty application configuration from stalling the device indefinitely. The system uses the timer to detect functional errors after an application configuration is successfully loaded into the Stratix IV device.

The user watchdog timer is a counter that counts down from the initial value loaded into the remote system upgrade control register by the factory configuration. The counter is 29-bits wide and has a maximum count value of 2^{29}. When specifying the user watchdog timer value, specify only the most significant 12 bits. The granularity of the timer setting is 2^{15} cycles. The cycle time is based on the frequency of the 10-MHz internal oscillator. Table 10–21 specifies the operating range of the 10-MHz internal oscillator.

Table 10–21. 10-MHz Internal Oscillator Specifications *(Note 1)*

Minimum	Typical	Maximum	Units
4.3	5.3	10	MHz

Note to Table 10–21:

(1) These values are preliminary.

The user watchdog timer begins counting once the application configuration enters device user mode. This timer must be periodically reloaded or reset by the application configuration before the timer expires by asserting RU_nRSTIMER. If the application configuration does not reload the user watchdog timer before the count expires, a time-out signal is generated by the remote system upgrade dedicated circuitry. The time-out signal tells the remote system upgrade circuitry to set the user watchdog timer status bit (Wd) in the remote system upgrade status register and reconfigures the device by loading the factory configuration.

The user watchdog timer is not enabled during the configuration cycle of the device. Errors during configuration are detected by the CRC engine. Also, the timer is disabled for factory configurations. Functional errors should not exist in the factory configuration because it is stored and validated during production and is never updated remotely.

☞ The user watchdog timer is disabled in factory configurations and during the configuration cycle of the application configuration. It is enabled after the application configuration enters user mode.

Quartus II Software Support

Quartus II software provides the flexibility to include the remote system upgrade interface between the Stratix IV device logic array and the dedicated circuitry, generate configuration files for productions, and allows remote programming of the system configuration memory.

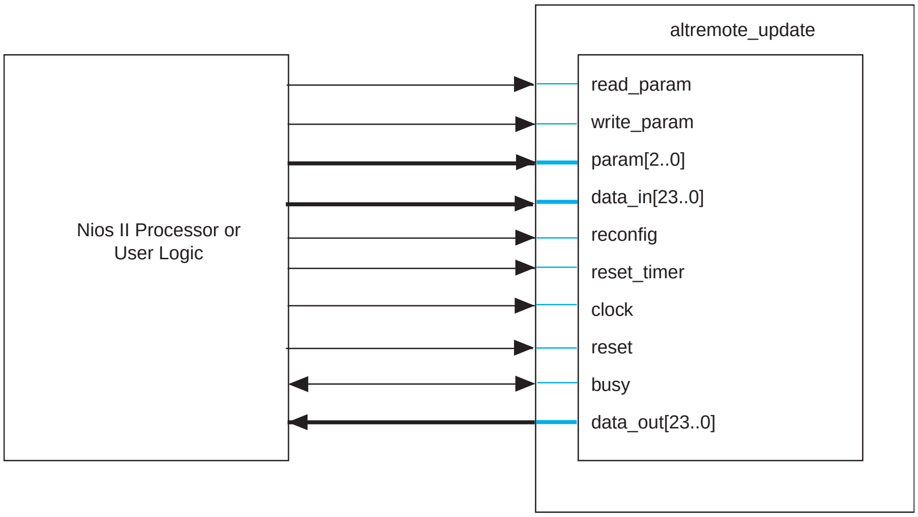
The ALTREMOTE_UPDATE megafunction is the implementation option in the Quartus II software that is used for the interface between the remote system upgrade circuitry and the device logic array interface. Using the megafunction block instead of creating your own logic saves design time and offers more efficient logic synthesis and device implementation.

ALTREMOTE_UPDATE Megafunction

The ALTREMOTE_UPDATE megafunction provides a memory-like interface to the remote system upgrade circuitry and handles the shift register read and write protocol in the Stratix IV device logic. This implementation is suitable for designs that implement the factory configuration functions using a Nios II processor or user logic in the device.

Figure 10–28 shows the interface signals between the ALTREMOTE_UPDATE megafunction and Nios II processor or user logic.

Figure 10–28. Interface Signals between the ALTREMOTE_UPDATE Megafunction and the Nios II Processor



For more information about the ALTREMOTE_UPDATE megafunction and the description of ports listed in Figure 10–28, refer to the *altremote_update Megafunction User Guide*.

Design Security

This section provides an overview of the design security feature and its implementation on Stratix IV devices using advanced encryption standard (AES). It also covers the new security modes available in Stratix IV devices for you to utilize in your designs.

As Stratix IV devices start to play a role in larger and more critical designs in competitive commercial and military environments, it is increasingly important to protect the designs from copying, reverse engineering, and tampering.

Stratix IV devices address these concerns with both volatile and non-volatile security feature support. Stratix IV devices have the ability to decrypt configuration bitstreams using the AES algorithm, an industry-standard encryption algorithm that is FIPS-197 certified. Stratix IV devices have a design security feature which utilizes a 256-bit security key.

Stratix IV devices store configuration data in SRAM configuration cells during device operation. Because SRAM memory is volatile, the SRAM cells must be loaded with configuration data each time the device powers up. It is possible to intercept configuration data when it is being transmitted from the memory source (flash memory or a configuration device) to the device. The intercepted configuration data could then be used to configure another device.

When using the Stratix IV design security feature, the security key is stored in the Stratix IV device. Depending on the security mode, you can configure the Stratix IV device using a configuration file that is encrypted with the same key, or for board testing, configured with a normal configuration file.

The design security feature is available when configuring Stratix IV devices using the fast passive parallel (FPP) configuration mode with an external host (such as a MAX II device or microprocessor), or when using fast active serial (AS) or passive serial (PS) configuration schemes. The design security feature is also available in remote update with fast AS configuration mode. The design security feature is not available when you are configuring your Stratix IV device using Joint Test Action Group (JTAG)-based configuration. For more details, refer to "Supported Configuration Schemes" on page 10–61.

☞ When using a serial configuration scheme such as PS or fast AS, configuration time is the same whether or not the design security feature is enabled. If the FPP scheme is used with the design security or decompression feature, a ×4 DCLK is required. This results in a slower configuration time when compared to the configuration time of a Stratix IV device that has neither the design security nor the decompression feature enabled.

Stratix IV Security Protection

Stratix IV device designs are protected from copying, reverse engineering, and tampering using configuration bitstream encryption.

Security Against Copying

The security key is securely stored in the Stratix IV device and cannot be read out through any interfaces. In addition, as configuration file read-back is not supported in Stratix IV devices, the design information cannot be copied.

Security Against Reverse Engineering

Reverse engineering from an encrypted configuration file is very difficult and time consuming because the Stratix IV configuration file formats are proprietary and the file contains million of bits which require specific decryption. Reverse engineering the Stratix IV device is just as difficult because the device is manufactured on the most advanced 40-nm process technology.

Security Against Tampering

The non-volatile keys are one-time programmable. Once the Tamper Protection bit is set in the key programming file generated by the Quartus II software, the Stratix IV device can only be configured with configuration files encrypted with the same key.

AES Decryption Block

The main purpose of the AES decryption block is to decrypt the configuration bitstream prior to entering data decompression or configuration.

Prior to receiving encrypted data, you must enter and store the 256-bit security key in the device. You can choose between a non-volatile security key and a volatile security key with battery backup.

The security key is scrambled prior to storing it in the key storage to make it more difficult for anyone to retrieve the stored key using de-capsulation of the device.

Flexible Security Key Storage

Stratix IV devices support two types of security key programming: volatile and non-volatile keys. Table 10–22 shows the differences between volatile keys and non-volatile keys.

Table 10–22. Security Key Options

Options	Volatile Key	Non-Volatile Key
Key programmability	Reprogrammable and erasable	One-time programmable
External battery	Required	Not required
Key programming method *(1)*	On-board	On and off board
Design protection	Secure against copying and reverse engineering	Secure against copying and reverse engineering. Tamper resistant if tamper protection bit is set.

Note to Table 10–22:

(1) Key programming is carried out using the JTAG interface.

You can program the non-volatile key to the Stratix IV device without an external battery. Also, there are no additional requirements to any of the Stratix IV power supply inputs.

V_{CCBAT} is a dedicated power supply for volatile key storage and not shared with other on-chip power supplies, such as V_{CCIO} or V_{CC}. V_{CCBAT} continuously supplies power to the volatile register regardless of the on-chip supply condition. The nominal voltage for this supply is 3.0 V, while its valid operating range is from 1.2 to 3.3 V. If you do not use the volatile security key, you may connect the V_{CCBAT} to either GND or a 3.0-V power supply.

☞ After power-up, you will need to wait 300 ms (PORSEL = 0) or 12 ms (PORSEL = 1) before beginning the key programming to ensure that V_{CCBAT} is at its full rail.

☞ As an example, the following are lithium coin-cell type batteries used for volatile key storage purposes: BR1220 (-30° to +80°C) and BR2477A (-40°C to +125°C).

For more information about battery specifications, refer to the *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook*.

Stratix IV Design Security Solution

Stratix IV devices are SRAM-based devices. To provide design security, Stratix IV devices require a 256-bit security key for configuration bitstream encryption.

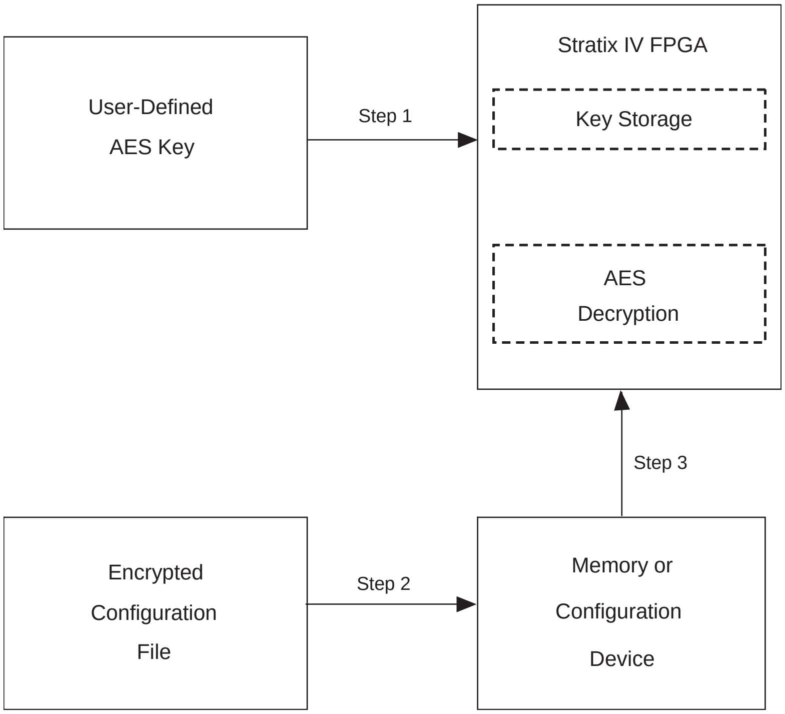
You can carry out secure configuration in the following three steps, as shown in Figure 10–29:

1. Program the security key into the Stratix IV device.
2. Program the user-defined 256-bit AES keys to the Stratix IV device through the JTAG interface.
3. Encrypt the configuration file and store it in the external memory.

4. Encrypt the configuration file with the same 256-bit keys used to program the Stratix IV device. Encryption of the configuration file is done using the Quartus II software. The encrypted configuration file is then loaded into the external memory, such as a configuration or flash device.
5. Configure the Stratix IV device.

At system power-up, the external memory device sends the encrypted configuration file to the Stratix IV device.

Figure 10–29. Design Security *(Note 1)*



Note to Figure 10–29:

(1) Step 1, Step 2, and Step 3 correspond to the procedure detailed in "Design Security" on page 10–57.

Security Modes Available

The following security modes are available on the Stratix IV device:

Volatile Key

Secure operation with volatile key programmed and required external battery: this mode accepts both encrypted and unencrypted configuration bitstreams. Use the unencrypted configuration bitstream support for board-level testing only.

Non-Volatile Key

Secure operation with one time programmable (OTP) security key programmed: this mode accepts both encrypted and unencrypted configuration bitstreams. Use the unencrypted configuration bitstream support for board level testing only.

Non-Volatile Key with Tamper Protection Bit Set

Secure operation in tamper resistant mode with OTP security key programmed: only encrypted configuration bitstreams are allowed to configure the device. Tamper protection disables JTAG configuration with unencrypted configuration bitstream.

☞ Enabling the Tamper Protection bit disables the test mode in Stratix IV devices. This process is irreversible and prevents Altera from carry-out failure analysis if test mode is disabled. Contact Altera Technical Support to enable the tamper protection bit.

No Key Operation

Only unencrypted configuration bitstreams are allowed to configure the device.

Table 10–23 summarizes the different security modes and configuration bitstream supported for each mode.

Table 10–23. Security Modes Supported

Mode (1)	Function	Configuration File
Volatile key	Secure	Encrypted
	Board-level testing	Unencrypted
Non-volatile key	Secure	Encrypted
	Board-level testing	Unencrypted
Non-volatile key with tamper protection bit set	Secure (tamper resistant) (2)	Encrypted

Notes to Table 10–23:

(1) In the No key operation, only the unencrypted configuration file is supported.

(2) The tamper protection bit setting does not prevent the device from being reconfigured.

Supported Configuration Schemes

The Stratix IV device supports only selected configuration schemes, depending on the security mode you select when you encrypt the Stratix IV device.

Figure 10–30 shows the restrictions of each security mode when encrypting Stratix IV devices.

Figure 10–30. Stratix IV Security Modes—Sequence and Restrictions

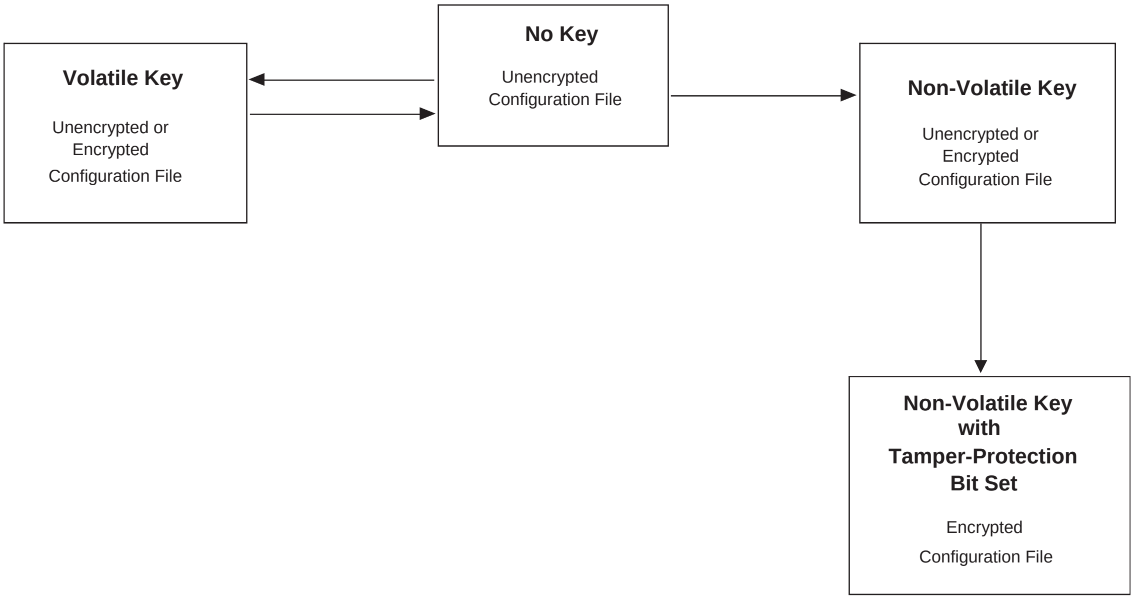

Volatile Key
Unencrypted or Encrypted Configuration File
No Key
Unencrypted Configuration File
Non-Volatile Key
Unencrypted or Encrypted Configuration File
Non-Volatile Key with Tamper-Protection Bit Set
Encrypted Configuration File

Table 10–24 shows the configuration modes allowed in each of the security modes.

Table 10–24. Allowed Configuration Modes for Various Security Modes *(Note 1)*

Security Mode	Configuration File	Allowed Configuration Modes
No key	Unencrypted	All configuration modes that do not engage the design security feature.
Secure with volatile key	Encrypted	■ Passive serial with AES (and/or with decompression) ■ Fast passive parallel with AES (and/or with decompression) ■ Remote update fast AS with AES (and/or with decompression) ■ Fast AS (and/or with decompression)
Board-level testing with volatile key	Unencrypted	All configuration modes that do not engage the design security feature.
Secure with non-volatile key	Encrypted	■ Passive serial with AES (and/or with decompression) ■ Fast passive parallel with AES (and/or with decompression) ■ Remote update fast AS with AES (and/or with decompression) ■ Fast AS (and/or with decompression)
Board-level testing with non-volatile key	Unencrypted	All configuration modes that do not engage the design security feature.
Secure in tamper resistant mode using non-volatile key with tamper protection set	Encrypted	■ Passive serial with AES (and/or with decompression) ■ Fast passive parallel with AES (and/or with decompression) ■ Remote update fast AS with AES (and/or with decompression) ■ Fast AS (and/or with decompression)

Note to Table 10–24:

(1) There is no impact to the configuration time required compared to unencrypted configuration modes except fast passive parallel with AES (and/or decompression), which requires DCLK that is 4× the data rate.

The design security feature is available in all configuration methods except JTAG. Therefore, you can use the design security feature in FPP mode (when using an external controller, such as a MAX II device or a microprocessor and a flash memory), or in fast AS and PS configuration schemes.

Table 10–25 summarizes the configuration schemes that support the design security feature both for volatile key and non-volatile key programming.

Table 10–25. Design Security Configuration Schemes Availability

Configuration Scheme	Configuration Method	Design Security
FPP	MAX II device or microprocessor and flash memory	✓
	Enhanced configuration device	—
Fast AS	Serial configuration device	✓
PS	MAX II device or microprocessor and flash memory	✓
	Download cable	✓
JTAG *(2)*	MAX II device or microprocessor and flash memory	—
	Download cable	—

Notes to Table 10–25:

(1) In this mode, the host system must send a DCLK that is 4× the data rate.

(2) JTAG configuration supports only unencrypted configuration file.

You can use the design security feature with other configuration features, such as compression and remote system upgrade features. When you use compression with the design security feature, the configuration file is first compressed and then encrypted using the Quartus II software. During configuration, the Stratix IV device first decrypts and then decompresses the configuration file.

Conclusion

You can configure Stratix IV devices in a number of different schemes to fit your system’s needs. In addition, Stratix IV devices provide the configuration data decompression feature to save configuration memory space and time.

Stratix IV devices offer remote system upgrade capability, where you can upgrade a system in real-time through any network. Remote system upgrade helps to deliver feature enhancements and bug fixes without costly recalls, reduces time to market, and extends product life cycles. The dedicated remote system upgrade circuitry in Stratix IV devices provides error detection, recovery, and status information to ensure reliable reconfiguration.

The need for design security is increasing as devices move from glue logic to implementing critical system functions. Stratix IV devices address this concern by providing built-in design security feature which protect your designs against IP theft and tampering of your configuration files.

Referenced Documents

This chapter references the following documents:

- *altremote_update Megafunction User Guide*
- *AN 122: Using Jam STAPL for ISP and ICR via an Embedded Processor*
- *AN 370: Using the Serial FlashLoader with Quartus II Software*
- *AN 386: Using the MAX II Parallel Flash Loader with the Quartus II Software*
- *ByteBlaster II Download Cable User Guide*
- *ByteBlasterMV Download Cable User Guide*
- *Configuring Mixed Altera FPGA Chains* in volume 2 of the *Configuration Handbook*
- *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook*
- *Device Configuration Options* and *Configuration File Formats* chapters in volume 2 of the *Configuration Handbook*
- *EthernetBlaster Communications Cable User Guide*
- *JTAG Boundary Scan Testing* chapter in volume 1 of the *Stratix IV Device Handbook*
- *MasterBlaster Serial/USB Communications Cable User Guide*
- *Programming Support for Jam STAPL Language*
- *Serial Configuration Devices (EPCS1, EPCS4, EPCS16, EPCS64, and EPCS128) Data Sheet* in volume 2 of the *Configuration Handbook*
- *USB Blaster USB Download Cable User Guide*

Revision History

Table 10–26 shows the revision history for this document.

Table 10–26. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	Updated sections: ■ "Fast Active Serial Configuration (Serial Configuration Devices)" on page 10–14, "JTAG Configuration" on page 10–31 Updated figures: ■ Figure 10–4, Figure 10–5, Figure 10–6, Figure 10–13 Updated tables: ■ Table 10–2, Table 10–13	Medium update.
May 2008 v1.0	Initial release.	—

11. SEU Mitigation in Stratix IV Devices

SIV51011-2.0

Introduction

In critical applications such as avionics, telecommunications, system control, and military applications, it is important to be able to do the following:

- Confirm that the configuration data stored in a Stratix® IV device is correct.
- Alert the system to the occurrence of a configuration error.

☞ The error detection feature has been enhanced in the Stratix IV family. Similar to Stratix III devices, the error detection and recovery time for single event upset (SEU) in Stratix IV devices is reduced in comparison to Stratix II devices.

Dedicated circuitry is built into Stratix IV devices and consists of a cyclic redundancy check (CRC) error detection feature that optionally checks for SEUs continuously and automatically.

This section describes how to activate and use the error detection CRC feature when a Stratix IV device is in user mode and describes how to recover from configuration errors caused by CRC errors.

☞ For Stratix IV devices, use of the error detection CRC feature is provided in the Quartus® II software starting with version 8.0.

Using CRC error detection for the Stratix IV family has no impact on fitting or performance.

This chapter contains the following sections:

- "Configuration Error Detection" on page 11–2
- "User Mode Error Detection" on page 11–2
- "Error Detection Pin Description" on page 11–5
- "Error Detection Block" on page 11–6
- "Error Detection Timing" on page 11–9
- "Recovering From CRC Errors" on page 11–11
- "Conclusion" on page 11–11

Error Detection Fundamentals

Error detection determines whether the data received is corrupted during transmission. To accomplish this, the transmitter uses a function to calculate a checksum value for the data and appends the checksum to the original data frame. The receiver uses the same calculation methodology to generate a checksum for the received data frame and compares the received checksum to the transmitted checksum. If the two checksum values are equal, the received data frame is correct and no data corruption occurred during transmission or storage.

The error detection CRC feature uses the same concept. When Stratix IV devices have been configured successfully and are in user mode, the error detection CRC feature ensures the integrity of the configuration data.

☞ There are two CRC error checks. One always runs during configuration and a second optional CRC error check that runs in the background in user mode. Both CRC error checks use the same CRC polynomial but different error detection implementations. For more information, refer to "Configuration Error Detection" on page 11–2 and "User Mode Error Detection" on page 11–2.

Configuration Error Detection

In configuration mode, a frame-based CRC is stored within the configuration data and contains the CRC value for each data frame.

During configuration, the Stratix IV device calculates the CRC value based on the frame of data that is received and compares it against the frame CRC value in the data stream. Configuration continues until either the device detects an error or configuration is complete.

In Stratix IV devices, the CRC value is calculated during the configuration stage. A parallel CRC engine generates 16 CRC check bits per frame and then stores them into configuration random access memory (CRAM). The CRAM chain used for storing CRC check bits is 16 bits in width, and its length is equal to the number of frames in the device.

User Mode Error Detection

Stratix IV devices have built-in error detection circuitry to detect data corruption by soft errors in the CRAM cells. This feature allows all CRAM contents to be read and verified to match a configuration-computed CRC value. Soft errors are changes in a CRAM's bit state due to an ionizing particle.

The error detection capability continuously computes the CRC of the configured CRAM bits and compares it with the pre-calculated CRC. If the CRCs match, there is no error in the current configuration CRAM bits. The process of error detection continues until the device is reset (by setting `nCONFIG` low).

After the device transitions into user mode, the error detection process is enabled if you enabled the CRC error detection option in the Quartus II software. The internal 100 MHz configuration oscillator is divided down by a factor of 2 to 256 (at powers of 2) to be used as the clock source during the error detection process. You set the clock divide factor in the option setting in the Quartus II software.

A single 16-bit CRC calculation is done on a per-frame basis. After it has finished the CRC calculation for a frame, the resulting 16-bit signature is `hex 0000` if there are no detected CRAM bit errors in a frame by the error detection circuitry and the output signal `CRC_ERROR` is `0`. If a CRAM bit error is detected by the circuitry within a frame in the device, the resulting signature is non-zero. This causes the CRC engine to start searching the error bit location.

The error detection in Stratix IV devices calculates CRC check bits for each frame and pulls the CRC_ERROR pin high when it detects bit errors in the chip. Within a frame, it can detect all single-bit, double-bit, and three-bit errors. The probability of more than three CRAM bits being flipped by an SEU event is very low. In general, for all error patterns the probability of detection is 99.998%.

The CRC engine reports the bit location and determines the type of error for all single-bit errors and over 99.641% of double-adjacent errors. The probability of other error patterns is very low and report of location of bit flips is not guaranteed by the CRC engine.

You can also read-out the error bit location through the Joint Test Action Group (JTAG) and the core interface. Shift these bits out through either the JTAG instruction, SHIFT_EDERROR_REG, or the core interface, before the CRC detects the next error in another frame. If the next frame also has an error, you have to shift these bits out within the amount of time of one frame CRC verification. You can choose to extend this time interval by slowing down the error detection clock frequency, but this slows down the error recovery time for the SEU event. Refer to Table 11–7 for the minimum update interval for Stratix IV devices. If these bits are not shifted out before the next error location is found, the previous error location and error message is overwritten by the new information. The CRC circuit continues to run, and if an error is detected, you need to decide whether to complete a reconfiguration or to ignore the CRC error.

The error detection logic continues to calculate the CRC_ERROR and 16-bit signatures for the next frame of data regardless if any error has occurred in the current frame or not. You need to monitor these signals and take the appropriate actions if a soft error occurs.

The error detection circuitry in Stratix IV devices uses a 16-bit CRC-ANSI standard (16-bit polynomial) as the CRC generator.

The computed 16-bit CRC signature for each frame is stored in the registers within the core. The total storage register size is 16 (number of bits per frame) × the number of frames.

The Stratix IV device error detection feature does not check memory blocks and I/O buffers. Thus, the CRC_ERROR signal may stay solid high or low depending on the error status of the previously checked CRAM frame. The I/O buffers are not verified during error detection because these bits use flip-flops as storage elements that are more resistant to soft errors compared to CRAM cells. For MLAB, M9K, and M144K memory blocks, they support parity bits that are used to check the contents of memory blocks for any error. The M144K TriMatrix memory block has a built in error correction code block that is able to check and correct the errors in the block.

For more information about error detection in the Stratix IV TriMatrix memory blocks, refer to the *TriMatrix Embedded Memory Blocks in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook.*

To provide testing capability of the error detection block, a JTAG instruction, EDERROR_INJECT, is provided. This instruction is able to change the content of the 21-bit JTAG fault injection register that is used for error injection in Stratix IV devices, enabling the testing of the error detection block.

☞ You can only execute the EDERROR_INJECT JTAG instruction when the device is in user mode.

Table 11–1 shows the description of the `EDERROR_INJECT` JTAG instruction.

Table 11–1. EDERROR_INJECT JTAG Instruction

JTAG Instruction	Instruction Code	Description
`EDERROR_INJECT`	00 0001 0101	This instruction controls the 21-bit JTAG fault injection register, which is used for error injection.

You can create a Jam™ file (**.jam**) to automate the testing and verification process. This allows you to verify the CRC functionality in-system, on-the-fly, without having to reconfigure the device. You can then switch to the CRC circuit to check for real errors induced by an SEU.

You can introduce a single error or double errors adjacent to each other to the configuration memory. This provides an extra way to facilitate design verification and system fault tolerance characterization. Use the JTAG fault injection register with `EDERROR_INJECT` instruction to flip the readback bits. The Stratix IV device is then forced into error test mode.

The content of the JTAG fault injection register is not loaded into the fault injection register during the processing of the last and the first frame. It is only loaded at the end of this period.

☞ You can only introduce error injection in the first data frame, but you can monitor the error information at any time. For more information about the JTAG fault injection register and fault injection register, refer to "Error Detection Registers" on page 11–7.

Table 11–2 shows how the fault injection register is implemented and describes error injection.

Table 11–2. Fault Injection Register

Bit	**Bit[20..19]**			**Bit[18..8]**	**Bit[7..0]**
Description	**Error Type**			**Byte Location of the Injected Error**	**Error Byte Value**
Content	Error Type *(1)*		Error injection type	Depicts the location of the injected error in the first data frame.	Depicts the location of the bit error and corresponds to the error injection type selection.
	Bit[20]	Bit[19]			
	0	1	Single byte error injection		
	1	0	Double-adjacent byte error injection		
	0	0	No error injection		

Note to Table 11–2:

(1) Bit[20] and Bit[19] cannot both be set to 1 as this is not a valid selection. The error detection circuitry decodes this as no error injection.

☞ After the test completes, Altera® recommends that you reconfigure the device.

Automated Single Event Upset Detection

Stratix IV devices offer on-chip circuitry for automated checking of single-event upset detection. Some applications that require the device to operate error-free in high-neutron flux environments require periodic checks to ensure continued data integrity. The error detection CRC feature ensures data reliability and is one of the best options for mitigating SEU.

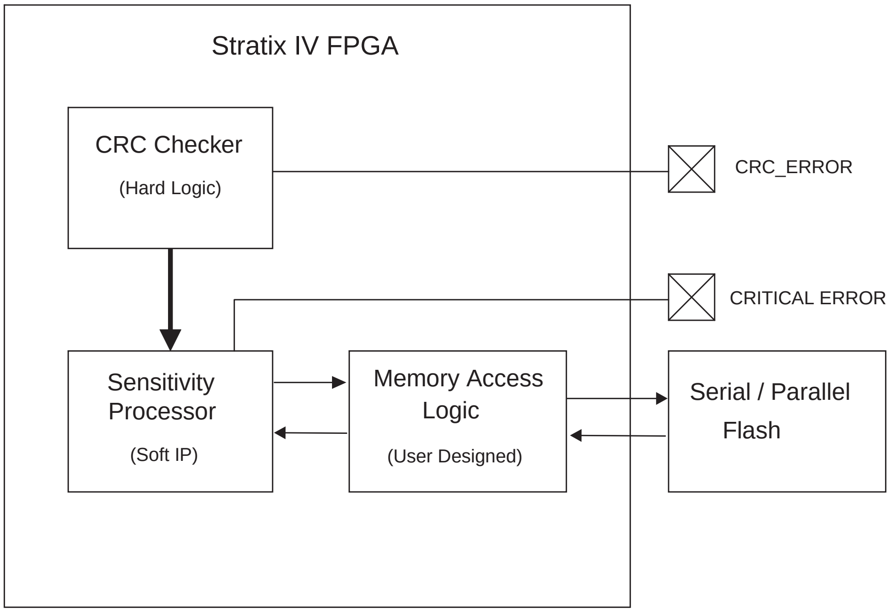
You can implement the error detection CRC feature with existing circuitry in Stratix IV devices, eliminating the need for external logic. The `CRC_ERROR` pin reports a soft error when configuration CRAM data is corrupted, and you would have to decide whether to reconfigure the device or to ignore the error.

Critical Error Detection

After the CRC circuit determines an error, a sensitivity processor determines the criticality of the identified error by accessing the masked configuration bitstream through the user-designed logic and alerts the system for reconfiguration. If it is a non-critical error, the error detection circuitry continues to calculate the `CRC_ERROR` and 16-bit signatures for the next data frame.

This feature uses a sensitivity processor reference design implementing a triple-module redundancy design technique to interface signals between the error detection block and the core IP logic. It implements three copies of the same circuit and performs a bit-wise "majority voting" on the output signals. The chance of three CRAM bits being flipped by an SEU event is very low. Figure 11–1 shows the critical error detection implementation block diagram.

Figure 11–1. Critical Error Detection Implementation Block Diagram



Error Detection Pin Description

Depending on the type of error detection feature you choose, you will need to use different error detection pins to monitor the data during user mode.

CRC_ERROR Pin

Table 11–3 describes the CRC_ERROR pin.

Table 11–3. CRC_ERROR Pin Description

Pin Name	Pin Type	Description
CRC_ERROR	I/O, output open-drain	Active high signal that indicates that the error detection circuit has detected errors in the configuration CRAM bits. This pin is optional and is used when the error detection CRC circuit is enabled. When the error detection CRC circuit is disabled, it is a user I/O pin. The CRC error output, when using the WYSIWYG function, is a dedicated path to the CRC_ERROR pin. To use CRC_ERROR pin, you can either tie this pin to VCCPGM thru a 10 k Ω resistor or, depending on input voltage specification of the system receiving the signal, you can tie this pin to a different pull-up voltage.

☞ The WYSIWYG (What You See Is What You Get) function is an optimization technique which performs optimization on VQM (Verilog Quartus Mapping) netlist within the Quartus II software.

The CRC_ERROR pin information for Stratix IV devices is reported in *Device Pin-Outs* on the Literature page of the Altera website (**www.altera.com**).

CRITICAL ERROR Pin

Table 11–4 describes the CRITICAL ERROR pin.

Table 11–4. CRITICAL ERROR Pin Description

Pin Name	Pin Type	Description
CRITICAL ERROR	I/O, output	Active high signal that indicates that the sensitivity processor reference design has detected errors in the configuration CRAM bits. This pin is optional and is used when the critical error detection is enabled.

☞ The CRITICAL ERROR pin information for Stratix IV devices is included in *Device Pin-Outs* on the Literature page of the Altera website (**www.altera.com**).

Error Detection Block

You can enable the Stratix IV device error detection block in the Quartus II software (refer to "Software Support" on page 11–10). This block contains the logic necessary to calculate the 16-bit CRC signature for the configuration CRAM bits in the device.

The CRC circuit continues running even if an error occurs. When a soft error occurs, the device sets the `CRC_ERROR` pin high. Two types of CRC detection checks the configuration bits:

- CRAM error checking ability (16-bit CRC), which occurs during user mode to be used by the `CRC_ERROR` pin.
  - For each frame of data, the pre-calculated 16-bit CRC enters the CRC circuit at the end of the frame data and determines whether there is an error or not.
  - If an error occurs, the search engine starts to find the location of the error.
  - The error messages can be shifted out through the JTAG instruction or core interface logics while the error detection block continues running.
  - The JTAG interface reads out the 16-bit CRC result for the first frame and also shifts the 16-bit CRC bits to the 16-bit CRC storage registers for test purpose.
  - Single error, double errors, or double errors adjacent to each other can be deliberately introduced to configuration memory for testing and design verification.
- 16-bit CRC which is embedded in every configuration data frame.
  - During configuration, after a frame of data is loaded into the Stratix IV device, the pre-computed CRC is shifted into the CRC circuitry.
  - At the same time, the CRC value for the data frame shifted-in is calculated. If the pre-computed CRC and calculated CRC values do not match, then `nSTATUS` is set low. Every data frame has a 16-bit CRC; therefore, there are many 16-bit CRC values for the whole configuration bitstream. Every device has different lengths of the configuration data frame.

☞ The “Error Detection Block” section focuses on the first type, the 16-bit CRC only when the device is in user mode.

Error Detection Registers

There is one set of 16-bit registers in the error detection circuitry that stores the computed CRC signature. A non-zero value on the syndrome register causes the `CRC_ERROR` pin to be set high. Figure 11–2 shows the block diagram of the error detection circuitry, the syndrome registers, and the error injection block.

Figure 11–2. Error Detection Block Diagram

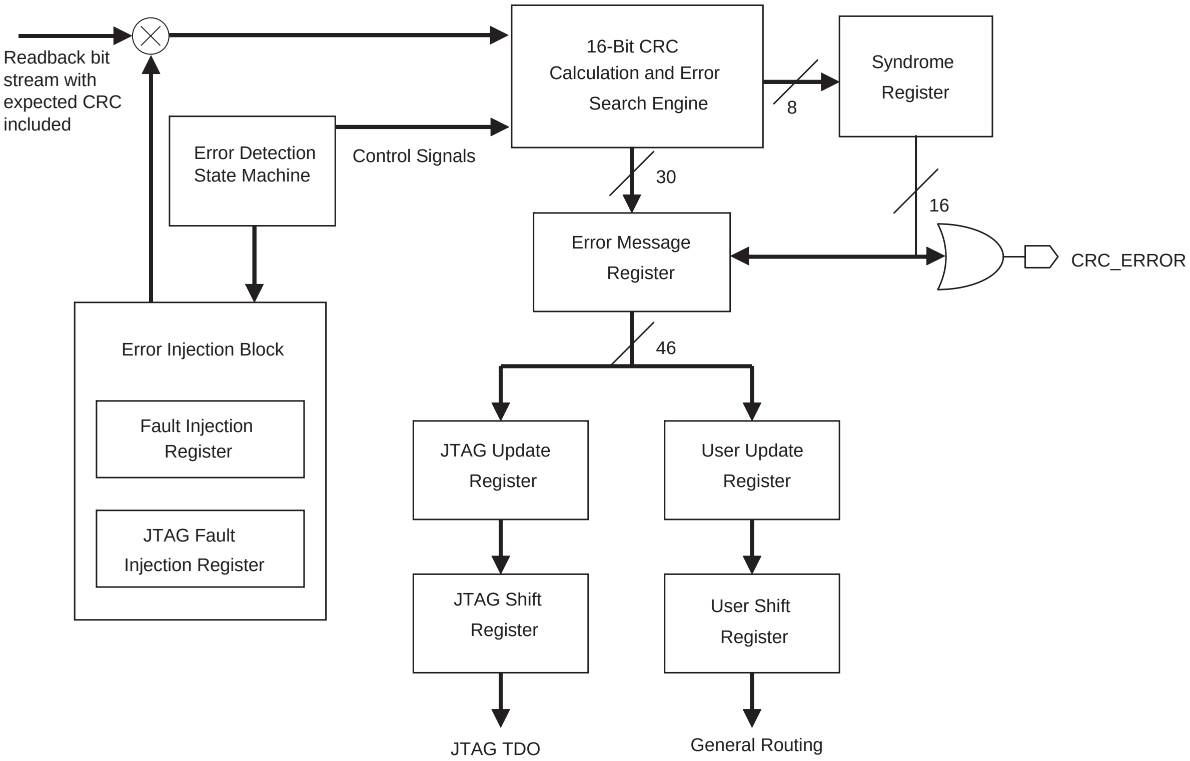


Table 11–5 defines the registers shown in Figure 11–2.

Table 11–5. Error Detection Registers (Part 1 of 2)

Register	Description
Syndrome Register	This register contains the CRC signature of the current frame through the error detection verification cycle. The CRC_ERROR signal is derived from the contents of this register.
Error Message Register	This 46-bit register contains information on the error type, location of the error, and the actual syndrome. The types of errors and location reported are single and double-adjacent bit errors. The location bits for other types of errors are not identified by the Error Message Register. The content of the register can be shifted out through the JTAG SHIFT_EDERROR_REG instruction or to the core through the core interface.
JTAG Update Register	This register is automatically updated with the contents of the Error Message Register one cycle after the 46-bit register content is validated. It includes a clock enable which must be asserted prior to being sampled into the JTAG Shift Register. This requirement ensures that the JTAG Update Register is not being written into by the contents of the Error Message Register at exactly the same time that the JTAG Shift Register is reading its contents.
User Update Register	This register is automatically updated with the contents of the Error Message Register, one cycle after the 46-bit register content is validated. It includes a clock enable which must be asserted prior to being sampled into the User Shift Register. This requirement ensures that the User Update Register is not being written into by the contents of the Error Message Register at exactly the same time that the User Shift Register is reading its contents.

Table 11–5. Error Detection Registers (Part 2 of 2)

Register	Description
JTAG Shift Register	This register is accessible by the JTAG interface and allows the contents of the JTAG Update Register to be sampled and read out by JTAG instruction `SHIFT_EDERROR_REG`.
User Shift Register	This register is accessible by the core logic and allows the contents of the User Update Register to be sampled and read by user logic.
JTAG Fault Injection Register	This 21-bit register is fully controlled by the JTAG instruction `EDERROR_INJECT`. This register holds the information of the error injection that you want in the bitstream.
Fault Injection Register	The content of the JTAG Fault Injection Register is loaded into this 21-bit register when it is being updated.

Error Detection Timing

When the CRC feature is enabled through the Quartus II software, the device automatically activates the CRC process upon entering user mode, after configuration, and after initialization is complete.

If an error is detected within a frame, CRC_ERROR is driven high at the end of error location search, after the Error Message Register is updated. At the end of this cycle, the CRC_ERROR pin is pulled low for minimum of 32 clock cycles. If the next frame contains an error, the CRC_ERROR is driven high again after the Error Message Register is overwritten by the new value. You can start to unload the error message on each rising edge of the CRC_ERROR pin. The error detection runs until the device is reset.

The error detection circuitry runs off an internal configuration oscillator with a divisor that sets the maximum frequency. Table 11–6 shows the minimum and maximum error detection frequencies.

Table 11–6. Minimum and Maximum Error Detection Frequencies

Device Type	Error Detection Frequency	Maximum Error Detection Frequency	Minimum Error Detection Frequency	Valid Divisors (n)
Stratix IV	100 MHz / 2n	50 MHz	390 kHz	1, 2, 3, 4, 5, 6, 7, 8

You can set a lower clock frequency by specifying a division factor in the Quartus II software (refer to "Software Support" on page 11–10). The divisor is a power of two (2), where *n* is between 1 and 8. The divisor ranges from 2 through 256. See the following equation:

Equation 11–1.

$$\text{error detection frequency} = \frac{100\text{MHz}}{2^n}$$

You need to monitor the error message to avoid missing information in the Error Message Register. The Error Message Register is updated whenever an error or errors occur. The minimum interval time between each update for the Error Message Register depends on the device and the error detection clock frequency. Table 11–7 shows the estimated minimum interval time between each update for the Error Message Register for Stratix IV devices.

Table 11–7. Minimum Update Interval for Error Message Register *(Note 1)*

Device	Timing Interval (µs)
EP4SGX70	15
EP4SGX110	15
EP4SGX230	21
EP4SGX290	23
EP4SGX360	23
EP4SGX530	28
EP4SE110	15
EP4SE230	21
EP4SE290	23
EP4SE360	23
EP4SE530	28
EP4SE680	30

Note to Table 11–7:

(1) These timing numbers are preliminary.

Software Support

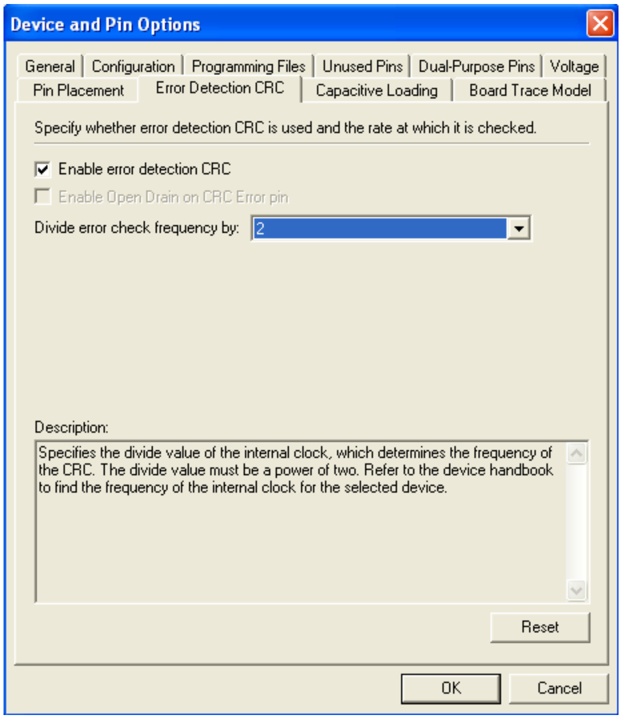
The Quartus II software, starting with version 8.0, supports the error detection CRC feature for Stratix IV devices. Enabling this feature generates the `CRC_ERROR` output to the optional dual purpose `CRC_ERROR` pin.

The error detection CRC feature is controlled by the **Device and Pin Options** dialog box in the Quartus II software.

Enable the error detection feature using CRC by performing the following steps:

1. Open the Quartus II software and load a project using a Stratix IV device.
2. On the Assignments menu, click **Settings**. The **Settings** dialog box is shown.
3. In the **Category** list, select **Device**. The **Device** page is shown.
4. Click **Device and Pin Options**. The **Device and Pin Options** dialog box is shown (see Figure 11–3).
5. In the **Device and Pin Options** dialog box, click the **Error Detection CRC** tab.
6. Turn on **Enable error detection CRC (**Figure 11–3**)**.

Figure 11–3. Enabling the Error Detection CRC Feature in the Quartus II Software



Specify whether error detection CRC is used and the rate at which it is checked.

☑ Enable error detection CRC

□ Enable Open Drain on CRC Error pin

Divide error check frequency by: 2

Description:

Specifies the divide value of the internal clock, which determines the frequency of the CRC. The divide value must be a power of two. Refer to the device handbook to find the frequency of the internal clock for the selected device.

7. In the **Divide error check frequency by** box, enter a valid divisor as listed in Table 11–6.

☞ The divide value divides the frequency of the configuration oscillator output clock that clocks the CRC circuitry.

8. Click **OK**.

Recovering From CRC Errors

The system that the Stratix IV device resides in must control the device reconfiguration. After detecting an error on the `CRC_ERROR` pin, strobing the `nCONFIG` signal low directs the system to perform the reconfiguration at a time when it is safe for the system to reconfigure the device.

When the data bit is rewritten with the correct value by reconfiguring the device, the device functions correctly.

While soft errors are uncommon in Altera devices, certain high-reliability applications may require a design to account for these errors.

Conclusion

The purpose of the error detection CRC feature is to detect a flip in any of the configuration CRAM bits in Stratix IV devices due to a soft error. By using the error detection circuitry, you can continuously verify the integrity of the configuration CRAM bits.

Referenced Documents

This chapter references the following documents:

- *TriMatrix Embedded Memory Blocks in Stratix IV Devices*

Document Revision History

Table 11–8 shows the revision history for this document.

Table 11–8. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	Minor text edits.	—
May 2008 v1.0	Initial Release.	—

12. JTAG Boundary-Scan Testing in Stratix IV Devices

SIV51012-2.0

Introduction

This chapter discusses how to use the IEEE Std. 1149.1 boundary-scan test (BST) circuitry specifically in Stratix® IV devices. The features are similar to Stratix III devices, unless stated in this document.

This chapter contains the following sections:

- "BST Architecture"
- "BST Operation Control"
- "I/O Voltage Support in a JTAG Chain"
- "BST Circuitry"
- "Boundary-Scan Description Language (BSDL) Support"

BST Architecture

A device operating in IEEE Std. 1149.1 BST mode uses four required pins, `TDI`, `TDO`, `TMS`, and `TCK`, and one optional pin, `TRST`. The `TCK` pin has an internal weak pull-down resistor, while the `TDI`, `TMS`, and `TRST` pins have internal weak pull-up resistors. The `TDO` output pin and all the JTAG input pins are powered by the 2.5-V/3.0-V V_{CCPD} supply of I/O Bank 1A. All user I/O pins are tri-stated during JTAG configuration.

For more information about the description and functionality of all JTAG pins, registers used by the IEEE Std. 1149.1 BST circuitry, and the test access port (TAP) controller, refer to the *IEEE 1149.1 (JTAG) Boundary-Scan Testing in Stratix III Devices* chapter in volume 1 of the *Stratix III Device Handbook*.

BST Operation Control

Table 12–1 shows the boundary-scan register length for Stratix IV devices.

Table 12–1. Stratix IV Boundary-Scan Register Length (Part 1 of 2)

Device	Boundary-Scan Register Length
EP4SGX70	1506
EP4SGX110	1506
EP4SGX230	2274
EP4SGX290	2682
EP4SGX360	2682
EP4SGX530	2970
EP4SE110	1506
EP4SE230	2274
EP4SE290	2682

Table 12–1. Stratix IV Boundary-Scan Register Length (Part 2 of 2)

Device	Boundary-Scan Register Length
EP4SE360	2682
EP4SE530	2970
EP4SE680	TBD

Table 12–2 shows the `IDCODE` information for Stratix IV devices.

Table 12–2. Stratix IV Device IDCODE

	IDCODE (32 Bits) *(1)*			
Device	**Version (4 Bits)**	**Part Number (16 Bits)**	**Manufacturer Identity (11 Bits)**	**LSB (1 Bit)** *(2)*
EP4SGX70	0000	0010 0100 0010 0000	000 0110 1110	1
EP4SGX110	0000	0010 0100 0000 0000	000 0110 1110	1
EP4SGX230	0000	0010 0100 0000 0001	000 0110 1110	1
EP4SGX290	0000	0010 0100 0010 0010	000 0110 1110	1
EP4SGX360	0000	0010 0100 0000 0010	000 0110 1110	1
EP4SGX530	0000	0010 0100 0000 0011	000 0110 1110	1
EP4SE110	0000	0010 0100 0001 0000	000 0110 1110	1
EP4SE230	0000	0010 0100 0001 0001	000 0110 1110	1
EP4SE290	0000	0010 0100 0100 0010	000 0110 1110	1
EP4SE360	0000	0010 0100 0001 0010	000 0110 1110	1
EP4SE530	0000	0010 0100 0001 0011	000 0110 1110	1
EP4SE680	0000	0010 0100 0000 0100	000 0110 1110	1

Notes to Table 12–2:

(1) The most significant bit (MSB) is on the left.

(2) The least significant bit (LSB) of `IDCODE` is always 1.

To read the IDCODE correctly, you should issue the IDCODE instruction after initialization, which is signaled by NSTATUS going high.

Refer to the *IEEE 1149.1 (JTAG) Boundary-Scan Testing in Stratix III Devices* chapter in volume 1 of the *Stratix III Device Handbook* for more information about the following topics:

- JTAG instruction codes with descriptions
- TAP controller state-machine
- Timing requirements for IEEE Std. 1149.1 signals
- Instruction mode
- Mandatory JTAG instructions (`SAMPLE/PRELOAD`, `EXTEST` and `BYPASS`)
- Optional JTAG instructions (`IDCODE`, `USERCODE`, `CLAMP` and `HIGHZ`)

I/O Voltage Support in a JTAG Chain

The JTAG chain supports several devices. However, you should use caution if the chain contains devices that have different V_{CCIO} levels.

For more information about the I/O voltage support in the JTAG chain, refer to the *IEEE 1149.1 (JTAG) Boundary-Scan Testing in Stratix III Devices* chapter in volume 1 of the *Stratix III Device Handbook*.

BST Circuitry

The IEEE Std. 1149.1 BST circuitry is enabled upon device power-up. You can perform BST on Stratix IV FPGAs before, during, and after configuration. Stratix IV FPGAs support the `BYPASS`, `IDCODE`, and `SAMPLE` instructions during configuration without interrupting configuration. To send all other JTAG instructions, you must interrupt configuration using the `CONFIG_IO` instruction.

For more information on JTAG or boundary-scan testing, refer to *AN 39: IEEE Std. 1149.1 (JTAG) Boundary-Scan Testing in Altera Devices*.

For more information about using the `CONFIG_IO` JTAG instruction for dynamic I/O buffer configuration, considerations when performing BST for configured devices, and JTAG pin connections to mask-out the BST circuitry, refer to the *IEEE 1149.1 (JTAG) Boundary-Scan Testing in Stratix III Devices* chapter in volume 1 of the *Stratix III Device Handbook*.

For information about using the IEEE Std.1149.1 circuitry for device configuration, refer to the *Configuration, Design Security, Remote System Upgrades* chapter in volume 1 of the *Stratix IV Device Handbook*.

☞ If you must perform BST for configured devices, you must use the Quartus II software version 8.1 or later to generate the design-specific BSDL files. Visit the Altera web site at **www.altera.com** for the procedure to generate post-configured BSDL files using the Quartus II software.

Boundary-Scan Description Language (BSDL) Support

The boundary-scan description language (BSDL), a subset of VHDL, provides a syntax that allows you to describe the features of an IEEE Std. 1149.1 BST-capable device that can be tested.

For more information about BSDL files for IEEE Std. 1149.1-compliant Stratix IV devices, visit the Altera® web site at **www.altera.com**.

BSDL files for IEEE std. 1149.1-compliant Stratix IV devices can also be generated using the Quartus II software version 8.0 or later. Visit the Altera web site at **www.altera.com** for the procedure to generate BSDL files using the Quartus II software.

Conclusion

The IEEE Std. 1149.1 BST circuitry available in Stratix IV devices provides a cost-effective and efficient way to test systems that contain devices with tight lead spacing. Circuit boards with Altera and other IEEE Std. 1149.1-compliant devices can use the EXTEST, SAMPLE/PRELOAD, and BYPASS modes to create serial patterns that internally test the pin connections between devices and check device operation.

Referenced Documents

This chapter references the following documents:

- *Configuration, Design Security, Remote System Upgrades* chapter in volume 1 of the *Stratix IV Device Handbook*
- *IEEE 1149.1 (JTAG) Boundary-Scan Testing in Stratix III Devices* chapter in volume 1 of the *Stratix III Device Handbook*
- *AN 39: IEEE Std. 1149.1 (JTAG) Boundary-Scan Testing in Altera Devices*
- *DC and Switching Characteristics* chapter in volume 2 of the *Stratix IV Device Handbook*

Revision History

Table 12–3 shows the revision history for this document.

Table 12–3. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	Minor text edits.	—
May 2008 v1.0	Initial release.	—

13. Power Management in Stratix IV Devices

SIV51013-2.0

Introduction

Altera® Stratix® IV FPGAs deliver a breakthrough level of system bandwidth and power efficiency for high-end applications, allowing you to innovate without compromise. Advanced power management techniques are used in Stratix IV devices to enable both density and performance increases, while simultaneously reducing power dissipation.

The total power of an FPGA includes static power and dynamic power. Static power is the power consumed by the FPGA when it is configured, but no clocks are operating. Dynamic power is comprised of switching power when the device is configured and running. Dynamic power is calculated using the equation shown in Equation 13–1.

Equation 13–1. Dynamic Power Equation

$$P = \frac{1}{2}CV^2 \times \text{frequency}$$

Equation 13–1 shows that frequency and toggle rate are design-dependent. However, voltage can be varied to lower dynamic power consumption by the square value of the voltage difference. Stratix IV devices minimize static and dynamic power with advanced process optimizations and programmable power technology. These technologies enable Stratix IV designs to optimally meet design-specific performance requirements with the lowest possible power.

The Quartus® II software optimizes all designs with Stratix IV power technology to ensure performance is met at the lowest power consumption. This automatic process allows you to concentrate on the functionality of the design instead of the power consumption of the design.

Power consumption also affects thermal management. Stratix IV devices offer a temperature sensing diode (TSD), which can self-monitor the device junction temperature and can be used with external circuitry for activities such as controlling air flow to the Stratix IV FPGA.

This chapter contains the following sections:

- "Introduction"
- "Stratix IV Power Technology"
- "Stratix IV External Power Supply Requirements"
- "Temperature Sensing Diode"

Stratix IV Power Technology

The following sections provide details about Stratix IV programmable power technology.

Programmable Power Technology

Stratix IV devices offer the ability to configure portions of the core, called tiles, for high-speed or low-power mode of operation performed by the Quartus II software without user intervention. Setting a tile to high-speed or low-power mode is accomplished with on-chip circuitry and does not require any extra power supply brought into the Stratix IV device. In a design compilation, the Quartus II software determines whether a tile must be in high-speed or low-power mode based on the timing constraints of the design.

For more information about how the Quartus II software uses programmable power technology when compiling a design, refer to *AN 514: Power Optimization in Stratix IV Devices*.

A Stratix IV tile can consist of the following:

- MLAB/logic array block (LAB) pairs with routing to the pair
- MLAB/LAB pairs with routing to the pair and to adjacent digital signal processing (DSP)/memory block routing
- TriMatrix memory blocks
- DSP blocks

All blocks and routing associated with the tile share the same setting of either high speed or low power. By default, tiles that include DSP blocks, memory blocks, or I/O interfaces are set to high-speed mode for optimum performance when used in the design. Unused DSP blocks, memory blocks, and I/O elements are set to low-power mode to minimize static power. Clock networks do not support programmable power technology.

With programmable power technology, faster speed grade FPGAs may require less power because there are fewer high-speed MLAB and LAB pairs, compared to slower speed grade FPGAs. The slower speed grade device may need to use more high-speed MLAB and LAB pairs to meet performance requirements, while the faster speed grade device can meet performance requirements with MLAB and LAB pairs in low-power mode.

The Quartus II software can set unshared inputs and unused device resources in the design to low-power mode to reduce static and dynamic power. The Quartus II software can set the following resources to low power when they are not used in the design:

- LABs and MLABs
- TriMatrix memory blocks
- DSP blocks

If the PLL is instantiated in the design, asserting a reset high keeps the PLL in low power mode.

Table 13–1 shows the available Stratix IV programmable power capabilities. Speed grade considerations can add to the permutations to give you flexibility in designing your system.

Table 13–1. Stratix IV Programmable Power Capabilities

Feature	Programmable Power Technology
LAB	Yes
Routing	Yes
Memory Blocks	Fixed setting *(1)*
DSP Blocks	Fixed setting *(1)*
Global Clock Networks	No

Note to Table 13–1:

(1) Tiles with DSP blocks and memory blocks that are used in the design are always set to high-speed mode. By default, unused DSP blocks, memory blocks, and I/O interfaces are set to low-power mode.

Stratix IV External Power Supply Requirements

This section describes the different external power supplies needed to power Stratix IV devices. Table 13–2 lists the external power supply pins for Stratix IV devices. You can supply some of the power supply pins with the same external power supply, provided they need the same voltage level.

Refer to the Pin Connections Guidelines for recommendations on sharing external power supplies.

☞ For each power supply input value, refer to the *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook*.

Table 13–2. Stratix IV External Power Supply Pins (Part 1 of 2)

Power Supply Pin	Power Supply Level (V)	Description
VCC	0.9	Core voltage and periphery circuitry power supply
VCCD_PLL	0.9	PLL digital power supply
VCCA_PLL	2.5	PLL analog power supply
VCCAUX	2.5	Auxiliary supply for the programmable power technology
VCCPT	1.5	Power supply for programmable power technology
VCCPGM	1.8/2.5/3.0	Configuration pins power supply
VCCPD	2.5/3.0	I/O pre-driver power supply
VCCIO	1.2/1.5/1.8/2.5/3.0	I/O power supply
VCC_CLKIN	2.5	Differential clock input pins power supply (top and bottom I/O banks only)
VCCBAT	3.0	Battery back-up power supply for design security volatile key register
VREF	VCCIO / 2	Voltage-referenced I/O standards power supply
GND	GND	Ground
VCCHIP_L	0.9	Transceiver HIP digital power (left side)
VCCHIP_R	0.9	Transceiver HIP digital power (right side)
VCCT_L	1.1	Transmitter power (left side)

Table 13–2. Stratix IV External Power Supply Pins (Part 2 of 2)

Power Supply Pin	Power Supply Level (V)	Description
VCCT_R	1.1	Transmitter power (right side)
VCCR_L	1.1	Receiver power (left side)
VCCR_R	1.1	Receiver power (right side)
VCCA_L	2.5/3.0	Transceiver high voltage power (left side)
VCCA_R	2.5/3.0	Transceiver high voltage power (right side)
VCCH_GXBL[#] *(1)*	1.4/1.5	Transceiver output buffer power for transceiver block # (left side)
VCCH_GXBR[#] *(1)*	1.4/1.5	Transceiver output buffer power for transceiver block # (right side)
VCCL_GXBL[#] *(1)*	1.1	Transceiver clock power for transceiver block # (left side)
VCCL_GXBR[#] *(1)*	1.1	Transceiver clock power for transceiver block # (right side)

Note to Table 13–2:

(1) The VCCH and VCCL powers are per transceiver blocks.

Temperature Sensing Diode

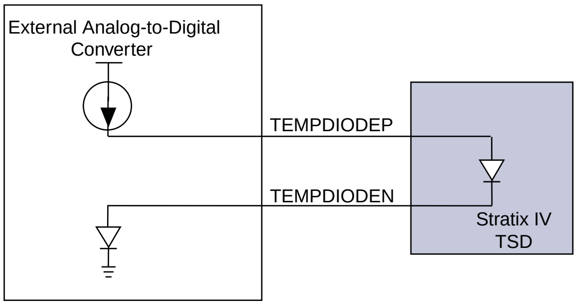
The Stratix IV temperature sensing diode (TSD) uses the characteristics of a PN junction diode to determine die temperature. Knowing the junction temperature is crucial for thermal management. Historically, junction temperature is calculated using ambient or case temperature, junction-to-ambient (ja) or junction to-case (jc) thermal resistance, and device power consumption. Stratix IV devices can monitor its die temperature with an embedded TSD, so you can control the air flow to the device.

The TSD of Stratix IV devices are used with an external digital thermometer device. These devices steer bias current through the Stratix IV TSD, measuring forward voltage and converting this reading to temperature in the form of an 8-bit signed number (7 bits plus sign). The external device's output represents the junction temperature of the Stratix IV device and can be used for intelligent power management.

External Pin Connections

The Stratix IV TSD, located in the top right corner of the die, requires two pins for voltage reference. You can connect the TSD with either an external or embedded analog-to-digital converter in the devices.

Figure 13–1. Stratix IV TSD External Pin Connections



The temperature sensing diode is a very sensitive circuit which can be influenced by noise coupled from other traces on the board and possibly within the device package itself, depending on device usage. The interfacing device registers temperature based on millivolts (mV) of difference, as seen at the TSD. Switching I/O near the TSD pins can affect the temperature reading. Altera recommends you take temperature readings during periods of no activity in the device.

Conclusion

As process geometries get smaller, power and thermal management becomes more crucial in FPGA designs. Stratix IV devices offer programmable power technology options for low-power operation. You can use these features, along with speed grade choices, in different permutations to give the best power and performance combination. Taking advantage of the silicon, the Quartus II software is able to manipulate designs to use the best combination to achieve lowest power at the required performance.

For thermal management, use the Stratix IV temperature sensing diode with either an external or embedded analog-to-digital converter in production devices to easily incorporate this feature in your designs. Being able to monitor the junction temperature of the device at any time also allows you the ability to control air flow to the device and save power for the whole system.

Referenced Document

This chapter references the following document:

- AN 514: Power Optimization in Stratix IV Devices
- *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook*

Revision History

Table 13–3 shows the revision history for this document.

Table 13–3. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	Minor text edits.	—
May 2008 v1.0	Initial release.	—

Stratix IV Device Handbook, Volume 2

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

SIV5V2-2.0

Contents

Chapter Revision Dates ix

Additional Information

About this Handbook . . . . Info-xi
How to Contact Altera . . . . Info-xi
Typographic Conventions . . . . Info-xi

Section I. Transceiver Architecture

Revision History I-1

Chapter 1. Stratix IV Transceiver Architecture

Introduction 1-1
Transceiver Channel Locations 1-2
Transceiver Block Architecture 1-5
Transceiver Port List 1-6
CMU Channels 1-20
  CMU0 Channel 1-21
    CMU0 PLL 1-22
    Power Down CMU0 PLL 1-24
    CMU0 Clock Divider Block 1-24
  CMU1 Channel 1-26
    Power Down CMU1 PLL 1-26
Transceiver Channel Architecture 1-27
Transmitter Channel Datapath 1-29
  TX Phase Compensation FIFO 1-29
    Input Data 1-30
    Output Data Destination Block 1-30
    TX Phase Compensation FIFO Status Signal 1-31
  Byte Serializer 1-31
    Single-Width Mode 1-31
    Double-Width Mode 1-32
  8B/10B Encoder 1-33
    Single-Width Mode 1-33
    Double-Width Mode 1-35
    Controlling Running Disparity 1-38
    Transmitter Polarity Inversion 1-40
    Transmitter Bit Reversal 1-42
  Serializer 1-44
  Transmitter Output Buffer 1-45
    Programmable Transmitter Termination 1-46
    Programmable Output Differential Voltage 1-46
    Programmable Pre-Emphasis 1-47
    Programmable Transmitter Output Buffer Power (VCCH) 1-48
    Common Mode Voltage (VCM) Settings 1-48
    PCI Express (PIPE) Receiver Detect 1-48
    PCI Express Electrical Idle 1-48
Transmitter Local Clock Divider Block 1-49

Receiver Channel Datapath 1-49
Receiver Input Buffer 1-50
Programmable Differential On-Chip Termination 1-51
Programmable Common Mode Voltage 1-51
Link Coupling 1-52
Programmable Equalization and DC Gain 1-57
Signal Threshold Detection Circuitry 1-58
Clock and Data Recovery Unit 1-58
Lock-to-Reference Mode 1-59
Lock-to-Data Mode 1-61
PCI Express (PIPE) Clock Switch Circuitry 1-62
LTR/LTD Controller 1-62
Deserializer 1-63
Word Aligner 1-64
Word Aligner in Single-Width Mode 1-65
Word Aligner in Double-Width Mode 1-71
Programmable Run Length Violation Detection 1-76
Receiver Polarity Inversion 1-77
Receiver Bit Reversal 1-79
Receiver Byte Reversal in Basic Double-Width Modes 1-81
Deskew FIFO 1-82
Rate Match (Clock Rate Compensation) FIFO 1-84
Rate Match FIFO in PCI Express (PIPE) Mode 1-85
Rate Match FIFO in XAUI Mode 1-87
Rate Match FIFO in GIGE Mode 1-89
Rate Match FIFO in Basic Single-Width Mode 1-90
Rate Match FIFO in Basic Double-Width Mode 1-92
8B/10B Decoder 1-95
8B/10B Decoder in Single-Width Mode 1-95
8B/10B Decoder in Double-Width Mode 1-97
Byte Deserializer 1-99
Byte Deserializer in Single-Width Mode 1-100
Byte Deserializer in Double-Width Mode 1-100
Byte Ordering Block 1-101
Byte Ordering Block in Single-Width Modes 1-102
Byte Ordering Block in Double-Width Modes 1-102
Word-Alignment-Based Byte Ordering 1-104
User-Controlled Byte Ordering 1-104
Receiver Phase Compensation FIFO 1-105
Receiver Phase Compensation FIFO Error Flag 1-106
Offset Cancellation in the Receiver Buffer and Receiver CDR 1-107
Offset Cancellation 1-107
Offset Cancellation Process 1-108

Functional Modes 1-108
Basic Functional Mode 1-109
Low Latency PCS Datapath 1-109
Basic Single-Width Mode Configurations 1-110
Basic Double-Width Mode Configurations 1-112
PCI Express (PIPE) Mode 1-114
PCI Express (PIPE) Mode Configurations 1-115
PCI Express (PIPE) Mode Datapath 1-116
PCI Express (PIPE) Interface 1-119
Fast Recovery Mode 1-124
Electrical Idle Inference 1-125
PCI Express Gen2 (5 Gbps) Support 1-126
PCI Express Cold Reset Requirements 1-137
XAUI Mode 1-139
XAUI Mode Datapath 1-143
XGMII-To-PCS Code Conversion at the Transmitter 1-143
PCS-To-XGMII Code Conversion at the Receiver 1-145
Word Aligner 1-145
Deskew FIFO 1-147
Rate Match FIFO 1-148
GIGE Mode 1-150
GIGE Mode Datapath 1-152
8B/10B Encoder 1-153
Word Aligner 1-155
Rate Match FIFO 1-157
SONET/SDH Mode 1-158
SONET/SDH Frame Structure 1-158
SONET/SDH OC-12 Datapath 1-159
SONET/SDH OC-48 Datapath 1-160
SONET/SDH OC-96 Datapath 1-160
SONET/SDH Transmission Bit Order 1-161
Word Alignment 1-161
OC-48 and OC-96 Byte Serializer and Deserializer 1-162
OC-48 Byte Ordering 1-162
SDI Mode 1-163
SDI Mode Datapath 1-164
(OIF) CEI PHY Interface Mode 1-165
(OIF) CEI PHY Interface Mode Datapath 1-167
(OIF) CEI PHY Interface Mode Clocking 1-167
Serial RapidIO Mode 1-169
Synchronization State Machine 1-170
Loopback Modes 1-171
Serial Loopback 1-172
Reverse Serial Loopback 1-172
Reverse Serial Pre-CDR Loopback 1-173
PCI Express (PIPE) Reverse Parallel Loopback 1-174
Calibration Blocks 1-175
Calibration Block Location 1-175
Calibration 1-177
Input Signals to the Calibration Block 1-177
Referenced Documents 1-178
Document Revision History 1-178

Chapter 2. Stratix IV Transceiver Clocking

Introduction 2-1
CMU PLL and Receiver CDR Input Reference Clocking 2-1
Input Reference Clock Source 2-1
refclk0 and refclk1 Pins 2-3
Inter-Transceiver Block (ITB) Clock Lines 2-5
Dedicated CLK Input Pins on the FPGA Global Clock Network 2-6
Clock Output from Left and Right PLLs in the FPGA Fabric 2-7
Transceiver Channel Datapath Clocking 2-7
Transmitter Channel Datapath Clocking 2-7
Non-Bonded Channel Configurations 2-8
Bonded Channel Configurations 2-10
Receiver Channel Datapath Clocking 2-18
Non-Bonded Channel Configurations 2-18
Bonded Channel Configurations 2-22
FPGA Fabric-Transceiver Interface Clocking 2-28
Input Reference Clocks 2-28
Phase Compensation FIFO Clocks 2-29
Other Transceiver Clocks 2-29
FPGA Fabric-Transmitter Interface Clocking 2-30
Quartus II-Selected Transmitter Phase Compensation FIFO Write Clock 2-30
User-Selected Transmitter Phase Compensation FIFO Write Clock 2-38
FPGA Fabric-Receiver Interface Clocking 2-43
Quartus II Software-Selected Receiver Phase Compensation FIFO Read Clock 2-43
User-Selected Receiver Phase Compensation FIFO Read Clock 2-52
FPGA Fabric PLLs-Transceiver PLLs Cascading 2-57
Dedicated Left and Right PLL Cascade Lines Network 2-58
FPGA Fabric PLLs-Transceiver PLLs Cascading in the 780-Pin Package 2-58
FPGA Fabric PLLs-Transceiver PLLs Cascading in the 1152-Pin Package 2-58
FPGA Fabric PLLs-Transceiver PLLs Cascading in the 1517-Pin Package 2-62
FPGA Fabric PLLs-Transceiver PLLs Cascading in the 1932-Pin Package 2-64
FPGA Fabric PLLs-Transceiver PLLs Cascading Rules 2-65
Referenced Documents 2-68
Document Revision History 2-69

Chapter 3. Configuring Multiple Protocols and Data Rates in a Transceiver Block

Overview 3-1
Creating Transceiver Channel Instances 3-1
General Requirements to Combine Channels 3-2
Transmitter Buffer Voltage (V_{CCH}) 3-2
Transceiver Analog Power ($V_{CCA\_L/R}$) 3-2
Control Signals 3-3
gxb_powerdown Port 3-3
reconfig_fromgxb and reconfig_togxb Ports 3-3
Calibration Clock and Power Down 3-3
Sharing CMU PLLs 3-4
Multiple Channels Sharing a CMU PLL 3-4
Example 1 3-4
Multiple Channels Sharing Two CMU PLLs 3-8
Example 2 3-8
Combining Receiver Only Channels 3-12
Combining Transmitter Channel and Receiver Channel Instances 3-12
Multiple Transmitter Channel and Receiver Channel Instances 3-12
Example 3 3-13

Combining Channels Configured in Protocol Functional Modes . . . 3-15
Basic x4 Mode . . . 3-15
Basic ×8 Mode . . . 3-16
(OIF) CEI Phy Interface Mode . . . 3-17
Combining Channels Using the PCI Express Hard IP Block with Other Channels . . . 3-18
Combining Transceiver Instances Using PLL Cascade Clocks . . . 3-19
Combining Transceiver Instances in Multiple Transceiver Blocks . . . 3-19
Example 1 . . . 3-20
Summary . . . 3-22
Referenced Documents . . . 3-23
Document Revision History . . . 3-23

Chapter 4. Reset Control and Power Down

Introduction . . . 4-1
User Reset and Power-Down Signals . . . 4-1
Blocks Affected by Reset and Power-Down Signals . . . 4-3
Transceiver Reset Sequences . . . 4-4
All Supported Functional Modes Except the PCI Express (PIPE) Functional Mode . . . 4-5
Bonded Channel Configuration . . . 4-6
Receiver and Transmitter Channel—Receiver CDR in Automatic Lock Mode . . . 4-7
Receiver and Transmitter Channel—Receiver CDR in Manual Lock Mode . . . 4-9
Receiver and Transmitter Channel—Receiver CDR in Automatic Lock Mode . . . 4-10
Receiver and Transmitter Channel—Receiver CDR in Manual Lock Mode . . . 4-12
Receiver Only Channel—Receiver CDR in Automatic Lock Mode . . . 4-13
Receiver Only Channel—Receiver CDR in Manual Lock Mode . . . 4-14
Receiver and Transmitter Channel—Receiver CDR in Automatic Lock Mode . . . 4-15
Receiver and Transmitter Channel—Receiver CDR in Manual Lock Mode . . . 4-17
PCI Express (PIPE) Functional mode . . . 4-18
PCI Express (PIPE) Reset Sequence . . . 4-18
PCI Express (PIPE) Initialization/Compliance Phase . . . 4-19
PCI Express Normal Phase . . . 4-20
Power Down . . . 4-21
Simulation Requirements . . . 4-22
Documents Referenced . . . 4-22
Document Revision History . . . 4-22

Chapter 5. Stratix IV Dynamic Reconfiguration

Introduction . . . 5-1
Dynamic Reconfiguration Modes . . . 5-1
PMA Controls Reconfiguration . . . 5-1
Offset Cancellation . . . 5-1
Conventions Used in this Chapter . . . 5-2

Quartus II MegaWizard Plug-In Manager Interfaces to Support Dynamic Reconfiguration 5-2
ALTGX MegaWizard Plug-In Manager 5-2
ALTGX_RECONFIG MegaWizard Plug-In Manager 5-3
Dynamic Reconfiguration Controller Architecture 5-4
Dynamic Reconfiguration Controller Interface 5-5
Dynamic Reconfiguration Controller Port List 5-7
Clock Requirements for the ALTGX instance and ALTGX_RECONFIG instance 5-12
Clock Requirements for ALTGX Instance 5-12
Clock Requirements for ALTGX_RECONFIG Instance 5-12
Interfacing ALTGX_RECONFIG Instance and ALTGX Instances 5-13
Logical Channel Addressing 5-14
Total Number of Channels Controlled by the ALTGX_RECONFIG Instance 5-25
Connecting reconfig_from_gxb/reconfig_to_gxb Ports 5-28
Offset Cancellation Control for Receiver Channels 5-31
Operation 5-31
Example for the Offset Cancellation Process of a Receiver Channel 5-34
PMA Controls Reconfiguration 5-37
Dynamic Reconfiguration Controller Ports for PMA controls 5-37
Dynamically Reconfiguring PMA Controls 5-38
Method 1 5-39
Method 2 5-43
Examples: Dynamic Reconfiguration Controller (ALTGX_RECONFIG) 5-48
Example 1: One Reconfiguration Controller Connected to Multiple ALTGX Instances 5-48
Example 2: Two ALTGX_RECONFIG Instances Connected to Two ALTGX Instances 5-51
Example 3: One ALTGX_RECONFIG Instance Connected to an ALTGX Instance Stamped Five Times 5-54
Error Indication in the ALTGX_RECONFIG MegaWizard Plug-In Manager 5-56
Combining Transceiver Channels with Dynamic Reconfiguration Enabled 5-56
Requirements 5-57
Dynamic Reconfiguration Duration and FPGA-Fabric Resource Utilization 5-57
Dynamic Reconfiguration Duration 5-57
PMA Controls Reconfiguration Duration 5-58
Offset Cancellation Duration 5-59
Dynamic Reconfiguration (ALTGX_RECONFIG Instance) Resource Utilization 5-60
Functional Simulation of the Offset Cancellation Process 5-60
Document Revision History 5-61

Chapter Revision Dates

The chapters in this book, *Stratix IV Device Handbook, Volume 2*, were revised on the following dates. Where chapters or groups of chapters are available separately, part numbers are listed.

Chapter 1 Stratix IV Transceiver Architecture
Revised: *November 2008*
Part Number: *SIV52001-2.0*

Chapter 2 Stratix IV Transceiver Clocking
Revised: *November 2008*
Part Number: *SIV52002-2.0*

Chapter 3 Configuring Multiple Protocols and Data Rates in a Transceiver Block
Revised: *November 2008*
Part Number: *SIV52003-2.0*

Chapter 4 Reset Control and Power Down
Revised: *November 2008*
Part Number: *SIV52004-2.0*

Chapter 5 Stratix IV Dynamic Reconfiguration
Revised: *November 2008*
Part Number: *SIV52005-1.0*

About this Handbook

This handbook provides comprehensive information about the Altera® Stratix® IV family of devices.

How to Contact Altera

For the most up-to-date information about Altera products, see the following table.

Contact *(Note 1)*	Contact Method	Address
Technical support	Website	www.altera.com/support
Technical training	Website	www.altera.com/training
	Email	custrain@altera.com
Altera literature services	Email	literature@altera.com
Non-technical support (General)	Email	nacomp@altera.com
(Software Licensing)	Email	authorization@altera.com

Note:

(1) You can also contact your local Altera sales office or sales representative.

Typographic Conventions

The following table shows the typographic conventions that this document uses.

Visual Cue	Meaning
Bold Type with Initial Capital Letters	Indicates command names, dialog box titles, dialog box options, and other GUI labels. For example, **Save As** dialog box. For GUI elements, capitalization matches the GUI.
bold type	Indicates directory names, project names, disk drive names, file names, file name extensions, dialog box options, software utility names, and other GUI labels. For example, **\qdesigns** directory, **d:** drive, and **chiptrip.gdf** file.
Italic Type with Initial Capital Letters	Indicates document titles. For example, *AN 519: Stratix IV Design Guidelines.*
Italic type	Indicates variables. For example, *n* + 1. Variable names are enclosed in angle brackets (< >). For example, <*file name*> and <*project name*>.**pof** file.
Initial Capital Letters	Indicates keyboard keys and menu names. For example, Delete key and the Options menu.
"Subheading Title"	Quotation marks indicate references to sections within a document and titles of Quartus II Help topics. For example, "Typographic Conventions."

Visual Cue	Meaning
`Courier type`	Indicates signal, port, register, bit, block, and primitive names. For example, `data1`, `tdi`, and `input`. Active-low signals are denoted by suffix `n`. For example, `resetn`. Indicates command line commands and anything that must be typed exactly as it appears. For example, `c:\qdesigns\tutorial\chiptrip.gdf`. Also indicates sections of an actual file, such as a Report File, references to parts of files (for example, the AHDL keyword `SUBDESIGN`), and logic function names (for example, `TRI`).
1., 2., 3., and a., b., c., and so on.	Numbered steps indicate a list of items when the sequence of the items is important, such as the steps listed in a procedure.
■ ■	Bullets indicate a list of items when the sequence of the items is not important.
☞	The hand points to information that requires special attention.
⚠ CAUTION	A caution calls attention to a condition or possible situation that can damage or destroy the product or your work.
⚠ WARNING	A warning calls attention to a condition or possible situation that can cause you injury.
↵	The angled arrow instructs you to press **Enter**.
👣	The feet direct you to more information about a particular topic.

Section I. Transceiver Architecture

This section includes the following chapters:

- Chapter 1, Stratix IV Transceiver Architecture
- Chapter 2, Stratix IV Transceiver Clocking
- Chapter 3, Configuring Multiple Protocols and Data Rates in a Transceiver Block
- Chapter 4, Reset Control and Power Down
- Chapter 5, Stratix IV Dynamic Reconfiguration

Revision History

Refer to each chapter for its own specific revision history. For information on when each chapter was updated, refer to the Chapter Revision Dates section, which appears in the full handbook.

1. Stratix IV Transceiver Architecture

SIV52001-2.0

Introduction

Altera Stratix® IV FPGAs deliver a breakthrough level of system bandwidth and power efficiency for high-end applications, allowing you to innovate without compromise. Stratix IV GX devices provide up to 32 full-duplex CDR-based transceivers with physical coding sublayer (PCS) and physical medium attachment (PMA), at serial data rates between 600 Mbps and 8.5 Gbps. Up to an additional 16 full-duplex CDR-based transceivers with PMA, supporting serial data rates between 600 Mbps and 3.2 Gbps, are also provided.

The transceiver channels are designed to support the following serial protocols:

- PCI Express (PIPE)
  - Gen1 at 2.5 Gbps
  - Gen2 at 5 Gbps
- XAUI (3.125 Gbps to 3.75 Gbps for HiGig support)
- GIGE (1.25 Gbps)
- Serial RapidIO (1.25 Gbps, 2.5 Gbps, 3.125 Gbps)
- SONET/SDH
  - OC-12 at 622 Mbps
  - OC-48 at 2.488 Gbps
  - OC-96 at 4.976 Gbps
- (OIF) CEI PHY Interface (3.125 Gbps to 6.375 Gbps for Interlaken support)
- Serial Digital Interface (SDI)
  - HD-SDI at 1.485 Gbps and 1.4835 Gbps
  - 3G-SDI at 2.97 Gbps and 2.967 Gbps

The transceiver channels also support the following highly flexible functional modes to implement proprietary protocols:

- Basic
  - Basic single-width (600 Mbps to 3.75 Gbps)
  - Basic double-width (1 Gbps to 8.5 Gbps)

This chapter includes the following sections:

- "Transceiver Channel Locations" on page 1–2
- "Transceiver Block Architecture" on page 1–5
- "Transceiver Port List" on page 1–6
- "CMU Channels" on page 1–20
- "Transceiver Channel Architecture" on page 1–27

- "Transmitter Channel Datapath" on page 1–29
- "Transmitter Local Clock Divider Block" on page 1–49
- "Receiver Channel Datapath" on page 1–49
- "Functional Modes" on page 1–108
- "Loopback Modes" on page 1–171
- "Calibration Blocks" on page 1–175

Transceiver Channel Locations

The Stratix IV GX transceivers are structured into full-duplex (Transmitter and Receiver) four-channel groups called transceiver blocks. The total number of transceiver channels and the location of transceiver blocks varies from device to device.

Table 1–1 summarizes the total number of transceiver channels and transceiver block locations in each Stratix IV GX device member.

Table 1–1. Number of Transceiver Channels and Transceiver Block Locations in Stratix IV GX Devices

Device Member	Total Number of Transceiver Channels	Transceiver Channel Location
EP4SGX70DF29 EP4SGX110DF29 EP4SGX230DF29	8	Eight transceiver channels located in two transceiver blocks, GXBR0 and GXBR1, on the right side of the device.
EP4SGX290FH29 EP4SGX360FH29 EP4SGX110FF35 EP4SGX230FF35 EP4SGX290FF35 EP4SGX360FF35	16	Eight transceiver channels located in two transceiver blocks, GXBR0 and GXBR1, on the right side of the device. Eight transceiver channels located in two transceiver blocks, GXBL0 and GXBL1, on the left side of the device.
EP4SGX230HF35 EP4SGX290HF35 EP4SGX360HF35 EP4SGX530HH35	16	Eight transceiver channels located in two transceiver blocks, GXBR0 and GXBR1, on the right side of the device. Eight transceiver channels located in two transceiver blocks, GXBL0 and GXBL1, on the left side of the device.
EP4SGX230KF40 EP4SGX290KF40 EP4SGX360KF40 EP4SGX530KF40	24	Twelve transceiver channels located in three transceiver blocks, GXBR0, GXBR1, and GXBR2, on the right side of the device. Twelve transceiver channels located in three transceiver blocks, GXBL0, GXBL1, and GXBL2, on the left side of the device.
EP4SGX530NF45	32	Sixteen transceiver channels located in four transceiver blocks, GXBR0, GXBR1, GXBR2, and GXBR3, on the right side of the device. Sixteen transceiver channels located in four transceiver blocks, GXBL0, GXBL1, GXBL2, and GXBL3, on the left side of the device.

Figure 1–1 to 1–4 show transceiver channel locations in each Stratix IV GX device member.

Figure 1–1. Stratix IV GX Devices with Eight Transceiver Channels

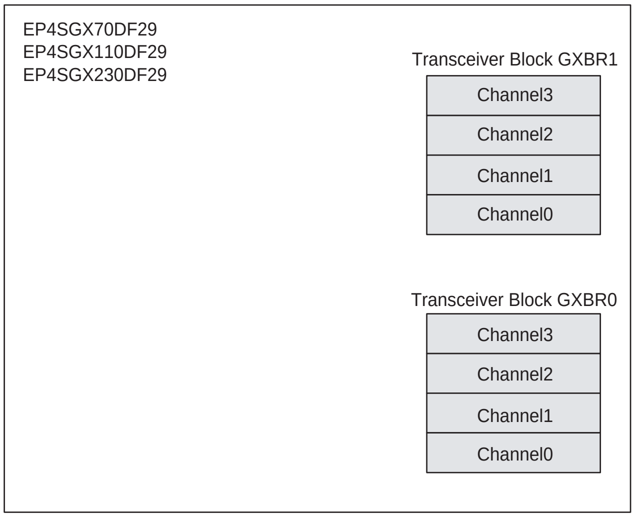


Figure 1–2. Stratix IV GX Devices with Sixteen Transceiver Channels

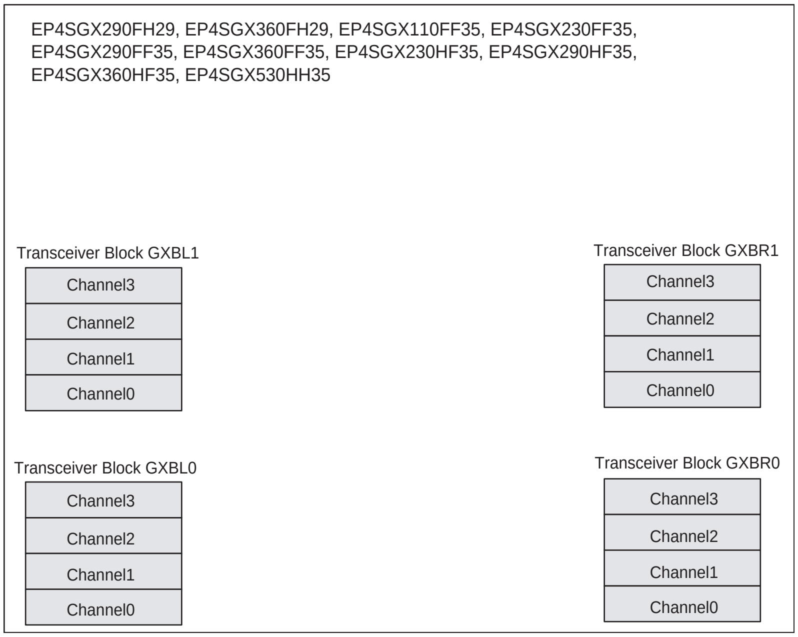


Figure 1–3. Stratix IV GX Devices with Twenty-Four Transceiver Channels

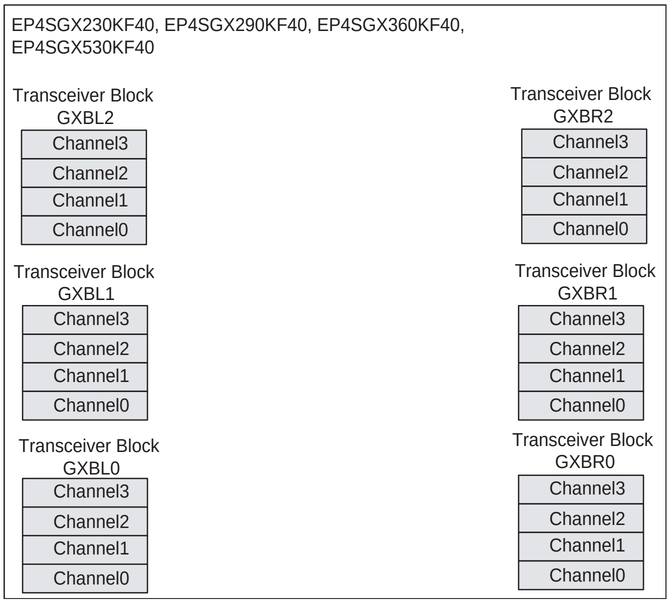
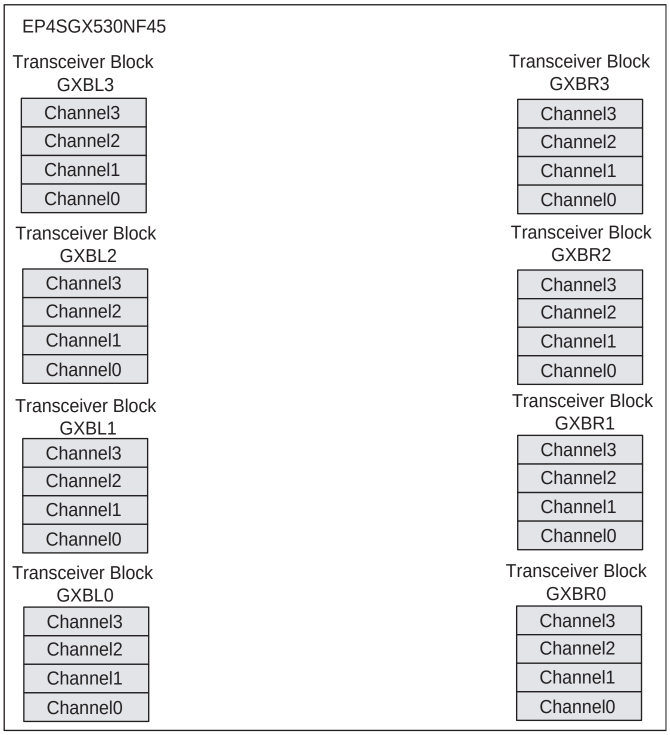


Figure 1–4. Stratix IV GX Device with Thirty-Two Transceiver Channels



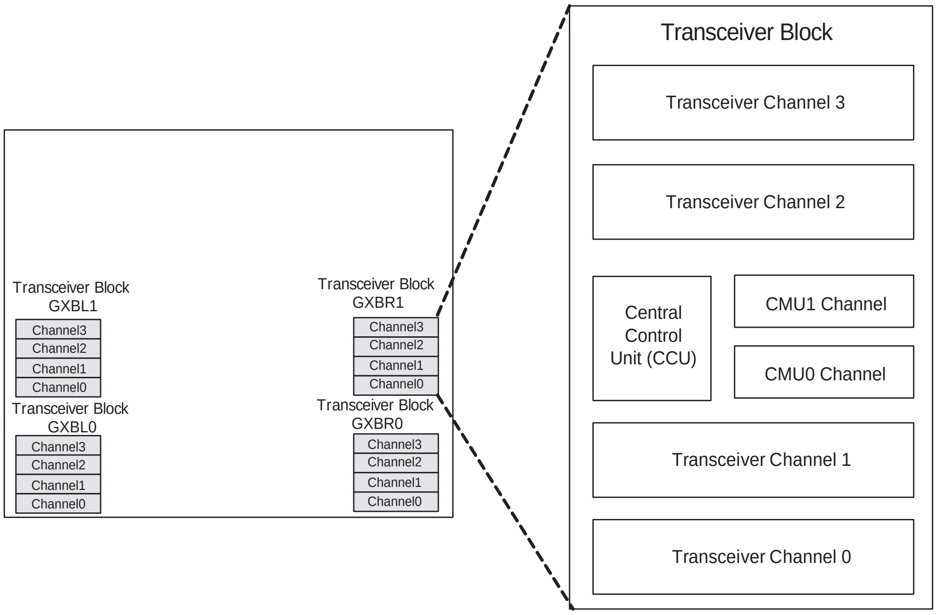
Transceiver Block Architecture

Each transceiver block has:

- Two clock multiplier unit (CMU) channels—CMU0 channel and CMU1 channel—that provide the high-speed serial and low-speed parallel clock to the transceiver channels
- Four full-duplex (Transmitter and Receiver) transceiver channels that support serial data rates from 600 Mbps to 8.5 Gbps
- Central control unit (CCU) that implements XAUI state machine for XGMII-to-PCS code group conversion, XAUI deskew state machine, shared control signal generation block, PCI Express (PIPE) rate switch controller block, and reset control logic
  - The shared control signal generation block provides control signals to the transceiver channels in bonded functional modes such as XAUI, PCI Express (PIPE), and Basic ×4.
  - The PCI Express (PIPE) rate switch controller block controls the rate switch circuit in the CMU0 channel, in ×4 configurations. In PCI Express (PIPE) ×8 configuration, the PCI Express (PIPE) rate switch controller block of the CCU in the Master Transceiver Block is active. For more information about rate switching in the PCI Express (PIPE), refer to "PCI Express Gen2 (5 Gbps) Support" on page 1–126.

Figure 1–5 shows a block diagram of the transceiver block architecture.

Figure 1–5. Top-Level View of a Transceiver Block



☞ For architecture details of CMU channels and transceiver channels, refer to "CMU Channels" on page 1–20 and "Transceiver Channel Architecture" on page 1–27.

Transceiver Port List

You instantiate the Stratix IV GX transceivers using the ALTGX megafunction instance in the Quartus® II MegaWizard® Plug-In Manager. The ALTGX megafunction instance allows you to configure the transceivers for your intended protocol and select optional control and status ports to and from the instantiated transceiver channels.

Table 1–2 provides a brief description of all the ALTGX megafunction ports.

Table 1–2. Stratix IV GX ALTGX Megafunction Ports (Part 1 of 15)

Port Name	Input/Output	Description	Scope
Clock Multiplier Unit (CMU)			
`pll_inclk`	Input	Input reference clock for the CMU phase locked loop (PLL).	Transceiver block
`pll_locked`	Output	CMU PLL lock indicator. A high level indicates that the CMU PLL is locked to the input reference clock; a low level indicates that the CMU PLL is not locked to the input reference clock. Asynchronous signal.	Transceiver block
`pll_powerdown`	Input	CMU PLL power down. When asserted high, the CMU PLL is powered down. When de-asserted low, the CMU PLL is active and locks to the input reference clock. Note: Assertion of the `pll_powerdown` signal does not power down the `refclk` buffers. Asynchronous signal. The minimum pulse-width is 1 us (pending characterization)	Transceiver block
`coreclkout`	Output	FPGA fabric-transceiver interface clock. Generated by the CMU0 clock divider in the transceiver block in ×4 bonded channel configurations. Generated by the CMU0 clock divider in the master transceiver block in ×8 bonded channel configurations. Not available in non-bonded channel configurations. This clock is used to clock the write port of the transmitter phase compensation FIFOs in all bonded channels. Use this clock signal to clock parallel data `tx_datain` from the FPGA fabric into the transmitter phase compensation FIFO of all bonded channels. This clock is used to clock the read port of the receiver phase compensation FIFOs in all bonded channels with rate match FIFO enabled. Use this signal to clock parallel data `rx_dataout` from the receiver phase compensation FIFOs of all bonded channels (with rate match FIFO enabled) into the FPGA fabric.	Transceiver block

Table 1–2. Stratix IV GX ALTGX Megafunction Ports (Part 2 of 15)

Port Name	Input/Output	Description	Scope
Receiver Physical Coding Sublayer (PCS) Ports			
Word Aligner			
`rx_enapatternalign`	Input	Manual word alignment enable control. Enables word aligner configured in manual alignment mode to align to the word alignment pattern. ■ In single-width modes with 10-bit PMA-PCS interface, this signal is level-sensitive. When high, the word aligner re-aligns if the word alignment pattern appears in a new word boundary. ■ In single-width modes with 8-bit PMA-PCS interface and all double-width modes, a low-to-high transition causes the word aligner to re-align once, if the word alignment pattern appears in a new word boundary. Asynchronous signal. The minimum pulse-width is 2 recovered clock cycles.	Channel
`rx_patterndetect`	Output	Word alignment pattern detect indicator. A high level indicates that the word alignment pattern is found on the current word boundary. The width of this signal depends on the channel width shown below: Channel Width `rx_patterndetect` 8/10 1 16/20 2 32/40 4	Channel

Table 1–2. Stratix IV GX ALTGX Megafunction Ports (Part 3 of 15)

Port Name	Input/Output	Description	Scope
`rx_syncstatus`	Output	Word alignment synchronization status indicator. For word aligner in automatic synchronization state machine mode. This signal is driven high if the conditions required to remain in synchronization are met. It is driven low if the conditions required to lose synchronization are met. For word aligner in manual alignment mode, the behavior of this signal depends on whether the transceiver is configured in single-width or double-width mode. Refer to "Word Aligner in Single-Width Mode" on page 1–65 and "Word Aligner in Double-Width Mode" on page 1–71 for more details. This signal is not available for word aligner in bit-slip mode. The width of this signal depends on the channel width shown below: Channel Width `rx_syncstatus` 8/10 1 16/20 2 32/40 4	Channel
`rx_bitslip`	Input	Bit-slip control for word aligner configured in bit-slip mode. At every rising edge of this signal, word aligner slips one bit into the received data stream, effectively shifting the word boundary by 1 bit. Asynchronous signal. The minimum pulse-width is 2 recovered clock cycles.	Channel
`rx_ala2size`	Input	Available only in SONET OC-12 and OC-48 modes to select between one of the following two word alignment options: ■ 0 - 16-bit A1A2 ■ 1 - 32-bit A1A1A2A2	Channel
`rx_rlv`	Output	Run-length violation indicator. A high pulse is driven when the number of consecutive 1's or 0's in the received data stream exceeds the programmed run length violation threshold. Asynchronous signal. Driven for a minimum of two recovered clock cycles in configurations without byte serializer and a minimum of three recovered clock cycles in configurations with byte serializer.	Channel

Table 1–2. Stratix IV GX ALTGX Megafunction Ports (Part 4 of 15)

Port Name	Input/Output	Description	Scope
`rx_invpolarity`	Input	Generic receiver polarity inversion control. Useful feature for correcting situations where the positive and negative signals of the differential serial link are accidentally swapped during board layout. When asserted high in single-width modes, the polarity of every bit of the 8-bit or 10-bit input data word to the word aligner gets inverted. When asserted high in double-width mode, the polarity of every bit of the 16-bit or 20-bit input data word to the word aligner gets inverted. Asynchronous signal.	Channel
`rx_revbitorderwa`	Input	Receiver bit reversal control. Available only in Basic single-width and double-width modes with word aligner configured in bit-slip mode. Useful feature where the link transmission order is MSBit to LSBit. When asserted high in Basic single-width modes, the 8-bit or 10-bit data `D[7:0]` or `D[9:0]` at the output of the word aligner gets rewired to `D[0:7]` or `D[0:9]`, respectively. When asserted high in Basic double-width modes, the 16-bit or 20-bit data `D[15:0]` or `D[19:0]` at the output of the word aligner gets rewired to `D[0:15]` or `D[0:19]`, respectively. Asynchronous signal.	Channel
`rx_revbyteorderwa`	Input	Receiver byte reversal control. Available only in Basic double-width mode. Useful feature in situations where the MSByte and LSByte of the transmitted data are erroneously swapped. When asserted high, the MSByte and LSByte of the 16/20-bit data at the output of the word aligner get swapped. Asynchronous signal.	Channel
Deskew FIFO			

Table 1–2. Stratix IV GX ALTGX Megafunction Ports (Part 5 of 15)

Port Name	Input/Output	Description	Scope
`rx_channelaligned`	Output	10-gigabit attachment unit interface (XAUI) deskew FIFO channel aligned indicator. Available only in XAUI mode. A high level indicates that the XAUI deskew state machine is either in ALIGN_ACQUIRED_1, ALIGN_ACQUIRED_2, ALIGN_ACQUIRED_3, or ALIGN_ACQUIRED_4 state, as specified in the PCS deskew state diagram in IEEE P802.3ae specification. A low level indicates that the XAUI deskew state machine is either in LOSS_OF_ALIGNMENT, ALIGN_DETECT_1, ALIGN_DETECT_2, or ALIGN_DETECT_3 state, as specified in the PCS deskew state diagram in IEEE P802.3ae specification.	Transceiver block
Rate Match (Clock Rate Compensation) FIFO			
`rx_rmfifodatainserted`	Output	Rate match FIFO insertion status indicator. A high level indicates that the rate match pattern byte got inserted to compensate for the parts-per-million (PPM) difference in reference clock frequencies between the upstream transmitter and the local receiver.	Channel
`rx_rmfifodatadeleted`	Output	Rate match FIFO deletion status indicator. A high level indicates that the rate match pattern byte got deleted to compensate for the PPM difference in reference clock frequencies between the upstream transmitter and the local receiver.	Channel
`rx_rmfifofull`	Output	Rate match FIFO full status indicator. A high level indicates that the rate match FIFO is full. Driven for a minimum of two recovered clock cycles in configurations without byte serializer and a minimum of three recovered clock cycles in configurations with byte serializer.	Channel
`rx_rmfifoempty`	Output	Rate match FIFO empty status indicator. A high level indicates that the rate match FIFO is empty. Driven for a minimum of two recovered clock cycles in configurations without byte serializer and a minimum of three recovered clock cycles in configurations with byte serializer.	Channel
8B/10B Decoder			

Table 1–2. Stratix IV GX ALTGX Megafunction Ports (Part 6 of 15)

Port Name	Input/Output	Description	Scope
`rx_ctrldetect`	Output	Receiver control code indicator. Available in configurations with 8B/10B decoder. A high level indicates that the associated received code group is a control (/Kx.y/) code group. A low-level indicates that the associated received code group is a data (/Dx.y/) code group. The width of this signal depends on the channel width shown below: Channel Width `rx_ctrldetect` 8 1 16 2 32 4	Channel
`rx_errdetect`	Output	8B/10B code group violation or disparity error indicator. Available in configurations with 8B/10B decoder. A high level indicates that a code group violation or disparity error was detected on the associated received code group. Use with the `rx_disperr` signal to differentiate between a code group violation and/or a disparity error as follows: `[rx_errdetect: rx_disperr]` 2'b00—no error 2'b10—code group violation 2'b11—disparity error or both The width of this signal depends on the channel width shown below: Channel Width `rx_errdetect` 8 1 16 2 32 4	Channel
`rx_disperr`	Output	8B/10B disparity error indicator port. Available in configurations with 8B/10B decoder. A high level indicates that a disparity error was detected on the associated received code group. The width of this signal depends on the channel width shown below: Channel Width `rx_disperr` 8 1 16 2 32 4	Channel

Table 1–2. Stratix IV GX ALTGX Megafunction Ports (Part 7 of 15)

Port Name	Input/Output	Description	Scope
`rx_runningdisp`	Output	8B/10B running disparity indicator. Available in configurations with the 8B/10B decoder. A high level indicates that data on the `rx_dataout` port was received with a negative running disparity. A low level indicates that data on the `rx_dataout` port was received with a positive running disparity. The width of this signal depends on the channel width shown below: Channel Width `rx_runningdisp` 8 1 16 2 32 4	Channel
Byte Ordering Block			
`rx_enabyteord`	Input	Enable byte ordering control. Available in configurations with byte ordering block enabled. The byte ordering block is rising edge sensitive to this signal. A low-to-high transition triggers the byte ordering block to restart the byte ordering operation. Asynchronous signal.	Channel
`rx_byteorderalignstatus`	Output	Byte ordering status indicator. Available in configurations with byte ordering block enabled. A high level indicates that the byte ordering block has detected the programmed byte ordering pattern in the LSByte of the received data from the byte deserializer.	Channel
Receiver Phase Compensation FIFO			
`rx_dataout`	Output	Parallel data output from the receiver to the FPGA fabric. The bus width depends on the channel width multiplied by the number of channels per instance.	
`rx_clkout`	Output	Recovered clock from the receiver channel. Available only when the rate match FIFO is not used in the receiver datapath.	Channel
`rx_coreclk`	Input	Optional read clock port for the receiver phase compensation FIFO. If not selected, the Quartus II software automatically selects `rx_clkout/tx_clkout/coreclkout` as the read clock for the receiver phase compensation FIFO. If selected, you must drive this port with a clock that has 0 PPM difference with respect to `rx_clkout/tx_clkout/coreclkout`.	Channel

Table 1–2. Stratix IV GX ALTGX Megafunction Ports (Part 8 of 15)

Port Name	Input/Output	Description	Scope
`rx_phase_comp_fifo_error`	Output	Receiver phase compensation FIFO full or empty indicator. A high level indicates that the receiver phase compensation FIFO is either full or empty.	Channel
Receiver Physical Media Attachment (PMA)			
`rx_datain`	Input	Receiver serial data input port.	Channel
`rx_cruclk`	Input	Input reference clock for the receiver clock and data recovery (CDR).	Channel
`rx_pll_locked`	Output	Receiver CDR lock-to-reference (LTR) indicator. A high level indicates that the receiver CDR is locked to the input reference clock. A low level indicates that the receiver CDR is not locked to the input reference clock. Asynchronous signal.	Channel
`rx_freqlocked`	Output	Receiver CDR lock mode indicator. A high level indicates that the receiver CDR is in lock-to-data (LTD) mode. A low level indicates that the receiver CDR is in lock-to-reference mode. Asynchronous signal.	Channel
`rx_locktodata`	Input	Receiver CDR lock-to-data mode control signal. When asserted high, the receiver CDR is forced to lock-to-data mode. When de-asserted low, the receiver CDR lock mode depends on the `rx_locktorefclk` signal level.	Channel
`rx_locktorefclk`	Input	Receiver CDR lock-to-reference mode control signal. The `rx_locktorefclk` signal along with `rx_locktodata` signal controls whether the receiver CDR is in lock-to-reference or lock-to-data mode, as follows: `rx_locktodata/ rx_locktorefclk` 0/0 - receiver CDR is in automatic mode 0/1 - receiver CDR is in LTR mode 1/x - receiver CDR is in LTD mode Asynchronous Signal.	Channel
`rx_signaldetect`	Output	Signal threshold detect indicator. Available only in PCI Express (PIPE) mode. A high level indicates that the signal present at the receiver input buffer is above the programmed signal detection threshold value. If the electrical idle inference block is disabled in PCI Express (PIPE) mode, the `rx_signaldetect` signal is inverted and driven on the `pipeelecidle` port. Asynchronous signal.	Channel

Table 1–2. Stratix IV GX ALTGX Megafunction Ports (Part 9 of 15)

Port Name	Input/Output	Description	Scope
`rx_seriallpbken`	Input	Serial loopback control port. 0 - normal data path, no serial loopback 1 - serial loopback	Channel
Transmitter Physical Coding Sublayer Ports			
Transmitter Phase Compensation FIFO			
`tx_datain`	Input	Parallel data input from the FPGA fabric to the transmitter. The bus width depends on the channel width multiplied by the number of channels per instance.	Channel
`tx_clkout`	Output	FPGA fabric-transceiver interface clock. Each channel has a `tx_clkout` signal in non-bonded channel configurations. Use this clock signal to clock the parallel data `tx_datain` from the FPGA fabric into the transmitter. This signal is not available in bonded channel configurations.	Channel
`tx_coreclk`	Input	Optional write clock port for the transmitter phase compensation FIFO. If not selected, the Quartus II software automatically selects `tx_clkout/coreclkout` as the write clock for transmitter phase compensation FIFO. If selected, you must drive this port with a clock that is frequency locked to `tx_clkout/coreclkout`.	Channel
`tx_phase_comp_fifo_error`	Output	Transmitter phase compensation FIFO full or empty indicator. A high-level indicates that the transmitter phase compensation FIFO is either full or empty.	Channel
8B/10B Encoder			
`tx_ctrlenable`	Input	8B/10B encoder /Kx.y/ or /Dx.y/ control. When asserted high, the 8B/10B encoder encodes the data on the `tx_datain` port as a /Kx.y/ control code group. When de-asserted low, it encodes the data on the `tx_datain` port as a /Dx.y/ data code group. The width of this signal depends on the channel width shown below: Channel Width `tx_ctrlenable` 8 1 16 2 32 4	Channel

Table 1–2. Stratix IV GX ALTGX Megafunction Ports (Part 10 of 15)

Port Name	Input/Output	Description	Scope
`tx_forcedisp`	Input	8B/10B encoder force disparity control. When asserted high, it forces the 8B/10B encoder to encode the data on the `tx_datain` port with a positive or negative disparity depending on the `tx_dispval` signal level. When de-asserted low, the 8B/10B encoder encodes the data on the `tx_datain` port according to the 8B/10B running disparity rules. The width of this signal depends on the channel width shown below: Channel Width `tx_forcedisp` 8 1 16 2 32 4	Channel
`tx_dispval`	Input	8B/10B encoder force disparity value. A high level on the `tx_dispval` signal when the `tx_forcedisp` signal is asserted high forces the 8B/10B encoder to encode the data on the `tx_datain` port with a negative starting running disparity. A low level on the `tx_dispval` signal when the `tx_forcedisp` signal is asserted high forces the 8B/10B encoder to encode the data on the `tx_datain` port with a positive starting running disparity. The width of this signal depends on the channel width shown below: Channel Width `tx_dispval` 8 1 16 2 32 4	Channel
`tx_invpolarity`	Input	Transmitter polarity inversion control. Useful feature for correcting situations where the positive and negative signals of the differential serial link are accidentally swapped during board layout. When asserted high in single-width modes, the polarity of every bit of the 8-bit or 10-bit input data to the serializer gets inverted. When asserted high in double-width modes, the polarity of every bit of the 16-bit or 20-bit input data to the serializer gets inverted. Asynchronous signal.	Channel
Transmitter Physical Media Attachment			
`tx_dataout`	Output	Transmitter serial data output port.	Channel
`fixedclk`	Input	125-MHz clock for receiver detect and offset cancellation in PCI Express (PIPE) mode.	

Table 1–2. Stratix IV GX ALTGX Megafunction Ports (Part 11 of 15)

Port Name	Input/Output	Description	Scope
Dynamic Reconfiguration			
`reconfig_clk`	Input	Dynamic Reconfiguration Clock. This clock is also used for offset cancellation in all modes except PCI Express (PIPE) mode. The frequency range of this clock is 2.5 MHz to 50 MHz when the transceiver channel is configured in **Transmitter only** mode. The frequency range of this clock is 37.5 MHz to 50 MHz when the transceiver channel is configured in **Receiver only** or **Receiver and Transceiver** mode.	
`reconfig_togxb`	Input	From dynamic reconfiguration controller.	
`reconfig_fromgxb`	Output	To dynamic reconfiguration controller.	
PIPE Interface (Available only in PCI Express [PIPE] functional mode)			
`powerdn`	Input	PCI Express (PIPE) power state control. Functionally equivalent to the `PowerDown[1:0]` signal defined in the PIPE specification revision 2.00. The width of this signal is two bits and is encoded as follows: ■ 2'b00: P0 - Normal Operation ■ 2'b01: P0s - Low Recovery Time Latency, Low Power State ■ 2'b10: P1 - Longer Recovery Time Latency, Lower Power State ■ 2'b11: P2 - Lowest Power State	Channel
`tx_forcedispcompliance`	Input	Force 8B/10B encoder to encode with a negative running disparity. Functionally equivalent to the `TxCompliance` signal defined in PIPE specification revision 2.00. Must be asserted high only when transmitting the first byte of the PCI Express Compliance Pattern to force the 8B/10B encode with a negative running disparity as required by the PCI Express protocol.	Channel
`tx_forceelecidle`	Input	Force transmitter buffer to PCI Express electrical idle signal levels. Functionally equivalent to the `TxElecIdle` signal defined in the PIPE specification revision 2.00.	Channel
`rateswitch`	Input	PCI Express (PIPE) rate switch control ■ 1’b0: Gen1 (2.5 Gbps) ■ 1’b1: Gen2 (5 Gbps)	

Table 1–2. Stratix IV GX ALTGX Megafunction Ports (Part 12 of 15)

Port Name	Input/Output	Description	Scope
`tx_pipeswing`	Input	Transmitter differential output voltage (VOD) swing control. Functionally equivalent to the `TxSwing` signal defined in the PIPE specification revision 2.00. Available only in PCI Express (PIPE) Gen2 configuration and decoded as follows: ■ 1'b0: - Full swing ■ 1'b1 - Low swing	
`tx_pipemargin`	Input	Transmitter differential output voltage (VOD) level control. Functionally equivalent to the `TxMargin` signal defined in the PIPE specification revision 2.00. Available only in PCI Express (PIPE) Gen2 configuration. The width of this signal is three bits per channel and is decoded as follows: ■ 3'b000 - Normal Operating Range ■ 3'b001 - Full Swing = 800 - 1200 mV Low Swing = 400 - 700mV ■ 3'b010 - TBD ■ 3'b011 - TBD ■ 3'b100 - If last value Full Swing = 200 - 400 mV Half Swing = 100 - 200mV Else TBD ■ 3'b101 - If last value Full Swing = 200 - 400 mV Half Swing = 100 - 200mV Else TBD ■ 3'b110 - If last value Full Swing = 200 - 400 mV Half Swing = 100 - 200mV Else TBD ■ 3'b111 - If last value Full Swing = 200 - 400 mV Half Swing = 100 - 200mV Else TBD	
`tx_pipedeemph`	Input	Transmitter buffer de-emphasis level control. Functionally equivalent to the `TxDeemph` signal defined in the PIPE specification revision 2.00. Available only in PCI Express (PIPE) Gen2 configuration. ■ 1'b0: -6 dB de-emphasis ■ 1'b1:-3.5 dB de-emphasis	

Table 1–2. Stratix IV GX ALTGX Megafunction Ports (Part 13 of 15)

Port Name	Input/Output	Description	Scope
pipe8b10binvpolarity	Input	PCI Express (PIPE) polarity inversion control. Functionally equivalent to the RxPolarity signal defined in the PIPE specification revision 2.00. Available only in PCI Express (PIPE) mode. When asserted high, the polarity of every bit of the 10-bit input data to the 8B/10B decoder gets inverted.	Channel
tx_detectrxloopback	Input	Receiver detect or PCI Express loopback control. Functionally equivalent to the TxDetectRx/Loopback signal defined in the PIPE specification revision 2.00. When asserted high in P1 power state with the tx_forceelecidle signal asserted, the transmitter buffer begins the receiver detection operation. Once the receiver detect completion is indicated on the pipephydonestatus port, this signal must be de-asserted. When asserted high in P0 power state with the tx_forceelecidle signal de-asserted, the transceiver datapath gets dynamically configured to support parallel loopback as described in "PCI Express (PIPE) Reverse Parallel Loopback" on page 1–174.	Channel
pipestatus	Output	PIPE receiver status port. Functionally equivalent to the RxStatus[2:0] signal defined in the PIPE specification revision 2.00. The width of this signal is three bits per channel. The encoding of receiver status on the pipestatus port is as follows: ■ 000 - Received data OK ■ 001 - 1 skip added ■ 010 - 1 skip removed ■ 011 - Receiver detected ■ 100 - 8B/10B decoder error ■ 101 - Elastic buffer overflow ■ 110 - Elastic buffer underflow ■ 111 - Received disparity error.	Channel
pipephydonestatus	Output	PHY function completion indicator. Functionally equivalent to the PhyStatus signal defined in the PIPE specification revision 2.00. Asserted high for one parallel clock cycle to communicate completion of several PHY functions, such as power state transition, receiver detection, and signaling rate change between Gen1 (2.5 Gbps) to Gen2 (5 Gbps).	Channel

Table 1–2. Stratix IV GX ALTGX Megafunction Ports (Part 14 of 15)

Port Name	Input/Output	Description	Scope
rx_pipedatavalid	Output	Valid data and control on the rx_dataout and rx_ctrldetect ports indicator. Functionally equivalent to the RxValid signal defined in the PIPE specification revision 2.00.	Channel
pipeelecidle	Output	Electrical idle detected or inferred at the receiver indicator. Functionally equivalent to the RxElecIdle signal defined in the PIPE specification revision 2.00. If the electrical idle inference block is enabled, it drives this signal high when it infers an electrical idle condition, as described in "Electrical Idle Inference" on page 1–125. Otherwise, it drives this signal low. If the electrical idle inference block is disabled, the rx_signaldetect signal from the signal detect circuitry in the receiver buffer is inverted and driven on this port. Asynchronous Signal.	Channel
Reset and Power Down			
gxb_powerdown	Input	Transceiver block power down. When asserted high, all digital and analog circuitry within the PCS, PMA, CMU channels, and the CCU of the transceiver block gets powered down. Note that asserting the gxb_powerdown signal does not power down the refclk buffers. Asynchronous signal. The minimum pulse-width is 1 us (pending characterization).	Transceiver block
rx_digitalreset	Input	Receiver PCS reset. When asserted high, the receiver PCS blocks get reset. Refer to the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook* for more details. The minimum pulse width is two parallel clock cycles.	Channel
rx_analogreset	Input	Receiver PMA reset. When asserted high, analog circuitry within the receiver PMA gets reset. Refer to the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook* for more details. The minimum pulse width is two parallel clock cycles.	Channel
tx_digitalreset	Input	Transmitter PCS reset. When asserted high, the transmitter PCS blocks get reset. Refer to the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook* for more details. The minimum pulse width is two parallel clock cycles.	Channel
Calibration Block			

Table 1–2. Stratix IV GX ALTGX Megafunction Ports (Part 15 of 15)

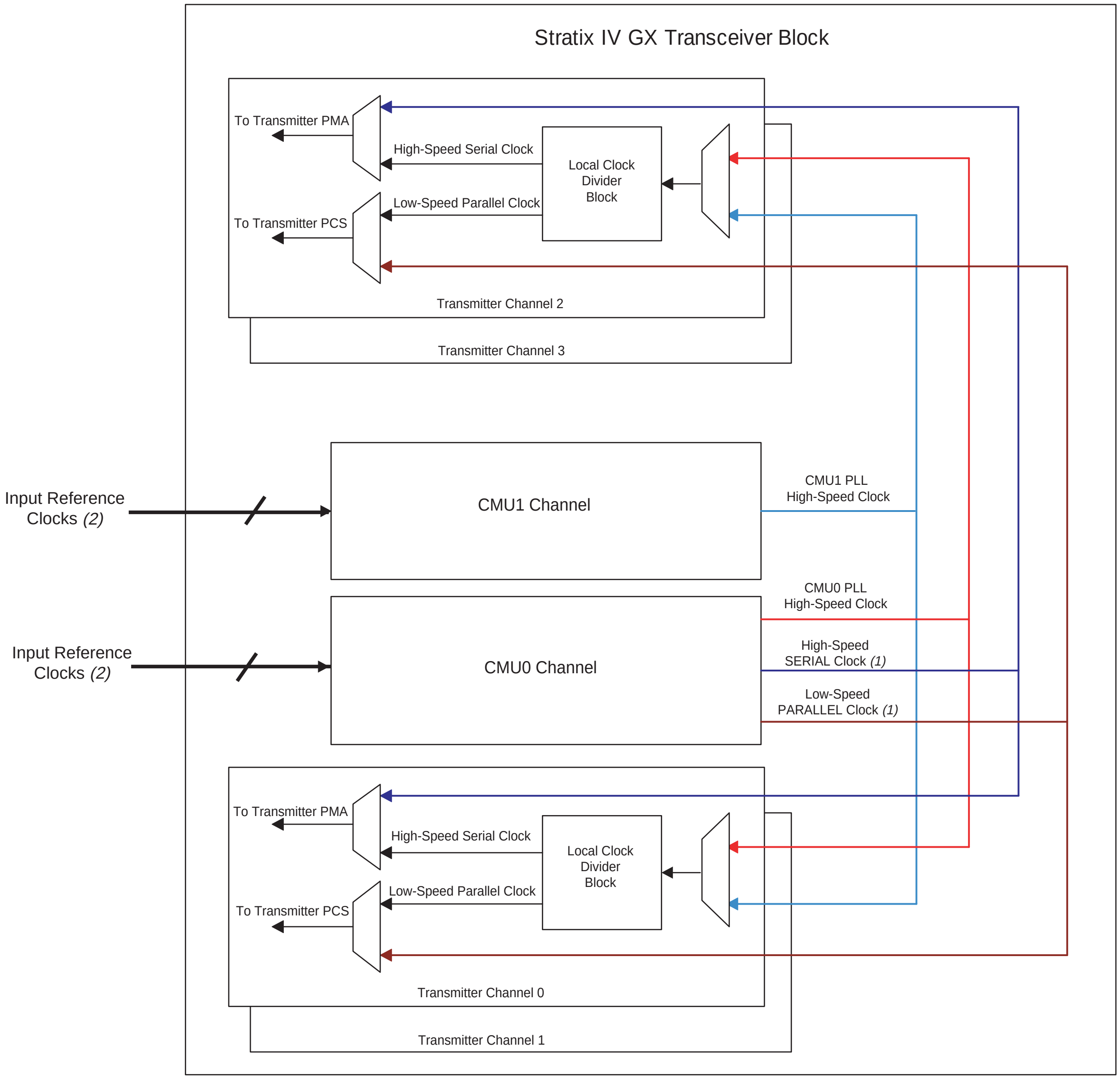
Port Name	Input/Output	Description	Scope
cal_blk_clk	Input	Clock for transceiver calibration blocks.	Device
cal_blk_powerdown	Input	Calibration block power down control.	Device

CMU Channels

The Stratix IV GX device contains two CMU channels—the CMU0 and CMU1 channels—within each transceiver block. Each CMU channel contains a CMU PLL that provides clocks to all the transmitter channels within the same transceiver block. The CMU0 channel has additional capabilities to support bonded protocol functional modes such as Basic ×4, XAUI, and PCI Express (PIPE). You can select these functional modes from the ALTGX MegaWizard Plug-In Manager. You can enable Basic ×4 functional mode in the ALTGX MegaWizard Plug-In Manager by selecting the ×4 option in Basic mode.

Figure 1–6 shows a top-level block diagram of the CMU channels in a transceiver block.

Figure 1–6. Top-Level Diagram of CMU Channels in a Transceiver Block



Notes to Figure 1–6:

(1) Clocks provided to support bonded channel functional mode.

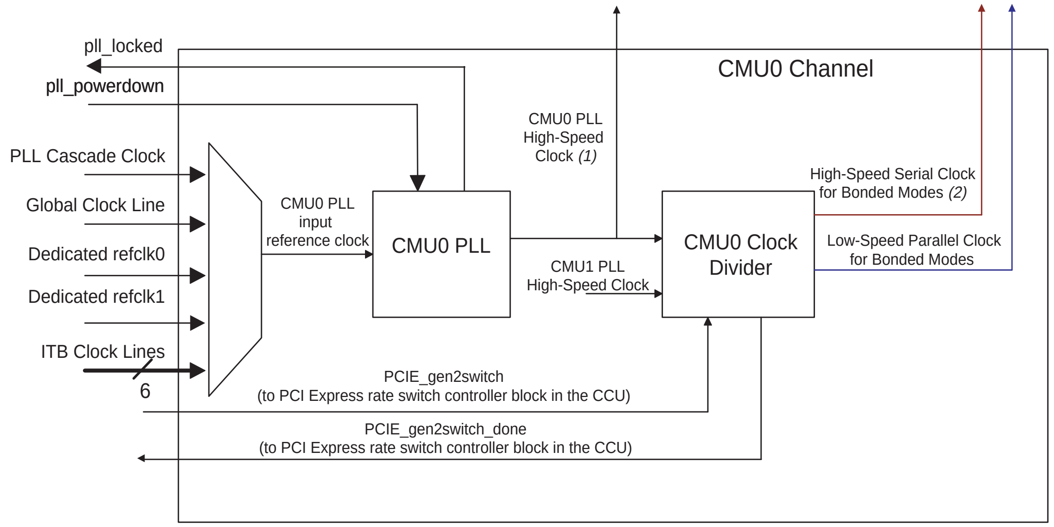
(2) For more information, refer to the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*.

CMU0 Channel

The CMU0 channel, shown in Figure 1–7, contains the following blocks:

- CMU0 PLL
- CMU0 clock divider

Figure 1–7. Diagram of CMU0 Channel


pll_locked
pll_powerdown
CMU0 Channel
PLL Cascade Clock
Global Clock Line
Dedicated refclk0
Dedicated refclk1
ITB Clock Lines
6
CMU0 PLL input reference clock
CMU0 PLL
CMU0 PLL High-Speed Clock (1)
CMU1 PLL High-Speed Clock
High-Speed Serial Clock for Bonded Modes (2)
CMU0 Clock Divider
Low-Speed Parallel Clock for Bonded Modes
PCIE_gen2switch
(to PCI Express rate switch controller block in the CCU)
PCIE_gen2switch_done
(to PCI Express rate switch controller block in the CCU)

Notes to Figure 1–7:

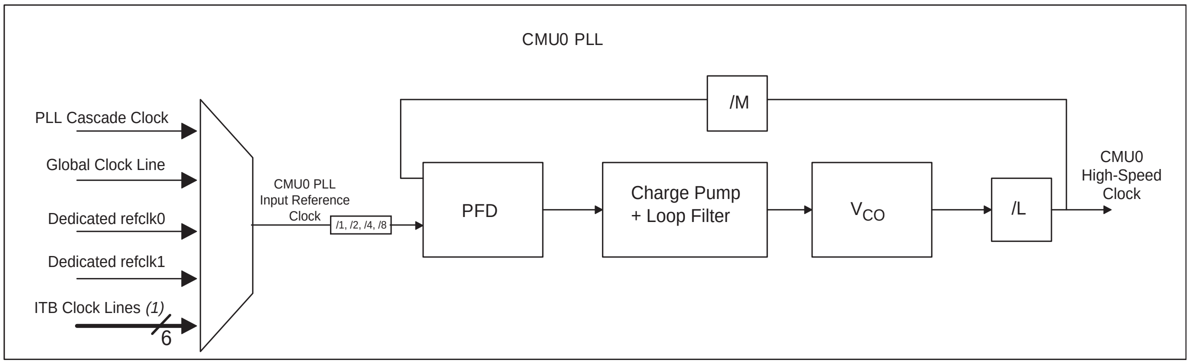
(1) In non-bonded functional modes (for example, GIGE functional mode), the transmitter channel uses the transmitter local clock divider to divide this high-speed clock output to provide clocks for its PMA and PCS blocks.

(2) Used in XAUI, Basic ×4, and PCI Express (PIPE) ×4 functional modes. In PCI Express (PIPE) ×8 functional mode, only the CMU0 channel of the master transceiver block provides clock output to all eight transceiver channels configured in PCI Express (PIPE) functional mode.

CMU0 PLL

Figure 1–8 shows the block diagram of the CMU0 PLL.

Figure 1–8. Diagram of the CMU0 PLL



Note to Figure 1–8:

(1) The ITB clock lines shown are the maximum value. The actual number of ITB lines in your device depends on the number of transceiver blocks on one side of the device.

For more information about input reference clocks, refer to the *CMU PLL and Receiver CDR Input Reference Clocks* section of the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*.

You can select the input reference clock to the CMU0 PLL from multiple clock sources. The various clock sources are:

- PLL cascade clock—the PLL cascade clock is the output from the general purpose PLLs in the FPGA fabric
- Global clock line—the input reference clock from the dedicated CLK pins are connected to the global clock line
- `refclk0`—dedicated `refclk` in the transceiver block
- `refclk1`—dedicated `refclk` in the transceiver block
- Inter transceiver block (ITB) lines—the ITB lines connect the `refclk0` and `refclk1` of all other transceiver blocks on the same side of the device.

The CMU0 PLL generates the high-speed clock from the input reference clock. The phase frequency detector (PFD) tracks the VCO output with the input reference clock. The input frequency range of the PFD is 50 to 325 MHz.

For more information about transceiver input reference clocks, refer to the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*

The voltage-controlled oscillator (VCO) in the CMU0 PLL is half rate and runs at half the serial data rate. The CMU0 PLL uses two multiplier blocks (/M and /L) in the feedback path (shown in Figure 1–8) to generate the high-speed clock needed to support a native data rate range of 600 Mbps to 8.5 Gbps. Table 1–3 lists the available /M and /L settings.

The Quartus II software automatically selects the /M and /L settings based on the input reference clock frequency and serial data rate.

Each CMU PLL (CMU0 PLL and CMU1 PLL) has a dedicated `pll_locked` signal that gets asserted to indicate that the CMU PLL is locked to the input reference clock. You can use the `pll_locked` signal in your transciever reset sequence as described in *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook*.

Table 1–3. Multiplier Block Settings in the CMU0 PLL

Multiplier Block	Available Values
/M	1, 4, 5, 8, 10,16, 20, 25
/L	1, 2, 4

PLL Bandwidth Setting

You can program the PLL bandwidth setting using the ALTGX MegaWizard Plug-In Manager. The bandwidth of a PLL is the measure of its ability to track input clock and jitter. It is determined by the –3 dB frequency of the closed-loop gain of the PLL. There are three bandwidth settings: high, medium, and low.

- The high bandwidth setting filters out internal noise from the VCO because it tracks the input clock above the frequency of the internal VCO noise.
- With the low bandwidth setting, if the noise on the input reference clock is greater than the internal noise of the VCO, the PLL filters out the noise above the –3 dB frequency of the closed-loop gain of the PLL.
- The medium bandwidth setting is a compromise between the high and low settings.

The –3 dB frequencies for these settings can vary because of the non-linear nature and frequency dependencies of the circuit.

Power Down CMU0 PLL

You can power down the CMU0 PLL by asserting the `pll_powerdown` signal.

For more information about recommended reset sequences, refer to the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook*.

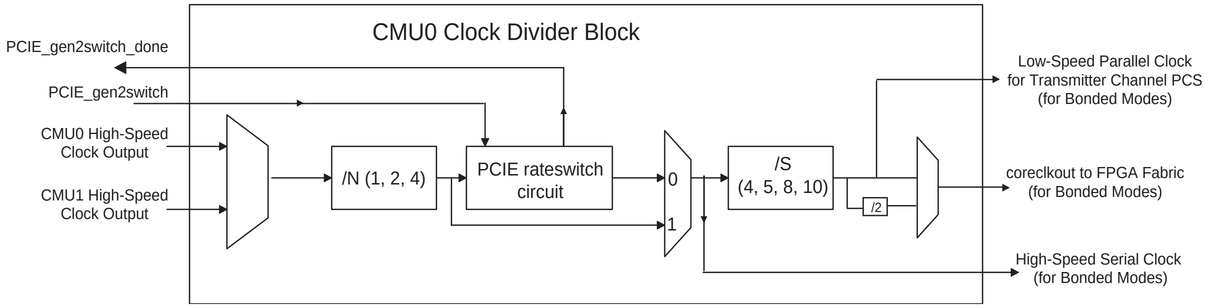
CMU0 Clock Divider Block

The high-speed clock output from the CMU0 PLL is forwarded to two clock divider blocks: the CMU0 clock divider block and the transmitter channel local clock divider block. This clock divider block is used only in bonded channel functional modes. In all non-bonded functional modes (example GIGE functional mode), the local clock divider block divides the high-speed clock to provide clocks for its PCS and PMA blocks. This section only discusses the CMU0 clock divider block.

For more information about the local clock divider block, refer to the *Transceiver Channel Datapath Clocking* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*.

You can configure the CMU0 clock divider block, shown in Figure 1–9, to select the high-speed clock output from the CMU0 PLL or CMU1 PLL. The CMU1 PLL is present in the CMU1 channel.

Figure 1–9. Diagram of the CMU0 Clock Divider Block


CMU0 Clock Divider Block
PCIE_gen2switch_done
PCIE_gen2switch
CMU0 High-Speed Clock Output
CMU1 High-Speed Clock Output
/N (1, 2, 4)
PCIE rateswitch circuit
0
1
/S (4, 5, 8, 10)
/2
Low-Speed Parallel Clock for Transmitter Channel PCS (for Bonded Modes)
coreclkout to FPGA Fabric (for Bonded Modes)
High-Speed Serial Clock (for Bonded Modes)

High-Speed Serial Clock Generation

The /N divider receives the high-speed clock output from one of the CMU PLLs and produces a high-speed serial clock. This high-speed serial clock is used for bonded functional modes such as Basic ×4, XAUI, and PCI Express (PIPE) ×4 configurations. In XAUI and Basic ×4 modes, the Quartus II software chooses the path (shown by 1 in the mux) and provides the high-speed serial clock to all the transmitter channels within the transceiver block.

In PCI Express (PIPE) ×4 mode, the clock path through the PCIE rateswitch circuit block is selected. This high-speed serial clock is provided to all the transmitter channels.

In PCI Express (PIPE) ×8 mode, only the CMU0 clock divider of the master transceiver block provides the high-speed serial clock to all eight channels.

In PCI Express (PIPE) ×1 mode, the CMU0 clock divider does NOT provide high-speed serial clock. Instead, the local clock divider block in the transmitter channel receives the CMU0 PLL or CMU1 PLL high-speed clock output and generates the high-speed serial clock to its serializer.

PCIE Rateswitch Circuit

The PCIE rateswitch circuit is enabled only in PCI Express (PIPE) ×4 mode. In PCI Express (PIPE) ×8 mode, the PCIE rateswitch circuit of the CMU0 clock divider of the master transceiver block is active. There are two paths in the PCIE rateswitch circuit. One path divides the /N output by two. The other path forwards the /N divider output.

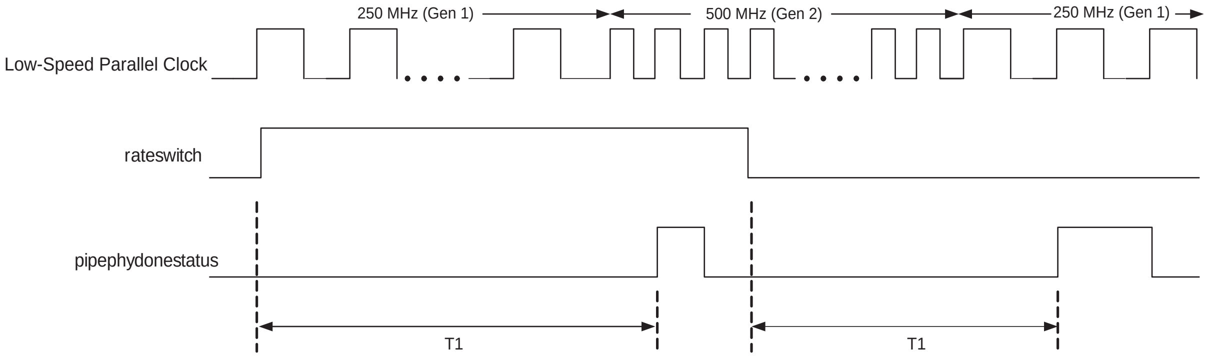
When you set the `rateswitch` port to 0, the PCI Express (PIPE) rate switch controller (in the CCU) signals the PCIE rateswitch circuit to select the divide by /2 to provide a high-speed serial clock for the Gen1 (2.5 Gbps) data rate. When the `rateswitch` port is set to 1, the /N divider output is forwarded, providing a high-speed serial clock for Gen2 (5 Gbps) data rate to the transmitter channels.

☞ The PCIE rateswitch circuit performs the rate switch operation only for the transmitter channels. For the receiver channels, the rateswitch circuit within the receiver CDR performs the rate switch operation.

The PCIE rateswitch circuit is controlled by the PCI Express (PIPE) rate switch controller in the CCU. The PCI Express (PIPE) rate switch controller asserts the `pipephydonestatus` signal for one clock cycle after the rate switch operation is completed for both the transmit and receive channels. The timing diagram for the rate switch operation is shown in Figure 1–10.

For more information about PCI Express (PIPE) functional mode rate switching, refer to "PCI Express Gen2 (5 Gbps) Support" on page 1–126.

Figure 1–10. Rate Switch in PCI Express (PIPE) Mode *(Note 1)*



Note to Figure 1–10:

(1) Time T1 is pending characterization.

☞ When you create a PCI Express (PIPE) Gen2 configuration, the CMU PLL is configured to 5 Gbps. This helps to generate the 2.5 Gbps and 5 Gbps high-speed serial clock using the rate switch circuit.

Low-Speed Parallel Clock Generation

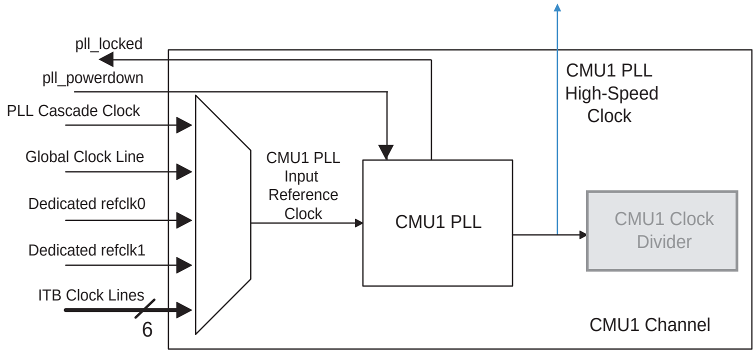
The /S divider receives the clock output from the /N divider or PCIE rateswitch circuit (only in PCI Express [PIPE] mode) and generates the low-speed parallel clock for the PCS block of all transmitter channels and `coreclkout` for the FPGA fabric. If the byte serializer block is enabled in the bonded channel modes, the /S divider output is divided by the /2 divider and sent out as `coreclkout` to the FPGA fabric. The Quartus II software automatically selects the /S values based on the deserialization width setting (single-width mode or double-width mode) that you select in the ALTGX MegaWizard Plug-In Manager. For more information about single-width or double-width mode, refer to "Transceiver Channel Architecture" on page 1–27.

☞ The Quartus II software automatically selects all the divider settings based on the input clock frequency, data rate, deserialization width, and channel width settings.

CMU1 Channel

The CMU1 channel shown in Figure 1–11 contains the CMU1 PLL that provides the high-speed clock to the transmitter channels within the transceiver block. The CMU1 PLL is similar to the CMU0 PLL. The functionality of the CMU0 PLL is discussed in "CMU0 PLL" on page 1–22.

Figure 1–11. CMU1 Channel (Grayed Area Shows the Inactive Block)



The CMU1 PLL generates the high-speed clock that is only used in non-bonded functional modes. In non-bonded functional modes, the transmitter channels within the transceiver block can receive high-speed clock from either of the two CMU PLLs and use local dividers to provide clocks to its PCS and PMA blocks.

For more information about using two CMU PLLs to configure transmitter channels, refer to the *Configuring Multiple Protocols and Data Rates* chapter in volume 2 of *Stratix IV Device Handbook*.

Power Down CMU1 PLL

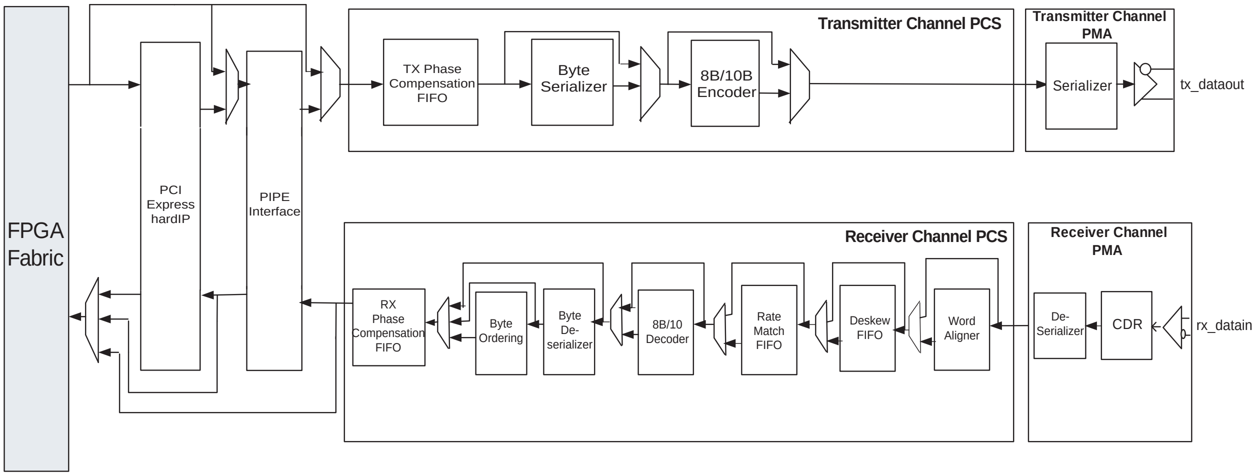
You can power down the CMU1 PLL by asserting the `pll_powerdown` signal.

For more information about using the `pll_powerdown` signal, refer to the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook*.

Transceiver Channel Architecture

Figure 1–12 shows the Stratix IV GX transceiver channel datapath.

Figure 1–12. Stratix IV Transceiver Datapath



Each transceiver channel consists of:

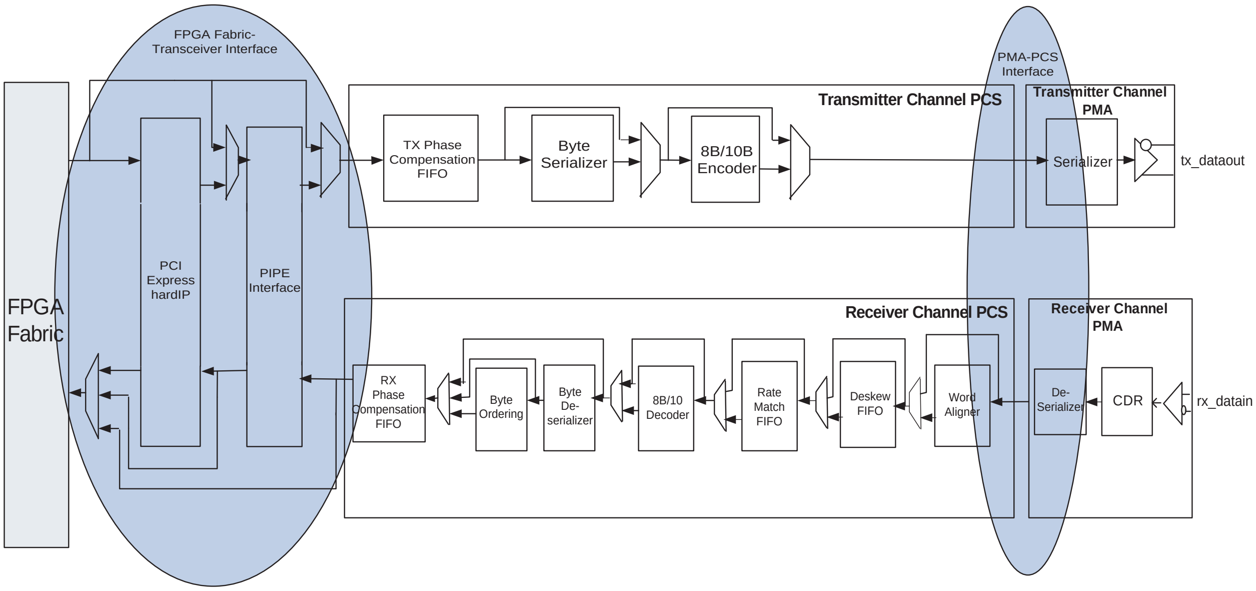
- Transmitter channel, further divided into
  - Transmitter channel PCS
  - Transmitter channel PMA
- Receiver channel, further divided into
  - Receiver channel PCS
  - Receiver channel PMA

Each transceiver channel interfaces to either the PCI Express hard IP block (PCI Express hard IP—transceiver interface) or directly to the FPGA fabric (FPGA fabric—transceiver interface). The transceiver channel interfaces to the PCI Express hard IP block if the hard IP block is used to implement the PCI Express PHY MAC, Data Link Layer, and Transaction Layer. Otherwise, the transceiver channel interfaces directly to the FPGA fabric.

☞ The PCI Express hard IP—transceiver interface is out of the scope of this chapter. This chapter focusses on the FPGA fabric—transceiver interface.

Figure 1–13 shows the FPGA fabric—transceiver interface and the transceiver PMA-PCS interface.

Figure 1–13. FPGA Fabric—Transceiver Interface and the Transceiver PMA-PCS Interface



The transceiver channel datapath can be broadly broken down into the following two modes based on the FPGA fabric—transceiver interface width (channel width) and transceiver channel PMA-PCS width (serialization factor):

- Single-width mode
- Double-width mode

Table 1–4 shows the FPGA fabric—transceiver interface widths (channel width) and transceiver PMA-PCS widths (serialization factor) allowed in single-width and double-width modes.

Table 1–4. FPGA Fabric—Transceiver Interface Width and Transceiver PMA-PCS Widths

Name	Single-Width	Double-Width
PMA-PCS interface widths	8/10 bit	16/20 bit
FPGA fabric—transceiver interface width	8/10 bit 16/20 bit	16/20 bit 32/40 bit
Supported functional modes	■ PCI Express (PIPE) Gen1 and Gen2 ■ XAUI ■ GIGE ■ Serial RapidIO ■ SONET/SDH OC12 and OC48 ■ SDI ■ Basic single width	■ (OIF) CEI PHY Interface ■ SONET/SDH OC96 ■ Basic double-width
Data rate range in Basic functional mode	0.6 Gbps - 3.75 Gbps	1 Gbps - 8.5 Gbps

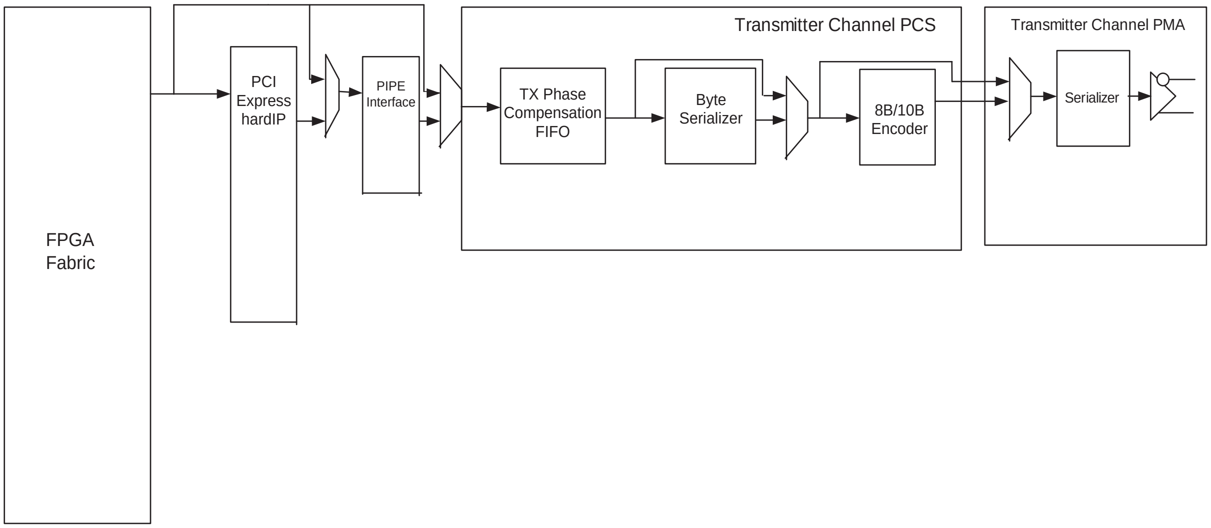
Transmitter Channel Datapath

The transmitter channel datapath, shown in Figure 1–14, consists of the following blocks:

- TX phase compensation FIFO
- Byte serializer
- 8B/10B encoder
- Transmitter output buffer

The Stratix IV GX transceiver provides the enable low latency PCS mode option in the ALTGX MegaWizard Plug-In Manager. If you select this option, the 8B/10B encoder in the data path is disabled.

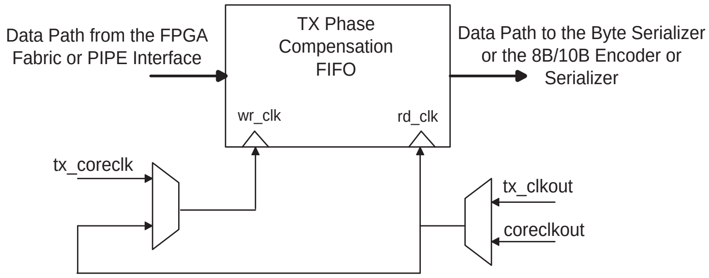
Figure 1–14. Transmitter Channel Datapath



TX Phase Compensation FIFO

The TX phase compensation FIFO interfaces the transmitter channel PCS and the FPGA fabric PIPE interface. It compensates for the phase difference between the low-speed parallel clock and the FPGA fabric interface clock. The two modes in which the TX phase compensation FIFO operates are low-latency and high-latency mode. Figure 1–15 shows the datapath and clocking of the TX phase compensation FIFO.

Figure 1–15. TX Phase Compensation FIFO


Data Path from the FPGA Fabric or PIPE Interface
TX Phase Compensation FIFO
wr_clk
rd_clk
Data Path to the Byte Serializer or the 8B/10B Encoder or Serializer
tx_coreclk
tx_clkout
coreclkout

TX phase compensation FIFO:

- In low-latency mode, the FIFO is four words deep. The latency through the FIFO is 2 to 3 FPGA fabric parallel clock cycles (pending characterization). Low-latency mode is chosen automatically in every mode.
- In high-latency mode, the FIFO is eight words deep. The latency through the FIFO is approximately 4 to 5 FPGA parallel cycles (pending characterization).

In non-bonded functional modes such as GIGE, the read port of the phase compensation FIFO is clocked by the low-speed parallel clock. The write clock is fed by the `tx_clkout` port of the associated channel. In bonded functional modes such as XAUI, the write clock of the FIFO is clocked by `coreclkout` provided by the CMU0 clock divider block**.** You can clock the write side using `tx_coreclk` provided from the FPGA fabric by enabling the `tx_coreclk` port in the ALTGX MegaWizard Plug-In Manager. If you use this port, ensure that there is 0 PPM difference in frequency between the write and read side. The Quartus II software requires that you provide a 0 PPM assignment in the assignment editor.

For more information about the TX phase compensation FIFO, refer to the *Limitation of the Quartus II Software Selected Transmitter Phase Compensation FIFO Clocks* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*.

Input Data

In PCI Express (PIPE) functional mode, the input data comes from the PIPE interface. In all other functional modes, the input data comes directly from the FPGA fabric.

Output Data Destination Block

The output from the TX phase compensation FIFO is used by the byte serializer block, 8B/10B encoder, or serializer block. Table 1–5 lists the conditions under which the TX phase compensation FIFO outputs are provided to these blocks.

Table 1–5. Output Data Destination Block for TX Phase Compensation FIFO Output Data

Byte Serializer	8B/10B Encoder	Serializer
If you select: single-width mode and channel width = 16 or 20	If you select: single-width mode and channel width = 8 and 8B/10B encoder enabled	If you select: low latency PCS bypass mode enabled or single-width mode and channel width = 8 or 10
If you select: double-width mode and channel width = 32 or 40	If you select: double-width mode and channel width = 16 and 8B/10B encoder enabled	If you select: low latency PCS bypass mode enabled or double-width mode and channel width = 16 or 20

TX Phase Compensation FIFO Status Signal

An optional `tx_phase_comp_fifo_error` port is available in all functional modes to indicate a receiver phase compensation FIFO overflow or underrun condition. The `tx_phase_comp_fifo_error` signal is asserted high when the TX phase compensation FIFO either overflows or underruns due to any frequency PPM difference between the FIFO read and write clocks. If the `tx_phase_comp_fifo_error` flag gets asserted, verify the FPGA fabric-transceiver interface clocking to ensure that there is 0 PPM difference between the TX phase compensation FIFO read and write clocks.

Byte Serializer

The byte serializer divides the input datapath by two. This allows you to run the transceiver channel at higher data rates while keeping the FPGA fabric interface frequency within the maximum limit of 250 MHz. In single-width mode, it converts the two-byte wide datapath to a one-byte wide datapath. In double-width mode, it converts the four-byte wide datapath to a two-byte wide datapath.

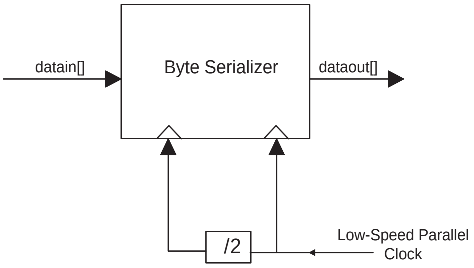
For example, if you would like to run the transceiver channel at 6.25 Gbps, without the byte serializer, in double-width mode, the FPGA fabric interface clock frequency must be 312.5 MHz (6.25/20). This violates the FPGA fabric interface frequency limit. When you use the byte serializer, the FPGA fabric interface frequency is 156.25 MHz (6.25G/40). You can enable the byte serializer in single-width or double-width mode.

☞ The byte deserializer is required in configurations that exceed the FPGA fabric—transceiver interface clock upper frequency limit. It is optional in configurations that does not exceed the FPGA fabric—transceiver interface clock upper frequency limit.

Single-Width Mode

Figure 1–16 shows the byte serializer datapath in single-width mode.

Figure 1–16. Byte Serializer Datapath in Single-Width Mode *(Note 1)*, *(2)*



Notes to Figure 1–16:

(1) Refer to Table 1–6 for the `datain[]` and `dataout[]` port width.

(2) The `datain` signal is the input from the FPGA fabric that has already passed through the TX phase compensation FIFO.

The byte serializer forwards the least significant byte first, followed by the most significant byte. The input data width to the byte serializer depends on the channel width option that you selected in the ALTGX MegaWizard Plug-In Manager. For example, in single-width mode, assuming a channel width of 20, the byte serializer sends out the least significant word `datain[9:0]` of the parallel data from the FPGA fabric, followed by `datain[19:10]`. Table 1–6 shows the input and output data widths of the byte serializer in single-width mode.

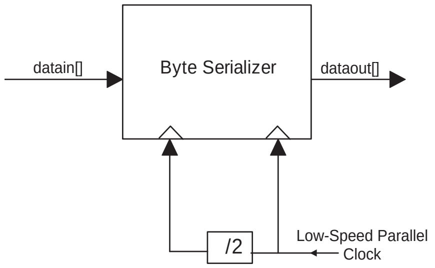
Table 1–6. Input and Output Data Width of the Byte Serializer in Single-Width Mode

Deserialization Width	Input Data Width to the Byte Serializer	Output Data Width from the Byte Serializer
Single-width mode	16	8
	20	10

Double-Width Mode

Figure 1–17 shows the byte serializer datapath in double-width mode.

Figure 1–17. Byte Serializer Datapath in Double-Width Mode *(Note 1)*, *(2)*



Notes to Figure 1–17:

(1) Refer to Table 1–7 for the `datain[]` and `dataout[]` port width.

(2) The `datain` signal is the input from the FPGA fabric that has already passed through the TX phase compensation FIFO.

The operation in double-width mode is similar to that of single-width mode. For example, assuming a channel width of 40, the byte serializer forwards `datain[19:0]` first, followed by `datain[39:20]`. Table 1–7 shows the input and output data widths of the byte serializer in double-width mode.

Table 1–7. Input and Output Data Width of the Byte Serializer in Double-Width Mode

Deserialization Width	Input Data Width to the Byte Serializer	Output Data Width from the Byte Serializer
Double-width mode	32	16
	40	20

Asserting the `tx_digitalreset` signal resets the byte serializer block.

If you select the **8B/10B Encoder** option in the ALTGX MegaWizard Plug-In Manager, the 8B/10B encoder uses the output from the byte serializer. Otherwise, the byte serializer output is forwarded to the serializer.

8B/10B Encoder

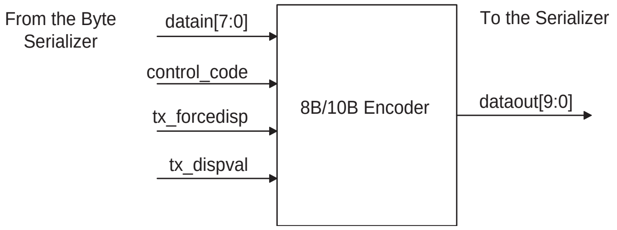
The 8B/10B encoder generates 10-bit code groups from the 8-bit data and 1-bit control identifier. The 8B/10B encoder operates in two modes: single-width and double-width.

- In single-width mode, the 8B/10B encoder generates a 10-bit code group from the 8-bit data and 1-bit control identifier.
- In double-width mode, there are two 8B/10B encoders that are cascaded together to generate two 10-bit code groups from two 8-bit data and their respective control identifiers.

Single-Width Mode

Figure 1–18 shows the inputs and outputs of the 8B/10B encoder.

Figure 1–18. 8B/10B Encoder in Single-Width Mode

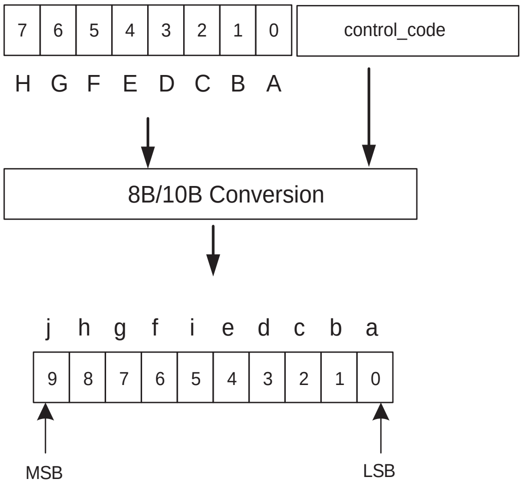


In single-width mode, the 8B/10B encoder translates the 8-bit data to a 10-bit code group (control word or data word) with proper disparity. If the `control_code` input is high, the 8B/10B encoder translates the input `data[7:0]` to a 10-bit control word. If the `control_code` input is low, the 8B/10B encoder translates the input `data[7:0]` to a 10-bit data word.

You can use the `tx_forcedisp` and `tx_dispval` ports to control the running disparity of the generated output data. For more information, see "Controlling Running Disparity" on page 1–38.

Figure 1–19 shows the conversion format. The LSB is transmitted first.

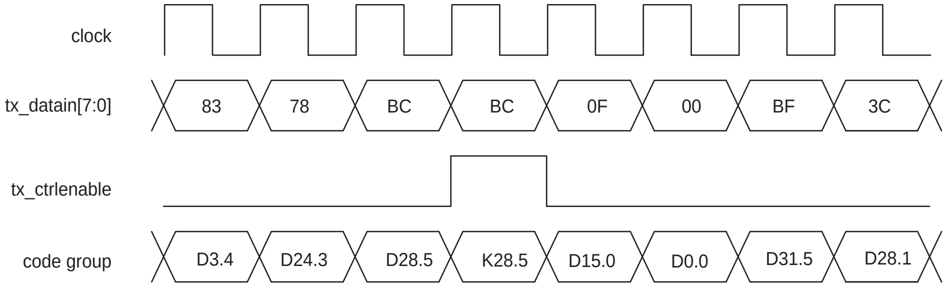
Figure 1–19. 8B/10B Conversion Format



Control Code Encoding

The ALTGX MegaWizard Plug-In Manager provides the `tx_ctrlenable` port to indicate whether the 8-bit data at the `tx_datain` port should be encoded as a control word (Kx.y). When `tx_ctrlenable` is low, the 8B/10B encoder block encodes the byte at the `tx_datain` port (the user-input port) as data (Dx.y). When `tx_ctrlenable` is high, the 8B/10B encoder encodes the input data as a Kx.y code group. The waveform in Figure 1–20 shows the second 0 × BC encoded as a control word (K28.5). The rest of the `tx_datain` bytes are encoded as a data word (Dx.y).

Figure 1–20. Control Word and Data Word Transmission



The IEEE 802.3 8B/10B encoder specification identifies only a set of 8-bit characters for which `tx_ctrlenable` should be asserted. If you assert `tx_ctrlenable` for any other set of bytes, the 8B/10B encoder might encode the output 10-bit code as an invalid code (it does not map to a valid Dx.y or Kx.y code), or unintended valid Dx.y code, depending on the value entered. It is possible for a downstream 8B/10B decoder to decode an invalid control word into a valid Dx.y code without asserting any code error flags.

☞ For example, depending on the current running disparity, the invalid code K24.1 (`tx_datain` = 8'h38 + `tx_ctrl` = 1'b1) can be encoded to 10'b0110001100 (0 × 18C), which is equivalent to a D24.6+ (8'hD8 from the RD+ column). Altera® recommends that you do not assert `tx_ctrlenable` for unsupported 8-bit characters.

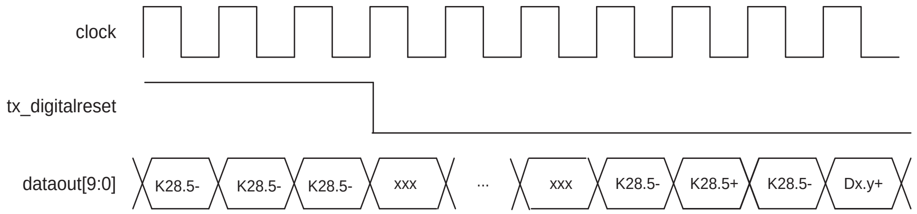
Reset Condition

The tx_digitalreset signal resets the 8B/10B encoder. During reset, the running disparity and data registers are cleared. Also, the 8B/10B encoder outputs a K28.5 pattern from the RD- column continuously until tx_digitalreset is deasserted. The input data and control code from the FPGA fabric is ignored during the reset state. Once out of reset, the 8B/10B encoder starts with a negative disparity (RD-) and transmits three K28.5 code groups for synchronization before it starts encoding and transmitting the data on its output.

☞ While tx_digitalreset is asserted, the downstream 8B/10B decoder that receives the data might observe synchronization or disparity errors.

Figure 1–21 shows the reset behavior of the 8B/10B encoder. When in reset (tx_digitalreset is high), a K28.5- (K28.5 10-bit code group from the RD- column) is sent continuously until tx_digitalreset is low. Due to some pipelining of the transmitter channel PCS, some “don’t cares” (10'hxxx) are sent before the three synchronizing K28.5 code groups. User data follows the third K28.5 code group.

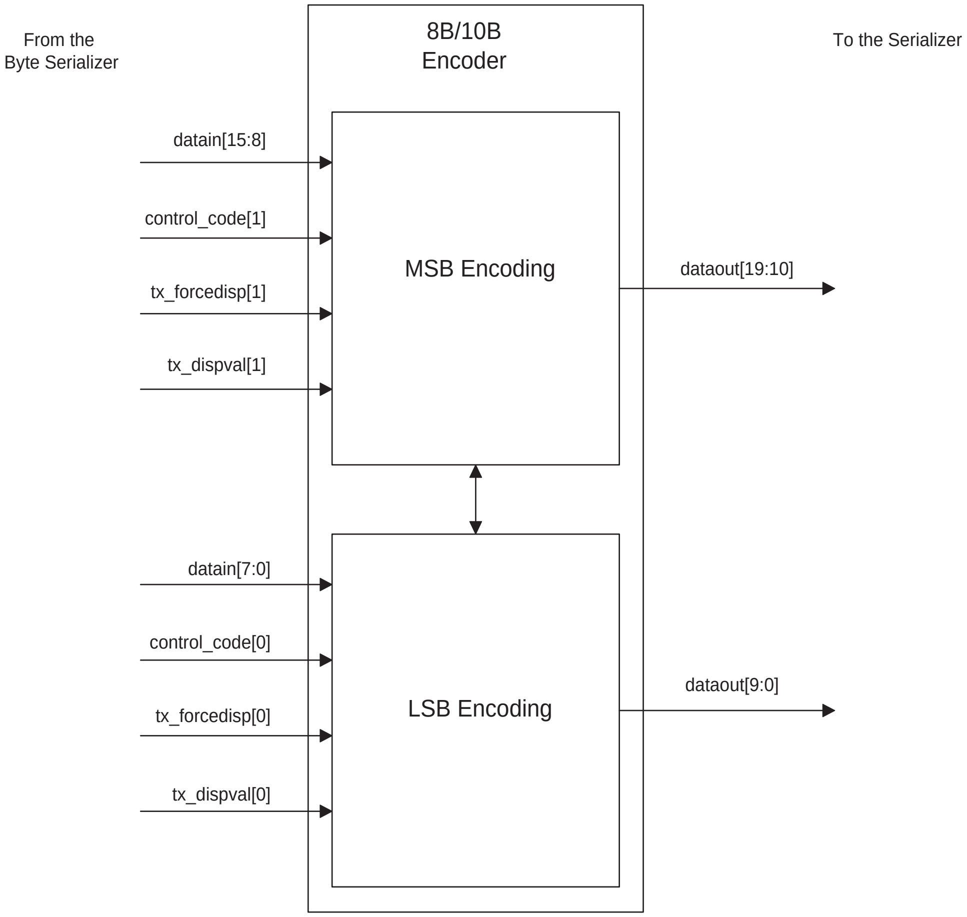
Figure 1–21. 8B/10B Encoder Output during tx_digitalreset Assertion



Double-Width Mode

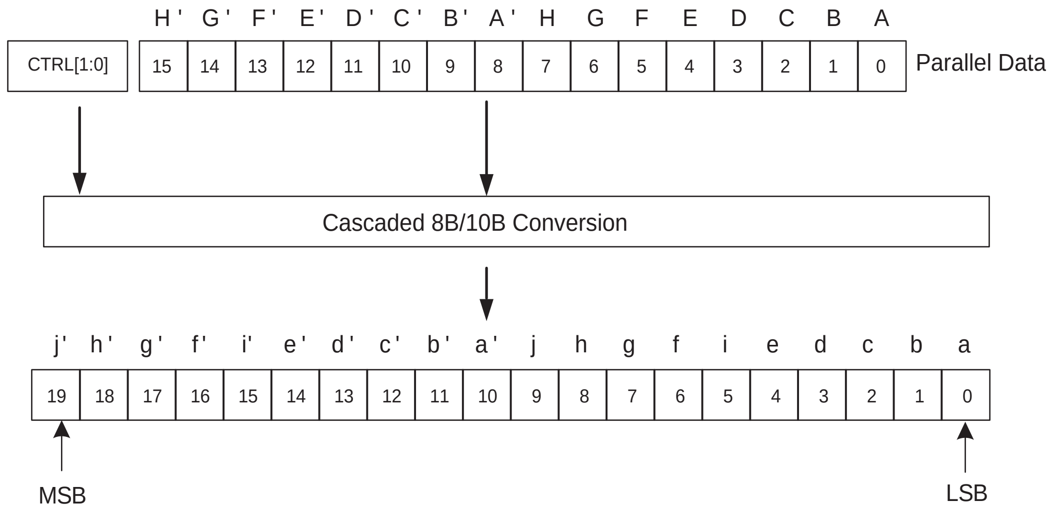
In double-width mode, the 8B/10B encoder operates in a cascaded mode, as shown in Figure 1–22. The LSByte of the input data is encoded and transmitted prior to the MSByte.

Figure 1–22. 8B/10B Encoder in Double-Width Mode



In double-width mode, the cascaded 8B/10B encoder generates two 10-bit code groups from two 8-bit data and their respective control code identifiers. The conversion format is shown in Figure 1–23. The LSB shown in the figure is transmitted first.

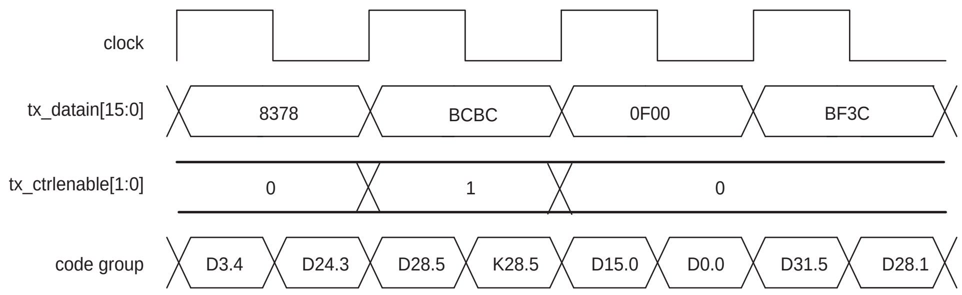
Figure 1–23. 8B/10B Conversion Format in Double-Width Mode


H' G' F' E' D' C' B' A' H G F E D C B A
CTRL[1:0]
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Parallel Data
Cascaded 8B/10B Conversion
j' h' g' f' i' e' d' c' b' a' j h g f i e d c b a
19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
MSB
LSB

Control Code Encoding

In double-width mode, the `tx_ctrlenable[1:0]` port is used to identify which 8-bit data is to be encoded as a control word. The lower bit, `tx_ctrlenable[0]`, is associated with the LSByte; the upper bit, `tx_ctrlenable[1]`, is associated with the MSByte. When `tx_ctrlenable` is low, the byte at the `tx_datain` port of the transceiver is encoded as data (Dx.y); otherwise, it is encoded as a control code (Kx.y). Figure 1–24 shows that only the lower byte of the `tx_datain`[15:0] port is encoded as a control code because `tx_ctrlenable[0]` is high in the second clock cycle.

Figure 1–24. Encoded Control Word and Data Word Transmission



The 8B/10B encoder does not check to see if the code word entered is one of the 12 valid control code groups specified in the IEEE 802.3 8B/10B encoder specification. If an invalid control code is entered, the resultant 10-bit code may be encoded as an invalid code (it does not map to a valid Dx.y or Kx.y code), or unintended valid Dx.y code, depending on the value entered.

The following is an example of an invalid control word encoded into a valid Dx.y code. Take the encoding of an invalid code K24.1 (`tx_datain` = 8'h38 + `tx_ctrl` = 1'b1), depending on the current running disparity, the K24.1 can be encoded to be 10'b0110001100 (0 × 18C), which is equivalent to a D24.6+ (8'hD8 from the RD+ column). An 8B/10B decoder can decode this and not assert any code error flag.

☞ Altera does not recommend sending invalid control words to the 8B/10B encoder.

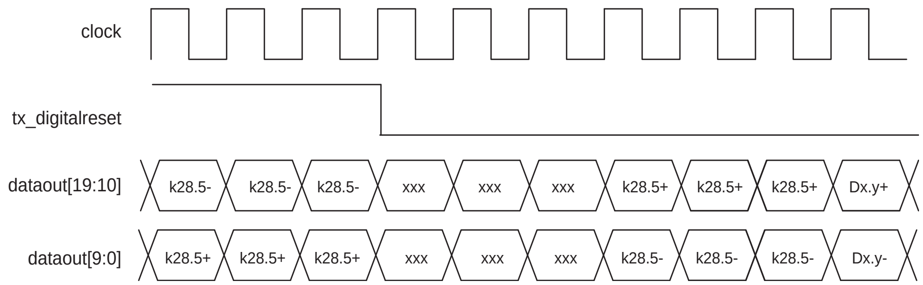
Reset Condition

The `tx_digitalreset` signal resets the 8B/10B encoder. During reset, the running disparity and data registers are cleared. Also, the 8B/10B encoder outputs a K28.5 pattern with proper disparity continuously until `tx_digitalreset` goes low. The inputs from the `tx_datain` and `tx_ctrlenable` ports are ignored during the reset state. After reset, the 8B/10B encoder starts the LSByte with a negative disparity (RD-) and the MSByte with a positive disparity (RD+) and transmits six K28.5 code groups (three on the LSByte and three on the MSByte encoder) for synchronizing before it starts encoding and transmitting data.

☞ If the `tx_digitalreset` signal is asserted, the downstream 8B/10B decoder receiving the data might get synchronization or disparity errors.

Figure 1–25 shows the reset behavior of the 8B/10B encoder. When in reset (`tx_digitalreset` is high), a K28.5- is sent continuously until `tx_digitalreset` is low. Due to some pipelining of the TX channel, there will be some “don’t cares” (10'hxxx) until the first K28.5 is sent (Figure 1–25 shows six “don’t cares”, but the number of “don’t cares” can vary). Both the LSByte and MSByte transmit three K28.5s before the data at the `tx_datain` port is encoded and sent out.

Figure 1–25. Transmitted Output Data when tx_digitalreset is Asserted



Controlling Running Disparity

Upon power on or reset, the 8B/10B encoder has a negative disparity and chooses the 10-bit code from the RD- column (refer to the 8B/10B encoder specification for the RD+ and RD- column values). The ALTGX MegaWizard Plug-In Manager provides the `tx_forcedisp` and `tx_dispval` ports to control the running disparity of the output from the 8B/10B encoder. These ports are available only Basic single-width and Basic double-width modes.

A high value on the `tx_forcedisp` port is the control signal to the disparity value of the output data. The disparity value (RD+ or RD-) is indicated by the value on the `tx_dispval` port. If the `tx_forcedisp` port is low, `tx_dispval` is ignored and the current running disparity is not altered. Forcing disparity can either maintain the current running disparity calculations if the forced disparity value (on the `tx_dispval` bit) happens to match the current running disparity, or flip the current running disparity calculations if it does not. If the forced disparity flips the current running disparity, the downstream 8B/10B decoder might detect a disparity error. Table 1–8 shows the `tx_forcedisp` and `tx_dispval` port values.

Table 1–8. tx_forcedisp and tx_dispval Port Values

tx_forcedisp	tx_dispval	Disparity Value
0	X	Current running disparity has no change
1	0	Encoded data has positive disparity
1	1	Encoded data has negative disparity

Figure 1–26 shows the current running disparity being altered in Basic single-width mode by forcing a positive disparity K28.5 when it was supposed to be a negative disparity K28.5. In this example, a series of K28.5 code groups are continuously being sent. The stream alternates between a positive running disparity (RD+) K28.5 and a negative running disparity (RD-) K28.5 to maintain a neutral overall disparity. The current running disparity at time n + 3 indicates that the K28.5 in time n + 4 should be encoded with a negative disparity. Because `tx_forcedisp` is high at time n + 4, and `tx_dispval` is also high, the K28.5 at time n + 4 is encoded as a positive disparity code group.

Figure 1–26. 8B/10B Encoder Force Running Disparity Operation in Single-Width Mode

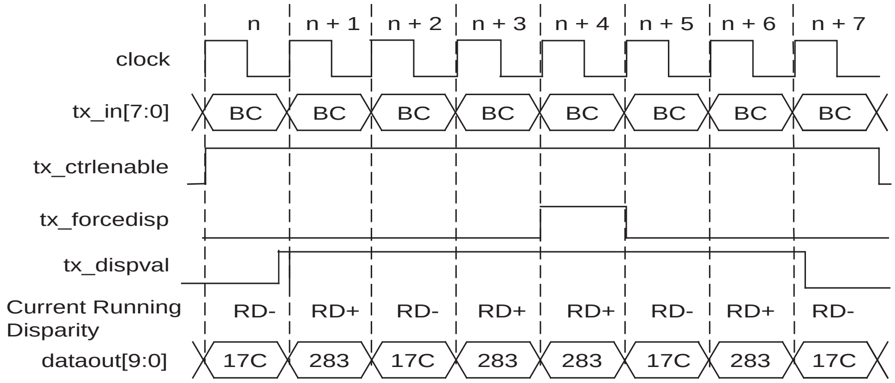
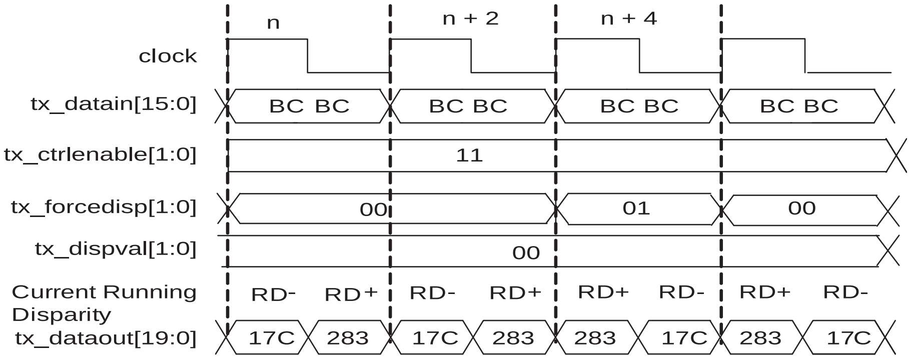


Figure 1–27 shows the current running disparity being altered in Basic double-width mode by forcing a positive disparity on a negative disparity K28.5. In this example, a series of K28.5 are continuously being sent. The stream alternates between a positive ending running disparity (RD+) K28.5 and a negative ending running disparity (RD-) K28.5 as governed by the 8B/10B encoder specification to maintain a neutral overall disparity. The current running disparity at the end of time n + 2 indicates that the K28.5 at the low byte position in time n + 4 should be encoded with a positive disparity. Because `tx_forcedisp` is high at time n + 4, the low signal level of `tx_dispval` is used to convert the lower byte K28.5 to be encoded as a positive disparity code word. As the upper bit of `tx_forcedisp` is low at n + 4, the high byte K28.5 takes the current running disparity from the low byte.

Figure 1–27. 8B/10B Encoder Force Current Running Disparity in Double-Width Mode



Transmitter Polarity Inversion

The positive and negative signals of a serial differential link might accidentally be swapped during board layout. Solutions like a board re-spin or major updates to the logic in the FPGA fabric can be expensive. The transmitter polarity inversion feature is provided to correct this situation. An optional `tx_invpolarity` port is available in all functional modes except (OIF) CEI PHY to dynamically enable the transmitter polarity inversion feature.

In single-width mode, a high value on the `tx_invpolarity` port inverts the polarity of every bit of the 8-bit or 10-bit input data word to the serializer in the transmitter datapath. In double-width mode, a high value on the `tx_invpolarity` port inverts the polarity of every bit of the 16-bit or 20-bit input data word to the serializer in the transmitter datapath. Because inverting the polarity of each bit has the same effect as swapping the positive and negative signals of the differential link, correct data is seen by the receiver. `tx_invpolarity` is a dynamic signal and might cause initial disparity errors at the receiver of an 8B/10B encoded link. The downstream system must be able to tolerate these disparity errors. Figure 1–28 shows the transmitter polarity inversion feature in a single-width 10-bit wide datapath configuration.

Figure 1–28. Transmitter Polarity Inversion in Single-Width Mode

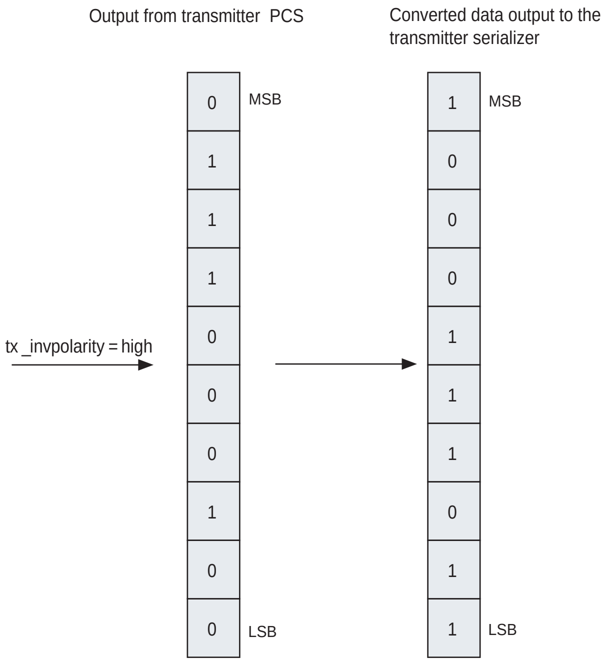
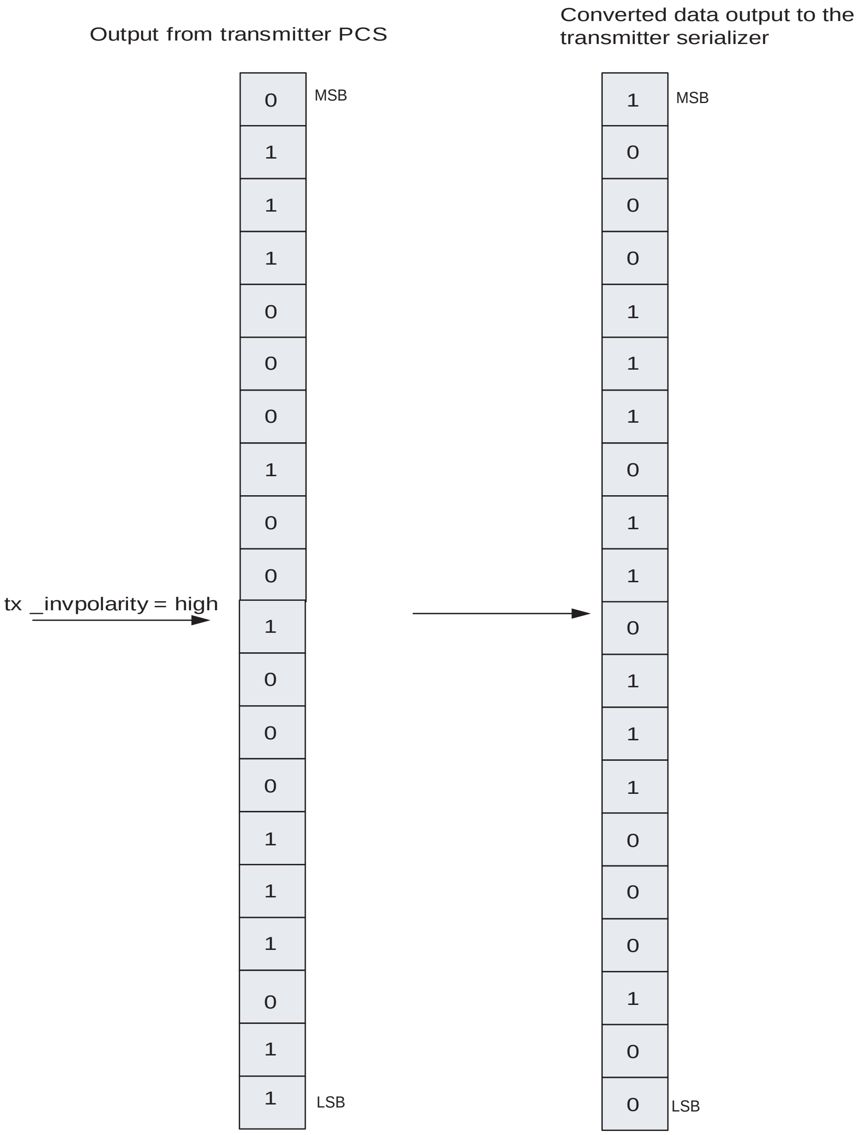

Output from transmitter PCS
Converted data output to the transmitter serializer
tx _invpolarity = high
MSB
LSB
MSB
LSB
0
1
1
1
0
0
0
1
0
0
1
0
0
0
1
1
1
0
1
1

Figure 1–29 shows the transmitter polarity inversion in double-width mode.

Figure 1–29. Transmitter Polarity Inversion in Double-Width Mode


Output from transmitter PCS
Converted data output to the transmitter serializer
MSB
0
1
1
1
0
0
0
1
0
0
1
0
0
0
1
1
1
0
1
1
LSB
tx _invpolarity = high
MSB
1
0
0
0
1
1
1
0
1
1
0
1
1
1
0
0
0
1
0
0
LSB

Transmitter Bit Reversal

By default, the Stratix IV GX transmit bit order is LSBit to MSBit. In single-width mode, the least significant bit of the 8- or 10-bit data word is transmitted first, followed by the most significant bit. In double-width mode, the least significant bit of the 16- or 20-bit data word is transmitted first, followed by the most significant bit. The transmitter bit reversal feature allows reversing the transmit bit order as MSBit to LSBit before it is forwarded to the serializer.

If you enable the transmitter bit reversal feature in Basic single-width mode, the 8-bit `D[7:0]` or 10-bit `D[9:0]` data at the input of the serializer gets rewired to `D[0:7]` or `D[0:9]`, respectively. If you enable the transmitter bit reversal feature in Basic double-width mode, the 16-bit `D[15:0]` or 20-bit `D[19:0]` data at the input of the serializer gets rewired to `D[0:15]` or `D[0:19]`, respectively.

Figure 1–30 shows the transmitter bit reversal feature in Basic single-width for a 10-bit wide datapath configuration. Figure 1–31 shows the transmitter bit reversal feature in basic double-width mode for a 20-bit wide datapath configuration.

Figure 1–30. Transmitter Bit Reversal Operation in Basic Single-Width Mode

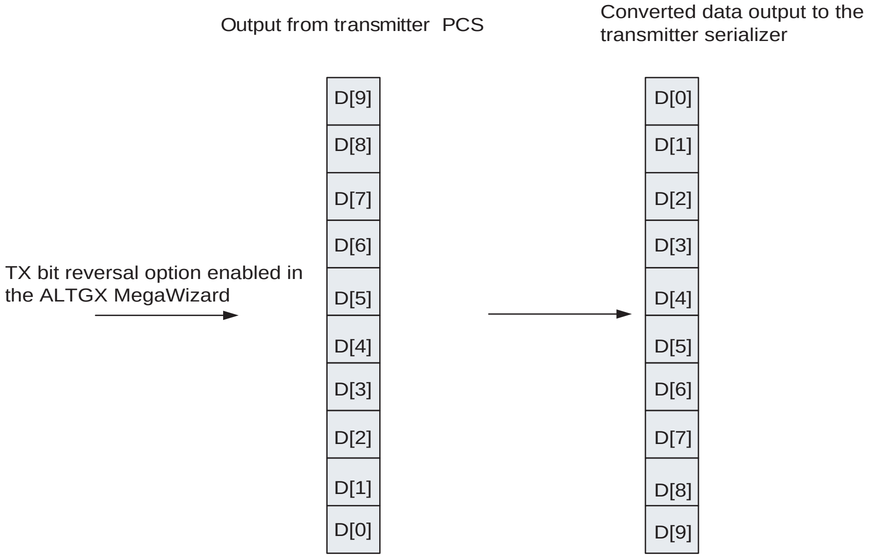
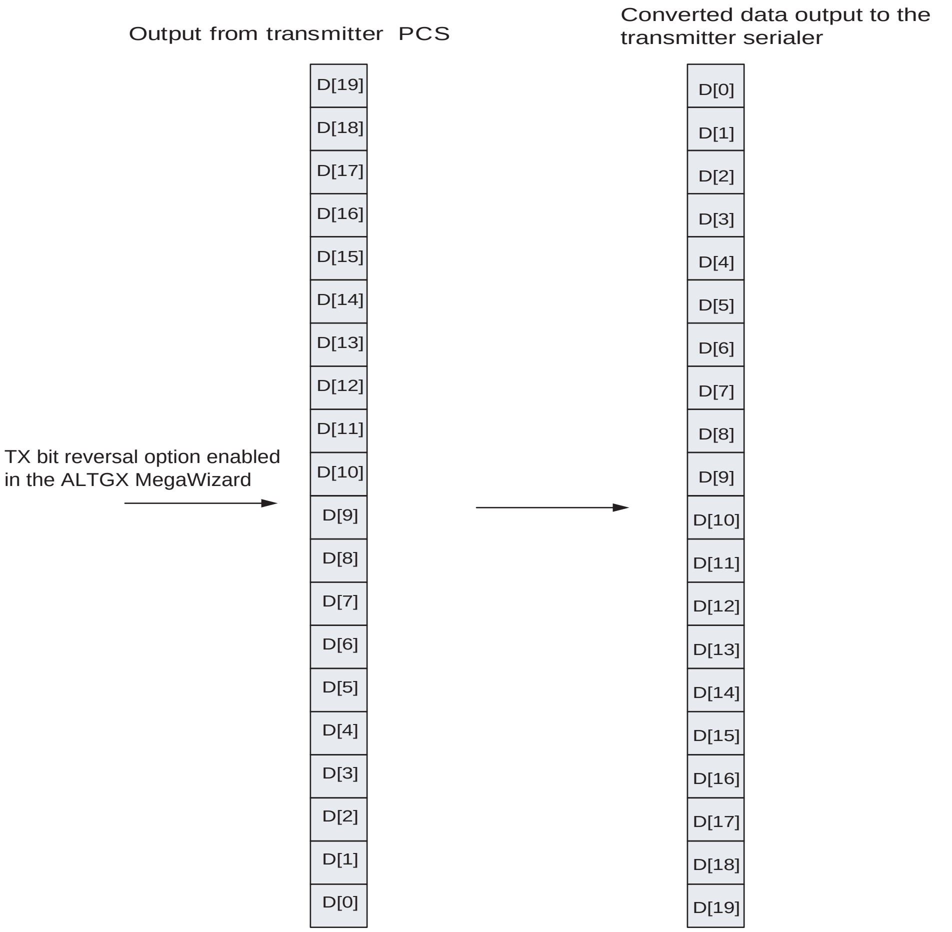

Output from transmitter PCS
Converted data output to the transmitter serializer
TX bit reversal option enabled in the ALTGX MegaWizard
D[9]
D[8]
D[7]
D[6]
D[5]
D[4]
D[3]
D[2]
D[1]
D[0]
D[0]
D[1]
D[2]
D[3]
D[4]
D[5]
D[6]
D[7]
D[8]
D[9]

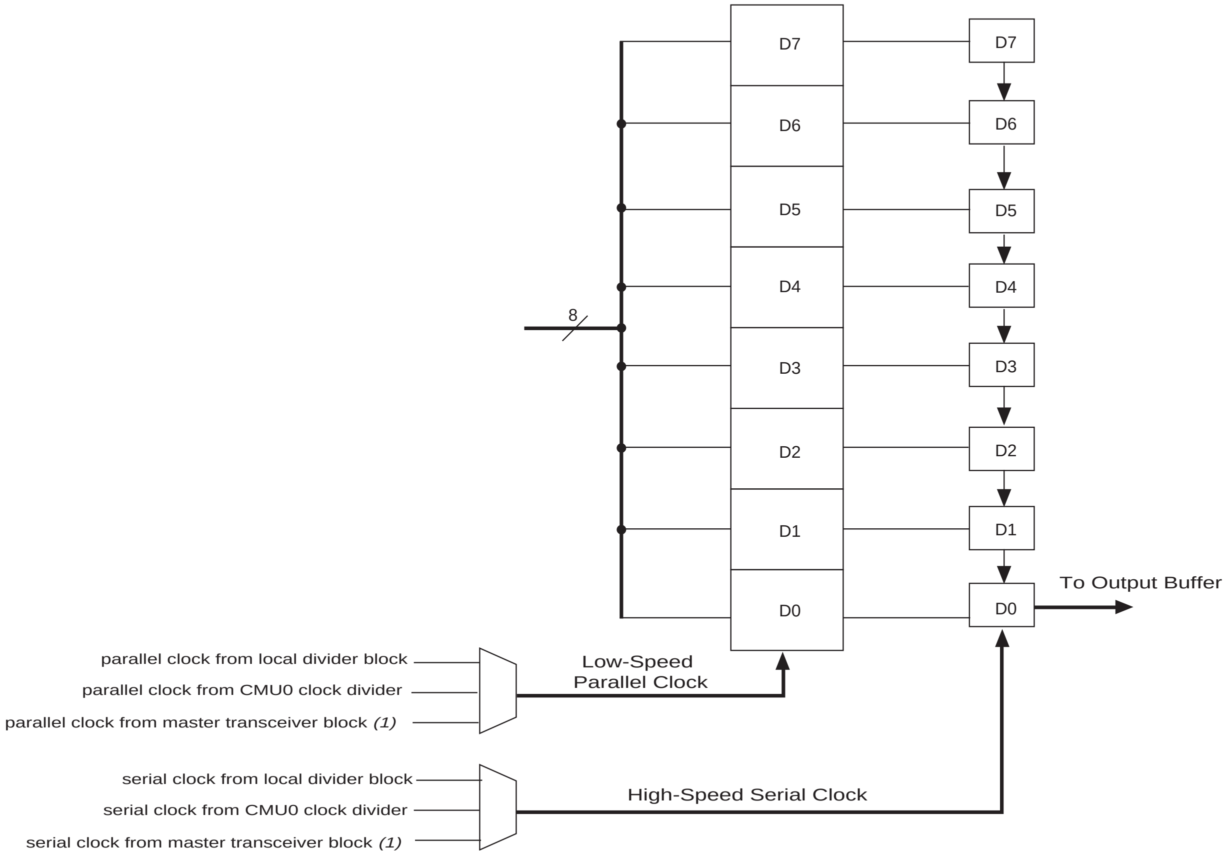
Figure 1–31. Transmitter Bit Reversal Operation in Basic Double-Width Mode



Serializer

The serializer converts the incoming low-speed parallel signal from the transceiver PCS to high-speed serial data and sends it to the transmitter buffer. The serializer supports an 8-bit or 10-bit serialization factor in single-width mode and a 16-bit or 20-bit serialization factor in double-width mode. The serializer block drives the serial data to the output buffer, as shown in Figure 1–32. The serializer block sends out the least significant bit of the input data. Figure 1–33 shows the serial bit order of the serializer block output. In this example, a constant 8'h6A (01101010) value is serialized and the serial data is transmitted from LSBit to MSBit.

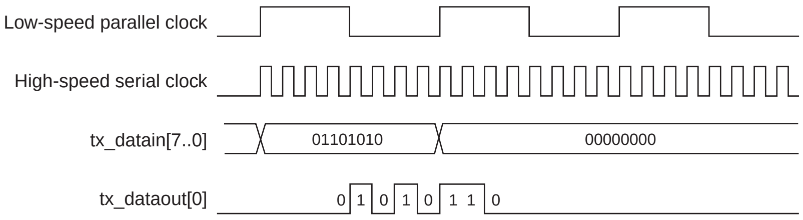
Figure 1–32. Serializer Block in 8-Bit PCS-PMA Interface


D7
D6
D5
D4
D3
D2
D1
D0
8
D7
D6
D5
D4
D3
D2
D1
D0
To Output Buffer
parallel clock from local divider block
parallel clock from CMU0 clock divider
parallel clock from master transceiver block (1)
Low-Speed
Parallel Clock
serial clock from local divider block
serial clock from CMU0 clock divider
serial clock from master transceiver block (1)
High-Speed Serial Clock

Note to Figure 1–32:

(1) The CMU0 clock divider of the master transceiver block provides the clocks. It is used only in PCI Express (PIPE) ×8 mode.

Figure 1–33. Serializer Bit Order *(Note 1)*



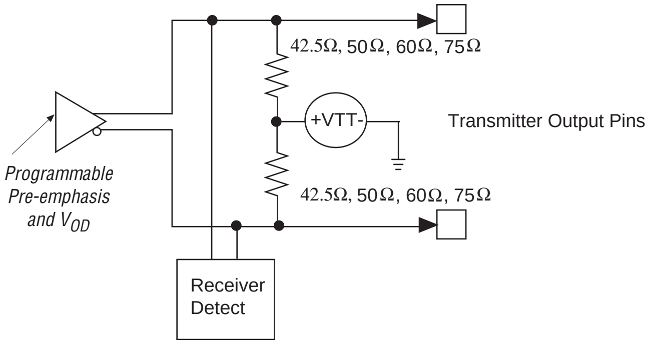
Note to Figure 1–33:

(1) This figure assumes that the input data to the serializer is 8 bits (channel width = 8 bits or 16 bits with the 8B/10B encoder disabled).

Transmitter Output Buffer

The Stratix IV GX transmitter buffers support 1.4-V and 1.5-V pseudo current mode logic (PCML) and can drive 40 inches of FR4 trace across two connectors. You can set the transmitter buffer voltage levels (VCCH) through the ALTGX MegaWizard Plug-In Manager. With the 1.4 V and 1.5 V settings, you can run the transmitter channel from 600 Mbps to 8.5 Gbps and 600 Mbps to 4.25 Gbps, respectively. The transmitter buffer power supply only provides voltage to the transmitter output buffers in the transceiver channels. The transmitter output buffer, as shown in Figure 1–34, has additional circuitry to improve signal integrity, such as VOD, programmable three-tap pre-emphasis circuit, internal termination circuitry, and receiver detect capability to support PCI Express (PIPE) functional mode.

Figure 1–34. Transmitter Output Buffer



Programmable Transmitter Termination

The Stratix IV GX transmitter buffer includes programmable on-chip differential termination of 85 Ω, 100 Ω, 120 Ω, or 150 Ω. The resistance is adjusted by the on-chip calibration circuit in the calibration block (refer to "Calibration Blocks" on page 1–175 for more information), which compensates for temperature, voltage, and process changes. The Stratix IV GX transmitter buffers in the transceiver are current mode drivers. Therefore, the resultant VOD is a function of the transmitter termination value. Refer to "Programmable Output Differential Voltage" on page 1–46 for more information about resultant VOD values.

You can disable OCT and use external termination. If you select external termination, the transmitter common mode is tri-stated. You can set the transmitter termination in the ALTGX MegaWizard Plug-In Manager.

You can also set the OCT through the assignment editor. Set the assignment shown in Table 1–9 to the transmitter serial output pin.

Table 1–9. Stratix IV GX OCT Assignment Settings

Assign To	Transmitter Serial Output Data Pin
Assignment Name:	output termination
Available Values:	OCT 85 Ω, OCT 100 Ω, OCT 120 Ω, OCT 150 Ω

Programmable Output Differential Voltage

The Stratix IV GX device allows you to customize the differential output voltage to handle different trace lengths, various backplanes, and receiver requirements, as shown in Figure 1–35.

Figure 1–35. V_{OD} (Differential) Signal Level

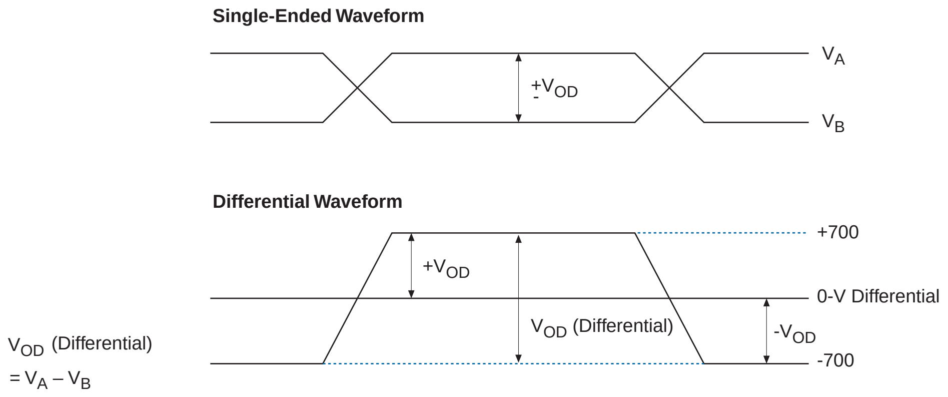


Table 1–10 shows the VOD values for different termination resistor settings.

Table 1–10. Programmable V_{OD} Differential Peak-to-Peak (mV) *(Note 1)*

Values Shown in the ALTGX MegaWizard Plug-In Manager	85 Ω	100 Ω	120 Ω	150 Ω	Unit
0	170	200	240	300	mV
1	340	400	480	600	mV
2	510	600	720	900	mV
3	595	700	840	1050	mV
4	680	800	960	1200	mV
5	765	900	1080	1350	mV
6	850	1000	1200	—	mV
7	1020	1200	—	—	mV

Note to Table 1–10:

(1) These values are preliminary.

Programmable Pre-Emphasis

The programmable pre-emphasis module in each transmit buffer boosts high frequencies in the transmit data signal, which might be attenuated in the transmission media. Using pre-emphasis can maximize the data eye opening at the far-end receiver. The transmission line's transfer function can be represented in the frequency domain as a low-pass filter. Any frequency components below the –3dB frequency can pass through with minimal loss. Frequency components greater than –3dB frequency are attenuated. This variation in frequency response yields data-dependent jitter and other inter symbol interference (ISI) effects. By applying pre-emphasis, the high-frequency components are boosted; that is, pre-emphasized. Pre-emphasis equalizes the frequency response at the receiver so the difference between the low-frequency and high-frequency components are reduced, which minimizes the ISI effects from the transmission medium. Pre-emphasis requirements increase as data rates through legacy backplanes increase. You set the pre-emphasis settings in the ALTGX MegaWizard Plug-In Manager. The ALTGX MegaWizard Plug-In Manager only shows the valid pre-emphasis tap values for a selected VOD and Transmitter Termination resistance setting.

Programmable Transmitter Output Buffer Power (VCCH)

The ALTGX MegaWizard Plug-In Manager provides you an option to select VCCH.

☞ Two options are available for V_{CCH}: 1.4 V or 1.5 V. With 1.4 V, the data rate range is 600 Mbps to 8.5 Gbps; with 1.5 V, the data rate range is 600 Mbps to 4.25 Gbps.

Common Mode Voltage (VCM) Settings

The Stratix IV GX devices provide a VCM of 650 mV.

PCI Express (PIPE) Receiver Detect

The Stratix IV GX transmitter buffer has a built-in receiver detection circuit for use in the PCI Express (PIPE) mode for Gen1 and Gen2 data rates. This circuit detects if there is a receiver downstream by sending out a pulse on the common mode of the transmitter and monitoring the reflection. This mode requires the transmitter buffer to be tri-stated (in Electrical Idle mode), OCT utilization, and a 125 MHz `fixedclk` signal. You can enable this feature in PCI Express (PIPE) mode by setting the `tx_forceelecidle` and `tx_detectrxloopback` ports to 1'b1. The receiver detect circuitry is active only in the P1 power state (refer to the PIPE 2.00 specification for more information about power states).

In the P1 power state, the transmitter output buffer is tri-stated because the transmitter output buffer is in electrical idle. A high on the `tx_detectrxloopback` port triggers the receiver detect circuitry to alter the transmitter output buffer common mode voltage. The sudden change in common mode voltage effectively appears as a step voltage at the tri-stated transmitter buffer output. If a receiver (that complies with PCI Express [PIPE] input impedance requirements) is present at the far end, the time constant of the step voltage is higher. If a receiver is not present or is powered down, the time constant of the step voltage is lower. The receiver detect circuitry snoops the transmitter buffer output for the time constant of the step voltage to detect the presence of the receiver at the far end. A high pulse is driven on the `pipephydonestatus` port and 3'b011 is driven on the `pipestatus` port to indicate that a receiver has been detected. There is some latency after asserting the `tx_detectrxloopback` signal, before the receiver detection is indicated on the `pipephydonestatus` port. For the signal timing to perform the receiver detect operation, refer to Figure 1–98 on page 1–121.

☞ The `tx_forceelecidle` port must be asserted at least 10 parallel clock cycles prior to the `tx_detectrxloopback` port to ensure that the transmitter buffer is tri-stated.

PCI Express Electrical Idle

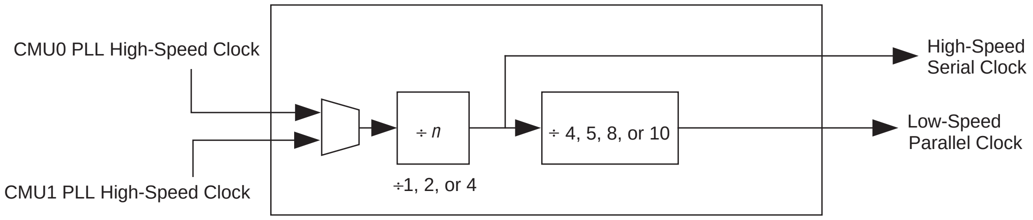
The Stratix IV GX transmitter output buffer supports transmission of PCI Express Electrical Idle (or individual transmitter tri-state). This feature is only active in the PCI Express (PIPE) mode. The `tx_forceelecidle` port puts the transmitter buffer in Electrical Idle mode. This port has a specific functionality in each power state. Refer to PIPE specification 2.00 for use of the `tx_forceelecidle` signal under different power states. For the signal timing to perform the electrical idle transmission in PCI Express (PIPE) mode, refer to Figure 1–97 on page 1–120.

Transmitter Local Clock Divider Block

Each transmitter channel contains a local clock divider block. It receives the high-speed clock from CMU0 PLL or CMU1 PLL and generates the high-speed serial clock for the serializer and the low-speed parallel clock for the transmitter PCS data path. The low-speed parallel clock is also forwarded to the FPGA fabric (`tx_clkout`). The local clock divider block allows each transmitter channel to run at /1, /2, or /4 of the CMU PLL data rate. Note that the local clock divider block is used only in non-bonded functional modes (for example, GIGE, SONET/SDH, and SDI mode).

Figure 1–36 shows the transmitter local clock divider block.

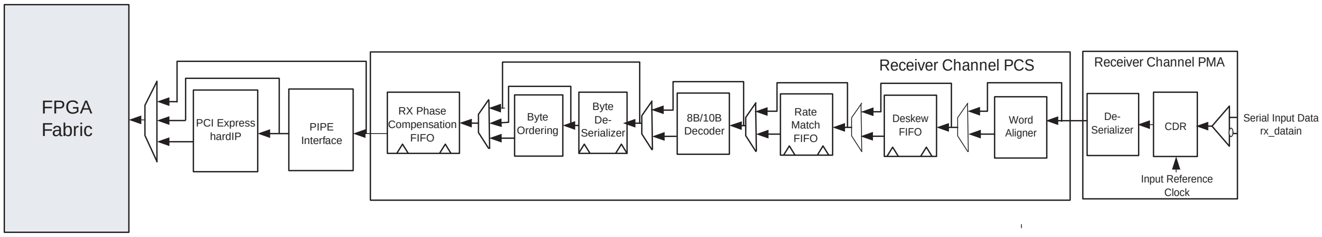
Figure 1–36. Transmitter Local Clock Divider Block



Receiver Channel Datapath

Figure 1–37 shows the receiver channel datapath in Stratix IV GX devices.

Figure 1–37. Receiver Channel Datapath



The receiver channel PMA datapath consists of the following blocks:

- Receiver input buffer
- Clock and data recovery (CDR) unit
- Deserializer

The receiver channel PCS datapath consists of the following blocks:

- Word aligner
- Deskew FIFO
- Rate match (clock rate compensation) FIFO
- 8B/10B decoder
- Byte deserializer
- Byte ordering

- Receiver phase compensation FIFO
- PIPE interface

The receiver datapath is very flexible and allows multiple configurations, depending on the selected functional mode. You can configure the receiver datapath using the ALTGX MegaWizard Plug-In Manager.

This section discusses Stratix IV GX receiver channel datapath architecture. The sub-blocks in the receiver datapath are described in order from the serial receiver input buffer to the receiver phase compensation FIFO buffer at the FPGA fabric—transceiver interface.

Receiver Input Buffer

The receiver input buffer receives serial data from the `rx_datain` port and feeds it to the CDR unit. In the reverse serial loopback (pre-CDR) configuration, it also feeds the received serial data to the transmitter output buffer. Figure 1–38 shows the receiver input buffer.

Figure 1–38. Receiver Input Buffer

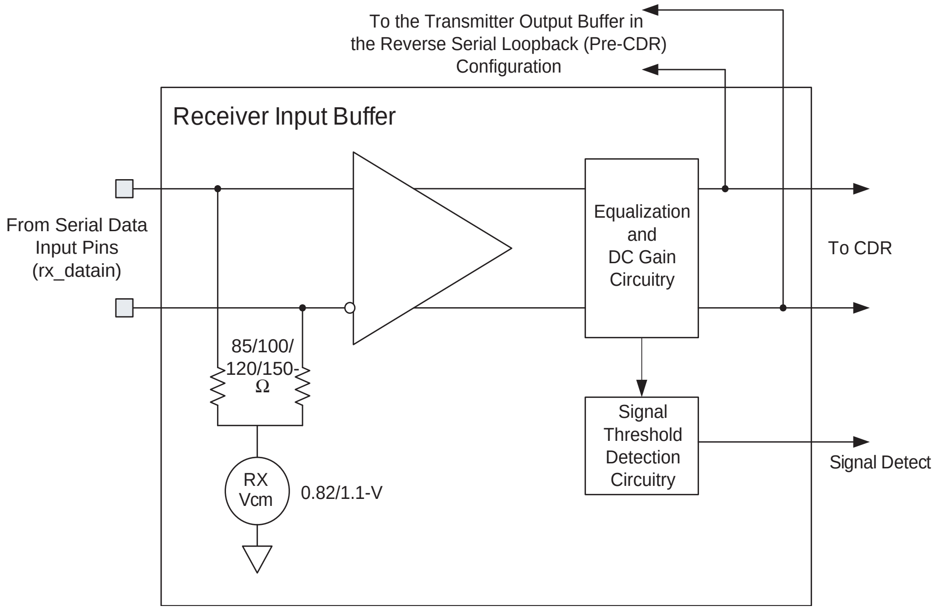


Table 1–11 shows the electrical features supported by the receiver input buffer.

Table 1–11. Electrical Features Supported by the Receiver Input Buffer

Data Rate (Gbps)	I/O Standard	Differential On-Chip Termination with Calibration (Ω)	Common Mode Voltage (V)	Coupling
0.6 – 8.5	1.4 V PCML	85, 100, 120, 150	0.82	AC, DC
	1.5 V PCML	85, 100, 120, 150	0.82	AC, DC
	2.5 V PCML	85, 100, 120, 150	0.82	AC
	LVPECL	85, 100, 120, 150	0.82	AC
	LVDS	85, 100, 120, 150	1.1	AC, DC

The Stratix IV GX receiver buffer supports the following features:

- Programmable differential OCT
- Programmable common mode voltage
- AC and DC coupling
- Programmable equalization and DC gain
- Signal threshold detection circuitry

Programmable Differential On-Chip Termination

The Stratix IV GX receiver buffers support optional differential on-chip termination resistors of 85, 100, 120, and 150 Ω. To select the desired receiver OCT resistor, make the assignments shown in Table 1–12 in the Quartus II software Assignment Editor.

Table 1–12. Stratix IV GX Receiver On-Chip Termination Assignment Settings

Assign To	rx_datain (Receiver Input Data Pins)
Assignment Name:	Input Termination
Available Values:	OCT 85 Ω, OCT 100 Ω, OCT 120 Ω, OCT 150 Ω, Off

☞ The Stratix IV GX receiver OCT resistors have calibration support to compensate for process, voltage, and temperature variations. For more information about OCT calibration support, refer to "Calibration Blocks" on page 1–175.

Programmable Common Mode Voltage

The Stratix IV GX receiver buffers have on-chip biasing circuitry to establish the required common mode voltage at the receiver input. It supports two common mode voltage settings of 0.82 V and 1.1 V that you can select in the ALTGX MegaWizard Plug-In Manager.

You must select 0.82 V as the receiver buffer common mode voltage for the following receiver input buffer I/O standards:

- 1.4-V PCML
- 1.5-V PCML
- 2.5-V PCML
- LVPECL

You must select 1.1 V as the receiver buffer common mode voltage for the following receiver input buffer I/O standard:

- LVDS

☞ On-chip biasing circuitry is effective only if you select on-chip receiver termination. If you select external termination, you must implement off-chip biasing circuitry to establish the common mode voltage at the receiver input buffer.

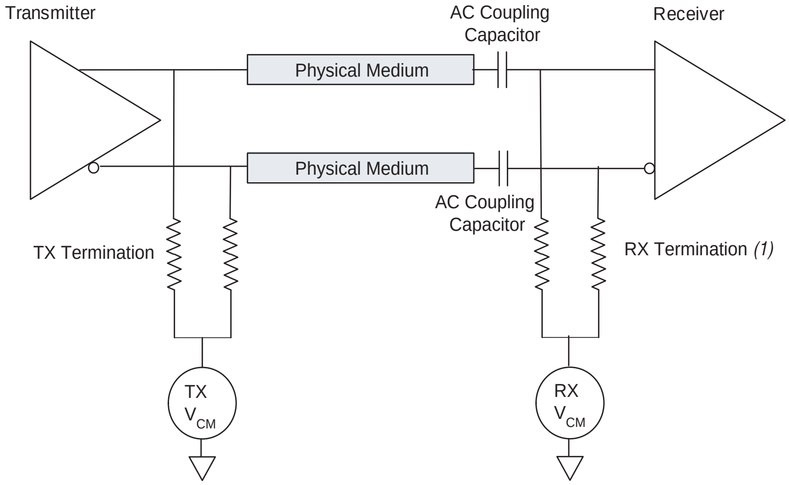
Link Coupling

A high-speed serial link can either be AC-coupled or DC-coupled, depending on the serial protocol being implemented. While most of the serial protocols require links to be AC-coupled, protocols like Common Electrical I/O (CEI) optionally allow DC coupling.

AC-Coupled Links

In an AC-coupled link, the AC coupling capacitor blocks the transmitter DC common mode voltage. The on-chip or off-chip receiver termination and biasing circuitry automatically restores the selected common mode voltage. Figure 1–39 shows an AC-coupled link.

Figure 1–39. AC-Coupled Link



Note to Figure 1–39:

(1) The receiver termination and biasing can be on-chip or off-chip.

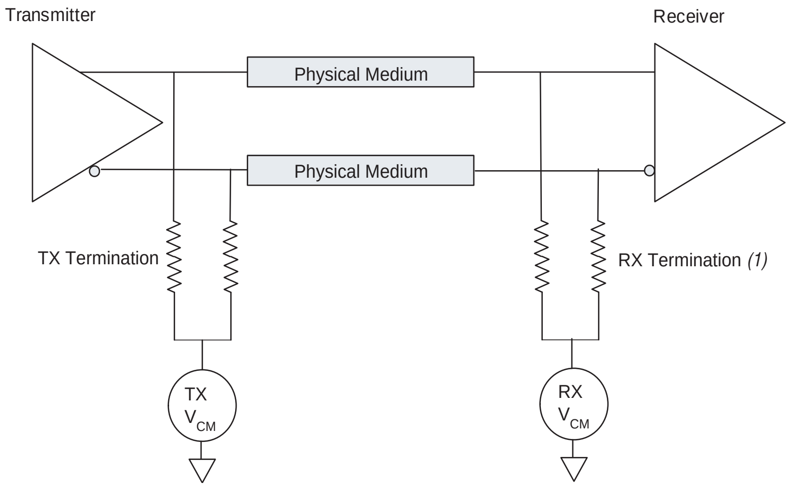
The following protocols supported by Stratix IV GX devices mandate AC-coupled links:

- PCI Express (PIPE)
- Gigabit Ethernet
- Serial RapidIO
- XAUI
- SDI

DC-Coupled Links

In a DC-coupled link, the transmitter DC common mode voltage is seen unblocked at the receiver buffer. The link common mode voltage depends on the transmitter common mode voltage and the receiver common mode voltage. The on-chip or off-chip receiver termination and biasing circuitry must ensure compatibility between the transmitter and the receiver common mode voltage. Figure 1–40 shows a DC-coupled link.

Figure 1–40. DC-Coupled Link



Note to Figure 1–39:

(1) The receiver termination and biasing can be on-chip or off-chip.

You might choose to use the DC-coupled high-speed link for the following functional modes only:

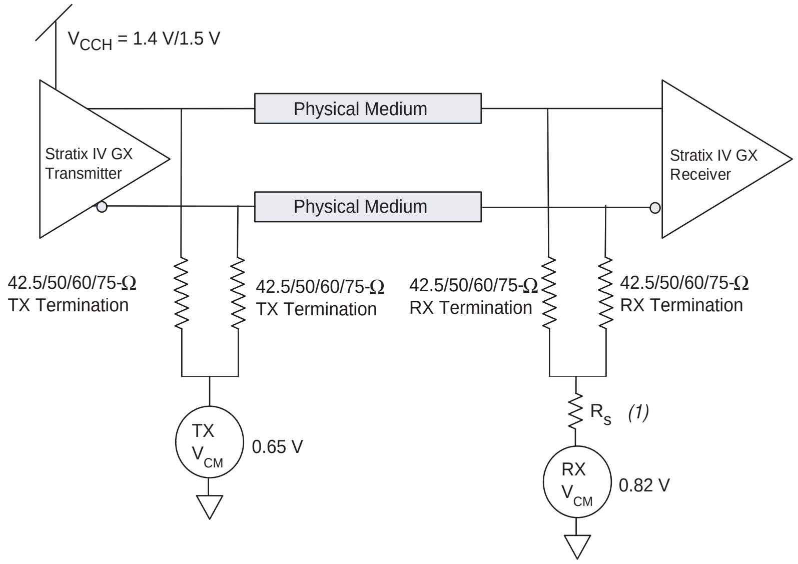
- Basic single- and double-width
- (OIF) CEI PHY interface

The following sections discuss DC-coupling requirements for a high-speed link with a Stratix IV GX device used as the transmitter, receiver, or both. Specifically, the following link configurations are discussed:

- Stratix IV GX Transmitter (PCML) to Stratix IV GX Receiver (PCML)
- Stratix II GX Transmitter (PCML) to Stratix IV GX Receiver (PCML)
- Stratix IV GX Transmitter (PCML) to Stratix II GX Receiver (PCML)
- LVDS Transmitter to Stratix IV GX Receiver (PCML)

Figure 1–41 shows a typical Stratix IV GX transmitter (PCML) to Stratix IV GX Receiver (PCML) DC coupled link.

Figure 1–41. Stratix IV GX Transmitter (PCML) to Stratix IV GX Receiver (PCML)


V_{CCH} = 1.4 V/1.5 V
Physical Medium
Physical Medium
Stratix IV GX Transmitter
Stratix IV GX Receiver
42.5/50/60/75-Ω TX Termination
42.5/50/60/75-Ω TX Termination
42.5/50/60/75-Ω RX Termination
42.5/50/60/75-Ω RX Termination
R_S (1)
TX V_{CM}
0.65 V
RX V_{CM}
0.82 V

Note to Figure 1–41:

(1) R_S is the parasitic resistance present in the on-chip RX termination and biasing circuitry

Table 1–13 shows the allowed transmitter and receiver settings in a Stratix IV GX transmitter (PCML) to Stratix IV GX receiver (PCML) DC coupled link.

Table 1–13. Settings for a Stratix IV GX Transmitter (PCML) to Stratix IV GX Receiver (PCML) DC Coupled Link

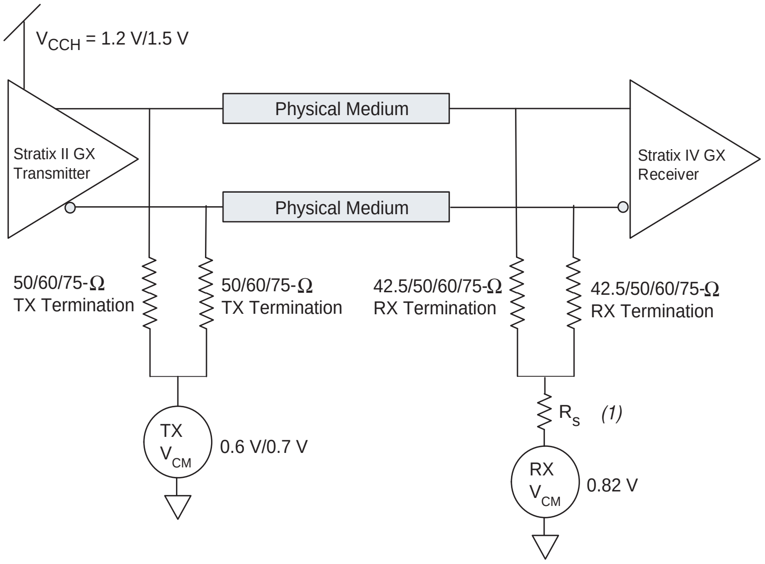
Transmitter (Stratix IV GX) Settings				Receiver (Stratix IV GX) Settings		
Data Rate	**VCCH *(1)***	**TX VCM *(1)***	**Differential Termination**	**Data Rate**	**RX VCM**	**Differential Termination**
600-8500 Mbps	1.4 V/1.5 V	0.65 V	85/100/120/150-Ω	600-8500 Mbps	0.82 V	85/100/120/150-Ω

Note to Table 1–13:

(1) VCCH = 1.5 V can support data rates from 600 Mbps to 4250 Mbps. VCCH = 1.4 V can support data rates from 600 Mbps to 8500 Mbps.

Figure 1–42 shows the Stratix II GX transmitter (PCML) to Stratix IV GX receiver (PCML) coupled link.

Figure 1–42. Stratix II GX Transmitter (PCML) to Stratix IV GX Receiver (PCML)


V_{CCH} = 1.2 V/1.5 V
Physical Medium
Physical Medium
Stratix II GX Transmitter
Stratix IV GX Receiver
50/60/75-Ω TX Termination
50/60/75-Ω TX Termination
42.5/50/60/75-Ω RX Termination
42.5/50/60/75-Ω RX Termination
R_S (1)
TX V_{CM}
0.6 V/0.7 V
RX V_{CM}
0.82 V

Note to Figure 1–42:

(1) R_S is the parasitic resistance present in the on-chip RX termination and biasing circuitry.

Table 1–14 shows the allowed transmitter and receiver settings in a Stratix II GX to Stratix IV GX DC coupled link.

Table 1–14. Settings for a Stratix II GX to Stratix IV GX DC Coupled Link

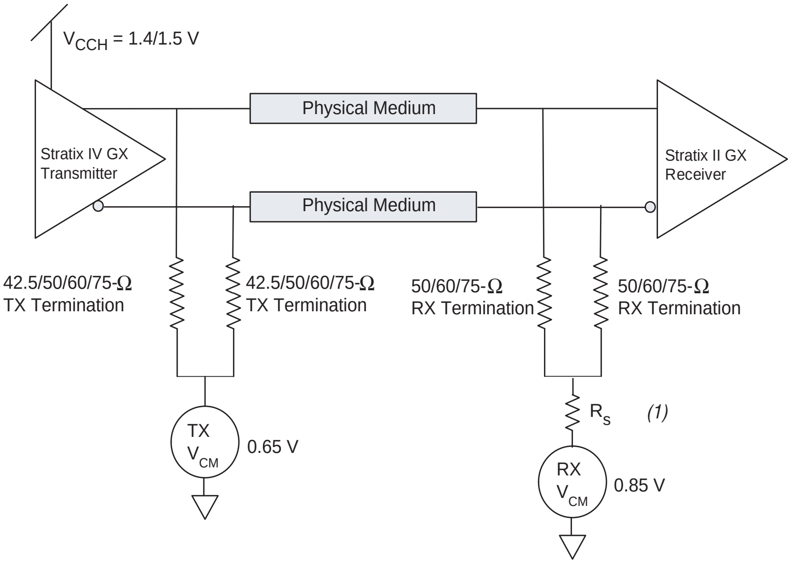
Transmitter (Stratix II GX) Settings				Receiver (Stratix IV GX) Settings		
Data Rate	**VCCH *(1)***	**TX VCM *(1)***	**Differential Termination**	**Data Rate**	**RX VCM**	**Differential Termination**
600-6375 Mbps	1.5 V (1.5 V PCML)	0.6 V/0.7 V	100/120/150-Ω	600-6375 Mbps	0.82 V	100/120/150-Ω

Note to Table 1–14:

(1) VCCH = 1.5 V with TX Vcm = 0.7 V can support data rates from 600 Mbps to 3125 Mbps. VCCH = 1.5 V with TX Vcm = 0.6 V can support data rates from 600 Mbps to 6375 Mbps.

Figure 1–43 shows the Stratix IV GX transmitter (PCML) to Stratix II GX receiver (PCML) coupled link.

Figure 1–43. Stratix IV GX Transmitter (PCML) to Stratix II GX Receiver (PCML)


VCCH = 1.4/1.5 V
Stratix IV GX Transmitter
Physical Medium
Physical Medium
Stratix II GX Receiver
42.5/50/60/75-Ω TX Termination
42.5/50/60/75-Ω TX Termination
50/60/75-Ω RX Termination
50/60/75-Ω RX Termination
TX VCM
0.65 V
RS
(1)
RX VCM
0.85 V

Note to Figure 1–43:

(1) R_S is the parasitic resistance present in the on-chip RX termination and biasing circuitry.

Table 1–15 shows the allowed transmitter and receiver settings in a Stratix IV GX transmitter (PCML) to Stratix II GX receiver (PCML) DC coupled link.

Table 1–15. Settings for a Stratix IV GX to Stratix II GX DC Coupled Link

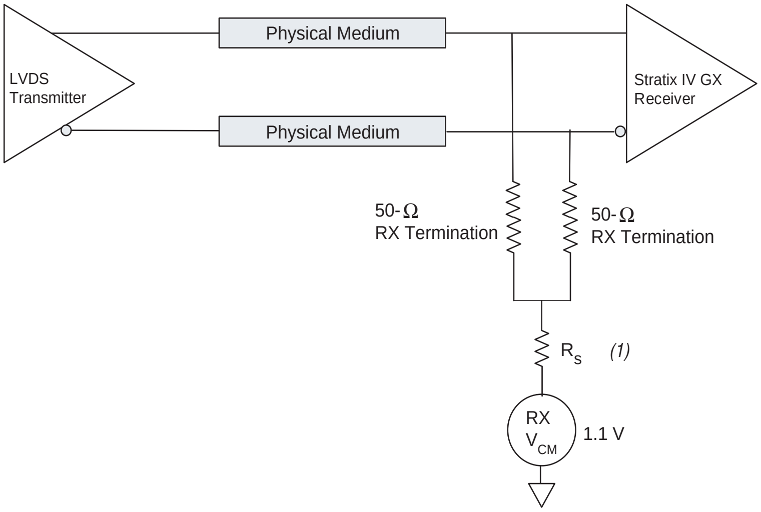
Transmitter (Stratix IV GX) Settings				Receiver (Stratix II GX) Settings			
Data Rate	**VCCH *(1)***	**TX VCM**	**Differential Termination**	**Data Rate**	**I/O Standard**	**RX VCM**	**Differential Termination**
600-6375 Mbps	1.4/1.5 V	0.65 V	100/120/150-Ω	600-6375 Mbps	1.4/1.5 V PCML	0.85 V	100/120/150-Ω

Note to Table 1–15:

(1) VCCH = 1.5 V can support data rates from 600 Mbps to 4250 Mbps. VCCH = 1.4 V can support data rates from 600 Mbps to 6375 Mbps.

Figure 1–44 shows the LVDS transmitter to Stratix IV GX receiver (PCML) coupled link.

Figure 1–44. LVDS Transmitter to Stratix IV GX Receiver (PCML)



Note to Figure 1–44:

(1) R$_S$ is the parasitic resistance present in the on-chip RX termination and biasing circuitry.

Table 1–16 shows the allowed transmitter and receiver settings in a LVDS transmitter to Stratix IV GX receiver DC coupled link.

Table 1–16. Settings for a LVDS transmitter to Stratix IV GX Receiver DC Coupled Link *(Note 1)*

Receiver (Stratix IV GX) Settings		
RX VCM	**Differential Termination**	**R$_S$**
1.1 V	100-Ω	*(2)*

Note to Table 1–16:

(1) When DC coupling an LVDS transmitter to the Stratix IV GX receiver, use RX Vcm = 1.1 V and series resistance value RS to verify compliance to the LVDS specification.

(2) Pending characterization.

Programmable Equalization and DC Gain

The transfer function of the physical medium can be represented as a low-pass filter in the frequency domain. Frequency components below –3 dB frequency pass through with minimal loss. Frequency components greater than –3 dB frequency get attenuated as a function of frequency due to skin-effect and dielectric losses. This variation in frequency response yields data-dependant jitter and other ISI effects, which can cause incorrect sampling of the input data.

Each Stratix IV GX receiver buffer has an independently programmable equalization circuitry that boosts the high-frequency gain of the incoming signal, thereby compensating for the low-pass filter effects of the physical medium. The amount of high-frequency gain required depends on the loss characteristics of the physical medium. The Stratix IV GX equalization circuitry supports 16 equalization settings that provide up to 16 dB of high-frequency boost. You can select the appropriate equalization setting in the ALTGX MegaWizard Plug-In Manager.

The Stratix IV GX receiver buffer also supports programmable DC gain circuitry. Unlike equalization circuitry, the DC gain circuitry provides equal boost to the incoming signal across the frequency spectrum. The receiver buffer supports DC gain settings of 0, 3, 6, 9, and 12 dB. You can select the appropriate DC gain setting in the ALTGX MegaWizard Plug-In Manager.

Signal Threshold Detection Circuitry

In PCI Express (PIPE) mode, you can enable the optional signal threshold detection circuitry by NOT selecting the **Force signal detection** option in the ALTGX MegaWizard Plug-In Manager. If enabled, this option senses whether the signal level present at the receiver input buffer is above the signal detect threshold voltage that you specified in the **What is the signal detect and signal loss threshold?** option in the ALTGX MegaWizard Plug-In Manager.

☞ The appropriate signal detect threshold level that complies with the PCI Express compliance parameter VRX-IDLE-DETDIFFp-p is pending characterization.

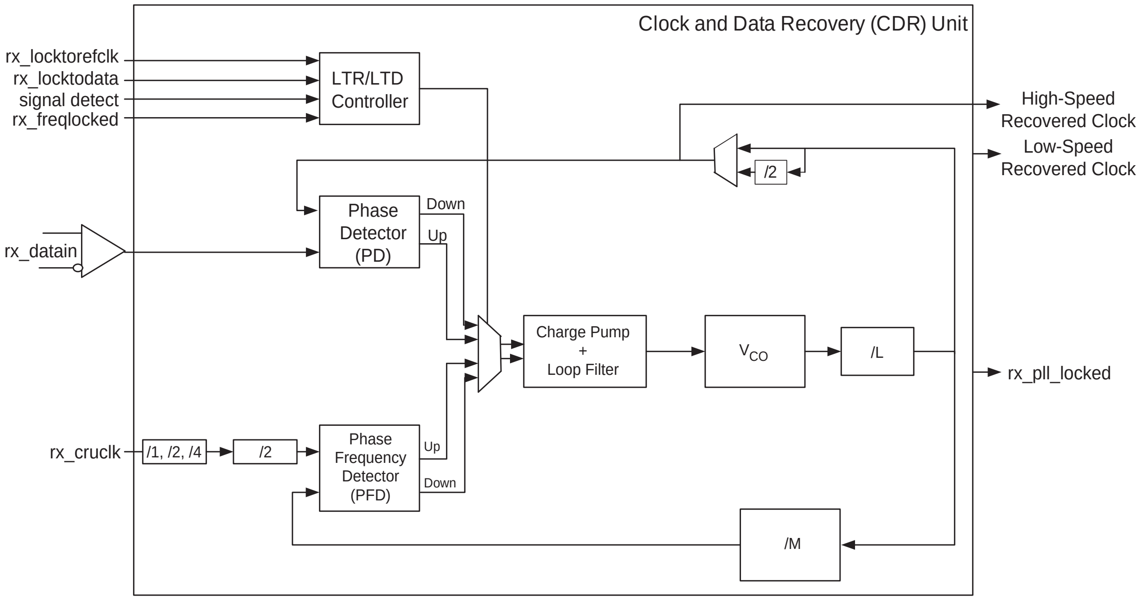
The signal threshold detection circuitry has a hysteresis response that filters out any high-frequency ringing caused by inter-symbol interference or high-frequency losses in the transmission medium. If the signal threshold detection circuitry senses the signal level present at the receiver input buffer to be higher than the signal detect threshold, it asserts the `rx_signaldetect` signal high. Otherwise, the signal threshold detection circuitry de-asserts the `rx_signaldetect` signal low. If you select the **Force signal detection** option in the ALTGX MegaWizard Plug-In Manager, `rx_signaldetect` is always asserted high, irrespective of the signal level on the receiver input buffer.

The `rx_signaldetect` signal is also used by the lock-to-reference/lock-to-data (LTR/LTD) controller in the receiver CDR to switch between LTR and LTD lock modes. When the signal threshold detection circuitry de-asserts the `rx_signaldetect` signal, the LTR/LTD controller switches the receiver CDR from LTD to LTR lock mode. For more information, refer to "LTR/LTD Controller" on page 1–62.

Clock and Data Recovery Unit

Each Stratix IV GX receiver channel has an independent CDR unit to recover the clock from the incoming serial data stream. The high-speed and low-speed recovered clocks are used to clock the receiver PMA and PCS blocks. Figure 1–45 shows the CDR block diagram.

Figure 1–45. Clock and Data Recovery Unit



The CDR operates either in LTR mode or LTD mode. In LTR mode, the CDR tracks the input reference clock. In LTD mode, the CDR tracks the incoming serial data.

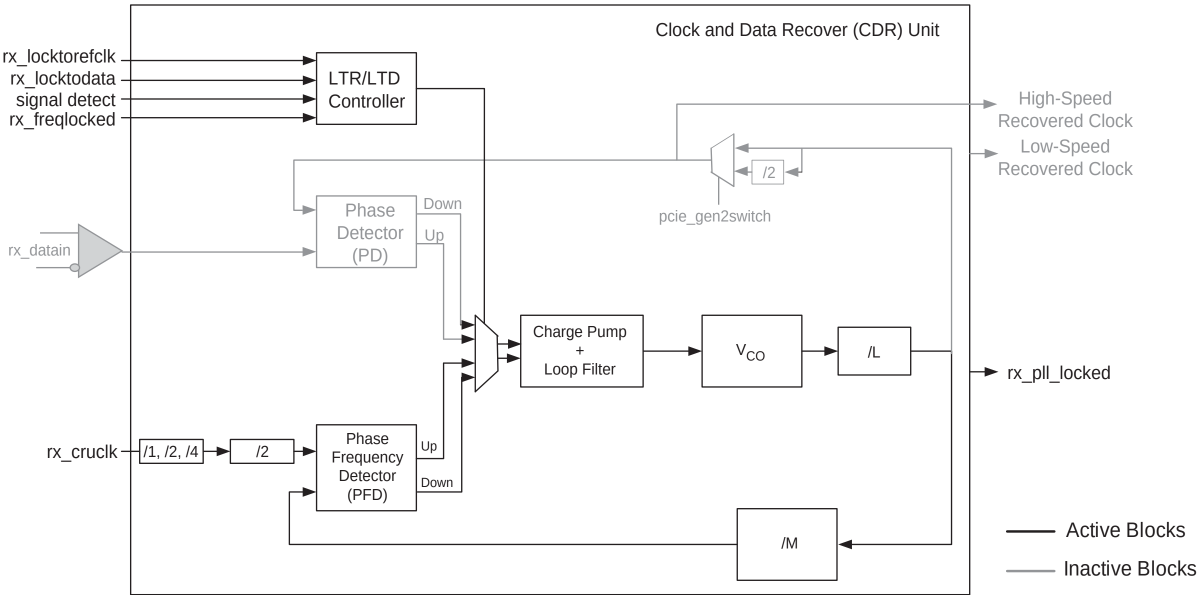
After receiver power-up and reset cycle, the CDR must be kept in LTR mode until it locks to the input reference clock. Once locked to the input reference clock, the CDR output clock is trained to the configured data rate. The CDR can now switch to LTD mode to recover the clock from incoming data. The LTR/LTD controller controls the switch between LTR and LTD modes.

Lock-to-Reference Mode

In LTR mode, the phase frequency detector in the CDR tracks the receiver input reference clock, `rx_cruclk`. The PFD controls the charge pump that tunes the VCO in the CDR. Depending on the data rate and the selected input reference clock frequency, the Quartus II software automatically selects the appropriate /M and /L divider values such that the CDR output clock frequency is half the data rate. An active high, the `rx_pll_locked` status signal is asserted to indicate that the CDR has locked to phase and frequency of the receiver input reference clock. Figure 1–46 shows active blocks when CDR is in LTR mode.

☞ The phase detector (PD) is inactive in LTR mode.

Figure 1–46. CDR in Lock-To-Reference Mode



You can drive the receiver input reference clock with the following clock sources:

- Dedicated `refclk` pins (`refclk0` and `refclk1`) of the associated transceiver block
- Inter-transceiver block clock lines from other transceiver blocks on the same side of the device (up to six ITB clock lines, two from each transceiver block)
- Global PLD clock driven by a dedicated clock input pin
- Clock output from the left and right PLLs in the FPGA fabric

Table 1–17 shows CDR specifications in LTR mode.

Table 1–17. CDR Specification in Lock-To-Reference Mode

Parameter	Value
Input Reference Clock Frequency	50 MHz – 637.5 MHz *(1)*
PFD Input Frequency	50 MHz – 325 MHz
/M Divider	4, 5, 8, 10, 16, 20, 25
/L Divider	1, 2, 4, 8

Note to Table 1–17:

(1) Reference clock frequency greater than 622.08 is available only in (OIF) CEI PHY Interface mode.

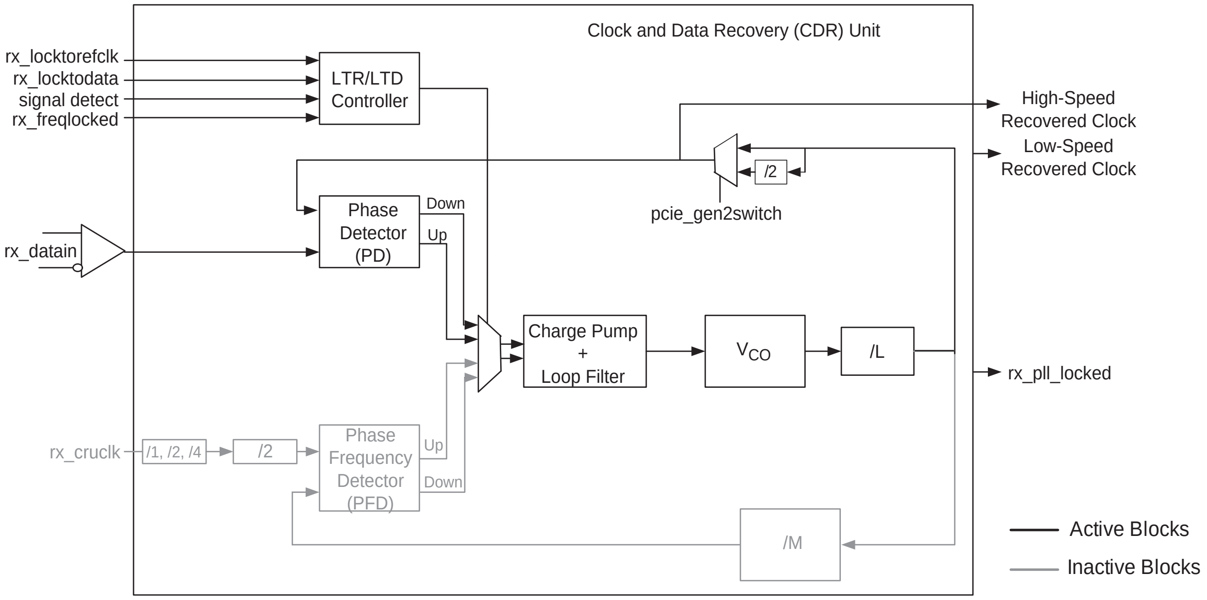
For input reference clock frequencies greater than 325 MHz, the Quartus II software automatically selects the appropriate /1, /2, /4 pre-divider to meet the PFD input frequency limitation of 325 MHz.

Lock-to-Data Mode

The CDR must be in LTD mode to recover clock from the incoming serial data during normal operation. In LTD mode, the phase detector (PD) in the CDR tracks the incoming serial data at the receiver buffer. Depending on the phase difference between the incoming data and the CDR output clock, the PD controls the CDR charge pump that tunes the VCO. Figure 1–47 shows active blocks when the CDR is in LTD mode.

The PFD is inactive in LTD mode. The `rx_pll_locked` signal toggles randomly and has no significance in LTD mode.

Figure 1–47. CDR in Lock-To-Data Mode



After switching to LTD mode, it can take a maximum of 1 ms for the CDR to get locked to the incoming data and produce a stable recovered clock. The actual lock time depends on the transition density of the incoming data and the PPM difference between the receiver input reference clock and the upstream transmitter reference clock. The receiver PCS logic must be held in reset until the CDR produces a stable recovered clock.

For more information about receiver reset recommendations, refer to the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook*.

PCI Express (PIPE) Clock Switch Circuitry

The feedback path from the CDR VCO to the PD has a /2 divider that is used in PCI Express (PIPE) mode configured at Gen2 (5 Gbps) data rate for dynamic switch between Gen1 (2.5 Gbps) and Gen2 (5 Gbps) signaling rates. When the PHY-MAC layer instructs a Gen2-to-Gen1 signaling rate switch, the /2 divider gets enabled. When the PHY-MAC layer instructs a Gen1-to-Gen2 signaling rate switch, the /2 divider gets disabled. For details about the PCI Express (PIPE) signaling rate switch, refer to "Dynamic Switch Between Gen1 (2.5 Gbps) and Gen2 (5 Gbps) Signaling Rate" on page 1–127.

☞ The /2 divider in the receiver CDR between the VCO and the PD is disabled in all other functional modes.

LTR/LTD Controller

The LTR/LTD controller controls whether the CDR is in LTR or LTD mode. You can configure the LTR/LTD controller either in automatic lock mode or manual lock mode.

Two optional input ports (`rx_locktorefclk` and `rx_locktodata`) allow you to configure the LTR/LTD controller in either automatic lock mode or manual lock mode. Table 1–18 shows the relationship between these optional input ports and the LTR/LTD controller lock mode.

Table 1–18. Optional Input Ports and LTR/LTD Controller Lock Mode

rx_locktorefclk	rx_locktodata	LTR/LTD Controller Lock Mode
1	0	Manual – LTR Mode
x	1	Manual – LTD Mode
0	0	Automatic Lock Mode

☞ If you do not instantiate the optional `rx_locktorefclk` and `rx_locktodata` signals, the Quartus II software automatically configures the LTR/LTD controller in automatic lock mode.

Automatic Lock Mode

In automatic lock mode, the LTR/LTD controller initially sets the CDR to lock to the input reference clock (LTR mode). After the CDR locks to the input reference clock, the LTR/LTD controller automatically sets it to lock to the incoming serial data (LTD mode) when the following three conditions are met:

- Signal threshold detection circuitry indicates the presence of valid signal levels at the receiver input buffer
- The CDR output clock is within the configured PPM frequency threshold setting with respect to the input reference clock (frequency locked)
- The CDR output clock and the input reference clock are phase matched within approximately 0.08 UI (phase locked)

The switch from LTR to LTD mode is indicated by the assertion of the `rx_freqlocked` signal.

In LTD mode, the CDR uses a phase detector to keep the recovered clock phase-matched to the data. If the CDR does not stay locked to data due to frequency drift or severe amplitude attenuation, the LTR/LTD controller switches the CDR back to LTR mode to lock to the input reference clock. In automatic lock mode, the LTR/LTD controller switches the CDR from LTD to LTR mode when the following conditions are met:

- Signal threshold detection circuitry indicates the absence of valid signal levels at the receiver input buffer
- The CDR output clock is not within the configured PPM frequency threshold setting with respect to the input reference clock

The switch from LTD to LTR mode is indicated by the deassertion of the `rx_freqlocked` signal.

Manual Lock Mode

In automatic lock mode, the LTR/LTD controller relies on the PPM detector and the phase relationship detector to set the CDR in LTR or LTD mode. The PPM detector and phase relationship detector reaction times can be too long for some applications that require faster CDR lock time. You can manually control the CDR to reduce its lock time using the `rx_locktorefclk` and `rx_locktodata` ports. In manual lock mode, the LTR/LTD controller sets the CDR in LTR or LTD mode depending on the logic level on the `rx_locktorefclk` and `rx_locktodata` signals.

When the `rx_locktorefclk` signal is asserted high, the LTR/LTD controller forces the CDR to lock to the reference clock. When the `rx_locktodata` signal is asserted high, it forces the CDR to lock to data. When both signals are asserted, the `rx_locktodata` signal takes precedence over the `rx_locktorefclk` signal, forcing the CDR to lock to data.

When the `rx_locktorefclk` signal is asserted high, the `rx_freqlocked` signal does not have any significance and is always driven low, indicating that the CDR is in LTR mode. When the `rx_locktodata` signal is asserted high, the `rx_freqlocked` signal is always driven high, indicating that the CDR is in LTD mode. If both signals are de-asserted, the CDR is in automatic lock mode.

The Altera-recommended transceiver reset sequence varies depending on the CDR lock mode.

For more information about reset sequence recommendations, refer to *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook*.

Deserializer

The deserializer block clocks in serial input data from the receiver buffer using the high-speed serial recovered clock and deserializes it using the low-speed parallel recovered clock. It forwards the deserialized data to the receiver PCS channel.

In single-width mode, the deserializer supports 8-bit and 10-bit deserialization factors. In double-width mode, the deserializer supports 16-bit and 20-bit deserialization factors.

Figure 1–48 shows the deserializer operation in single-width mode with a 10-bit deserialization factor.

Figure 1–48. Deserializer Operation in Single-Width Mode

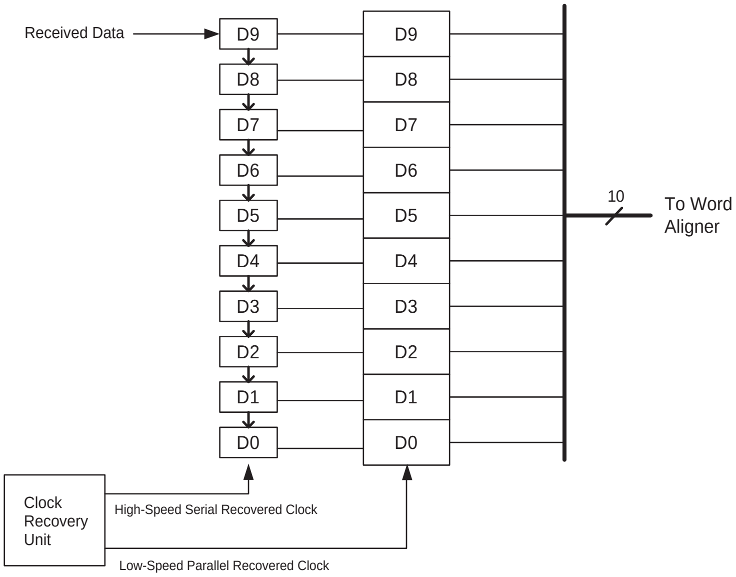
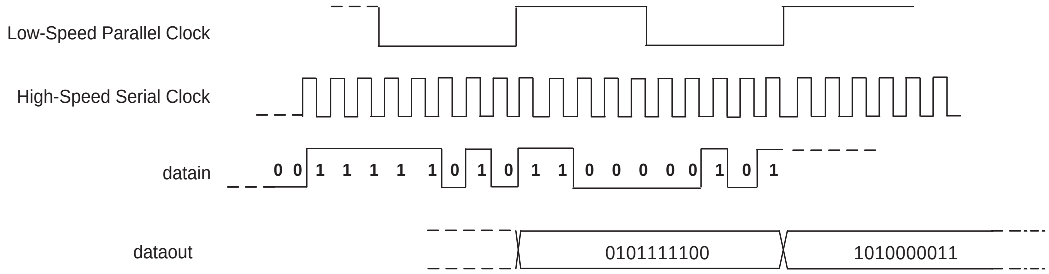


Figure 1–49 shows the serial bit order of the deserializer block input and the parallel data output of the deserializer block in single-width mode with 10-bit deserialization factor. The serial stream (0101111100) is deserialized to a value 10'h17C. The serial data is assumed to be received LSB to MSB.

Figure 1–49. Deserializer Bit Order in Single-Width Mode



Word Aligner

Because the data is serialized before transmission and then deserialized at the receiver, it loses the word boundary of the upstream transmitter upon deserialization. The word aligner receives parallel data from the deserializer and restores the word boundary based on a pre-defined alignment pattern that must be received during link synchronization.

Serial protocols like PCI Express (PIPE), XAUI, Gigabit Ethernet, Serial RapidIO, and SONET/SDH, specify a standard word alignment pattern. For proprietary protocols, the Stratix IV GX transceiver architecture allows you to select a custom word alignment pattern specific to your implementation.

In addition to restoring the word boundary, the word aligner also implements the following features:

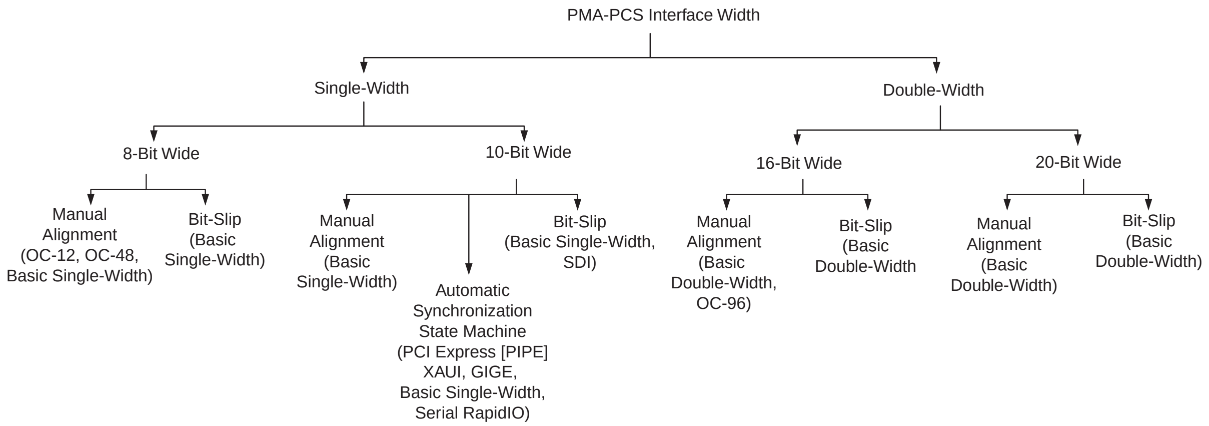
- Synchronization state machine in functional modes like PCI Express (PIPE), XAUI, GIGE, Serial RapidIO, and Basic single-width
- Programmable run length violation detection in all functional modes
- Receiver polarity inversion in all functional modes except PCI Express (PIPE)
- Receiver bit reversal in Basic single-width and Basic double-width modes
- Receiver byte reversal in Basic double-width modes

Depending on the configured functional mode, the word aligner operates in one of the following three modes:

- Manual alignment mode
- Automatic synchronization state machine mode
- Bit-slip mode

Figure 1–50 shows the word aligner operation in all supported configurations.

Figure 1–50. Word Aligner in All Supported Configurations


PMA-PCS Interface Width
Single-Width
Double-Width
8-Bit Wide
10-Bit Wide
16-Bit Wide
20-Bit Wide
Manual Alignment (OC-12, OC-48, Basic Single-Width)
Bit-Slip (Basic Single-Width)
Manual Alignment (Basic Single-Width)
Automatic Synchronization State Machine (PCI Express [PIPE] XAUI, GIGE, Basic Single-Width, Serial RapidIO)
Bit-Slip (Basic Single-Width, SDI)
Manual Alignment (Basic Double-Width, OC-96)
Bit-Slip (Basic Double-Width
Manual Alignment (Basic Double-Width)
Bit-Slip (Basic Double-Width)

Word Aligner in Single-Width Mode

In single-width mode, the PMA-PCS interface is either 8-bit or 10-bit wide. In 8-bit wide PMA-PCS interface modes, the word aligner receives 8-bit wide data from the deserializer. In 10-bit wide PMA-PCS interface modes, the word aligner receives 10-bit wide data from the deserializer. Depending on the configured functional mode, you can configure the word aligner in manual alignment mode, automatic synchronization state machine mode, or bit-slip mode.

Word Aligner in Single-Width Mode with 8-Bit PMA-PCS Interface Modes

The following functional modes support the 8-bit PMA-PCS interface:

- SONET/SDH OC-12
- SONET/SDH OC-48
- Basic single-width

Table 1–19 shows the word aligner configurations allowed in functional modes with an 8-bit wide PMA-PCS interface.

Table 1–19. Word Aligner Configurations with an 8-Bit Wide PMA-PCS Interface

Functional Mode	Allowed Word Configurations	Allowed Word Alignment Pattern Length
SONET/SDH OC-12	Manual Alignment	16 bits
SONET/SDH OC-48	Manual Alignment	16 bits
Basic single-width	Manual Alignment, Bit-Slip	16 bits

Manual Alignment Mode Word Aligner in 8-Bit PMA-PCS Interface Modes

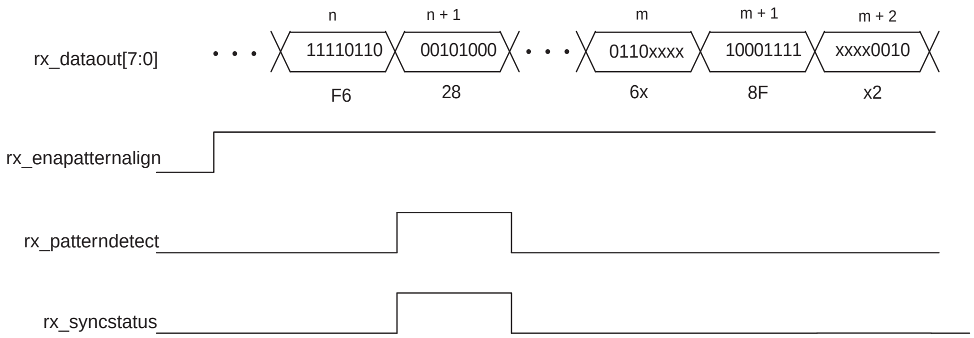
In manual alignment mode, the word aligner operation is controlled by the input signal `rx_enapatternalign`. The word aligner operation is edge-sensitive to the `rx_enapatternalign` signal. After de-assertion of `rx_digitalreset`, a rising edge on the `rx_enapatternalign` signal triggers the word aligner to look for the word alignment pattern in the received data stream. In SONET/SDH OC-12 and OC-48 modes, the word aligner looks for 16'hF628 (A1A2) or 32'hF6F62828 (A1A1A2A2), depending on whether the input signal `rx_a1a2size` is driven low or high, respectively. In Basic single-width mode, the word aligner looks for the 16-bit word alignment pattern programmed in the ALTGX MegaWizard Plug-In Manager. The word aligner aligns the 8-bit word boundary to the first word alignment pattern received after the rising edge on the `rx_enapatternalign` signal.

Two status signals, `rx_syncstatus` and `rx_patterndetect`, with the same latency as the datapath, are forwarded to the FPGA fabric to indicate word aligner status. On receiving the first word alignment pattern after the rising edge on the `rx_enapatternalign` signal, both the `rx_syncstatus` and `rx_patterndetect` signals are driven high for one parallel clock cycle synchronous to the MSByte of the word alignment pattern. Any word alignment pattern received thereafter in the same word boundary causes only the `rx_patterndetect` signal to go high for one clock cycle.

In order for the word aligner to re-synchronize to a new word boundary, you must de-assert `rx_enapatternalign` and re-assert it again to create a rising edge. After a rising edge on `rx_enapatternalign` signal, if the word alignment pattern is found in a different word boundary, the word aligner re-synchronizes to the new word boundary and asserts the `rx_syncstatus` and `rx_patterndetect` signals for one parallel clock cycle.

Figure 1–51 shows word aligner behavior in SONET/SDH OC-12 functional mode. The LSByte (8'hF6) and the MSByte (8'h28) of the 16-bit word alignment pattern are received in parallel clock cycles n and n + 1, respectively. The `rx_syncstatus` and `rx_patterndetect` signals are both driven high for one parallel clock cycle synchronous to the MSByte (8'h28) of the word alignment pattern. After initial word alignment, the 16-bit word alignment pattern is again received across the word boundary in clock cycles m, m + 1, and m + 2. The word aligner does not re-align to the new word boundary because of lack of a preceding rising edge on `rx_enapatternalign` signal. If you create a rising edge on the `rx_enapatternalign` signal before the word alignment pattern received across clock cycles m, m + 1, and m + 2, the word aligner re-aligns to the new word boundary causing both the `rx_syncstatus` and `rx_patterndetect` signals to go high for one parallel clock cycle.

Figure 1–51. Bit-Slip Mode in 8-Bit PMA-PCS Interface Mode



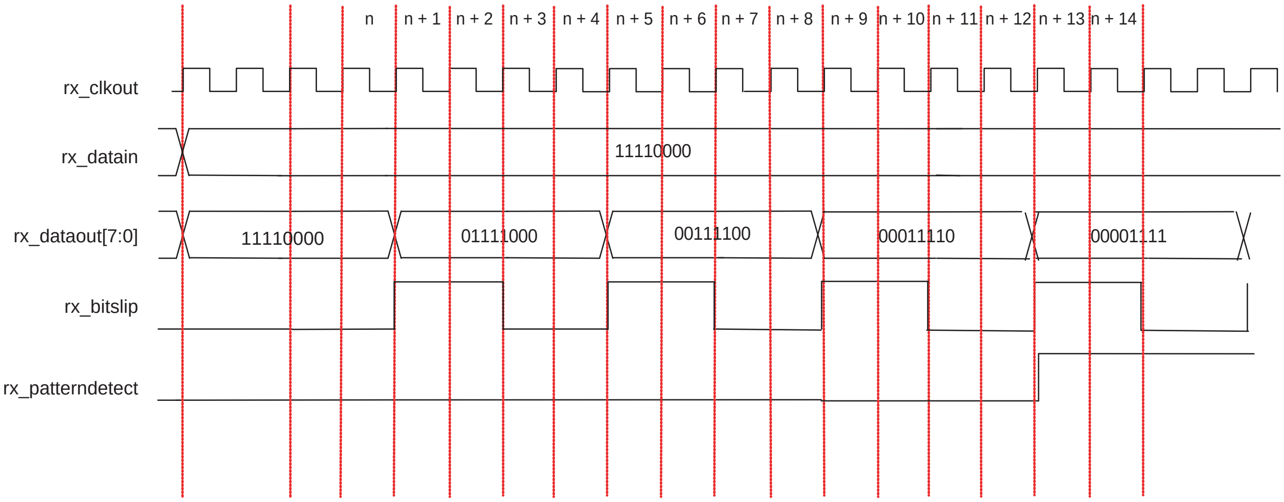
Bit-Slip Mode Word Aligner in 8-Bit PMA-PCS Interface Modes

Basic single-width mode with 8-bit PMA-PCS interface width allows the word aligner to be configured in bit-slip mode. The word aligner operation is controlled by the input signal `rx_bitslip` in bit-slip mode. At every rising edge of the `rx_bitslip` signal, the bit-slip circuitry slips one bit into the received data stream, effectively shifting the word boundary by one bit. In bit-slip mode, the word aligner status signal `rx_patterndetect` is driven high for one parallel clock cycle when the received data after bit-slipping matches the 16-bit word alignment pattern programmed in the ALTGX MegaWizard Plug-In Manager.

You can implement a bit-slip controller in the FPGA fabric that monitors either the `rx_dataout` signal and/or the `rx_patterndetect` signal and controls the `rx_bitslip` signal to achieve word alignment.

Figure 1–52 shows an example of word aligner configured in bit-slip mode. For this example, consider that 8'b11110000 is received back-to-back and 16'b0000111100011110 is specified as the word alignment pattern. A rising edge on the `rx_bitslip` signal at time n + 3 slips a single bit 0 at the MSB position forcing the `rx_dataout` to 8'b01111000. Another rising edge on the `rx_bitslip` signal at time n + 5 forces `rx_dataout` to 8'b00111100. Another rising edge on the `rx_bitslip` signal at time n + 9 forces `rx_dataout` to 8'b00011110. Another rising edge on the `rx_bitslip` signal at time n + 13 forces the `rx_dataout` to 8'b00001111. At this instance, `rx_dataout` in cycles n + 12 and n + 13 is 8'b00011110 and 8'b00001111, respectively, which matches the specified 16-bit alignment pattern 16'b0000111100011110. This results in the assertion of the `rx_patterndetect` signal.

Figure 1–52. Word Aligner Configured in Bit-Slip Mode



Word Aligner in Single-Width Mode with 10-Bit PMA-PCS Interface Modes

The following functional modes support the 10-bit PMA-PCS interface:

- PCI Express (PIPE) Gen1 and Gen2
- Serial RapidIO
- XAUI
- GIGE
- SDI
- Basic single-width mode

This section covers the following word aligner 10-bit PMA-PCS interface modes:

- Automatic synchronization state machine mode in 10-bit PMA-PCS interface mode
- Manual alignment mode in 10-bit PMA-PCS interface mode
- Bit-slip mode in 10-bit PMA-PCS interface mode

Table 1–20 shows the word aligner configurations allowed in functional modes with a 10-bit wide PMA-PCS interface.

Table 1–20. *Word Aligner Configurations in a 10-Bit Wide PMA-PCS Interface*

Functional Mode	Allowed Word Aligner Configurations	Allowed Word Alignment Pattern Length
PCI express (PIPE)	Automatic Synchronization State Machine	10 bits
Serial RapidIO	Automatic Synchronization State Machine	10 bits
XAUI	Automatic Synchronization State Machine	7 bits, 10 bits
GIGE	Automatic Synchronization State Machine	7 bits, 10 bits
SDI	Bit-Slip	N/A
Basic single-width mode	Manual Alignment, Automatic Synchronization State Machine, Bit-Slip	7 bits, 10 bits

Automatic Synchronization State Machine Mode Word Aligner in 10-Bit PMA-PCS Interface Mode

Protocols like PCI Express (PIPE), XAUI, Gigabit Ethernet, and Serial RapidIO require the receiver PCS logic to implement a synchronization state machine to provide hysteresis during link synchronization. Each of these protocols defines a specific number of synchronization code groups that the link must receive in order to acquire synchronization and a specific number of erroneous code groups that it must receive to fall out of synchronization.

In PCI Express (PIPE), XAUI, Gigabit Ethernet, and Serial RapidIO functional modes, the Quartus II software configures the word aligner in automatic synchronization state machine mode. It automatically selects the word alignment pattern length and the word alignment pattern as specified by each protocol. In each of these functional modes, the protocol-compliant synchronization state machine is implemented in the word aligner.

In Basic single-width functional mode with 10-bit PMA-PCS interface, you can configure the word aligner in automatic synchronization state machine mode by selecting the **Use the built-in synchronization state machine** option in the ALTGX MegaWizard Plug-In Manager. It also allows you to program a custom 7-bit or 10-bit word alignment pattern that the word aligner uses for synchronization.

☞ The 10-bit input data to the word aligner configured in automatic synchronization state machine mode must be 8B/10B encoded.

Table 1–21 shows the synchronization state machine parameters that the Quartus II software allows in supported functional modes. The synchronization state machine parameters are fixed for PCI Express (PIPE), XAUI, GIGE, and Serial RapidIO modes as specified by the respective protocol. For Basic single-width mode, you can program these parameters as suited to your proprietary protocol implementation.

Table 1–21. Synchronization State Machine Functional Modes

Functional Mode	PCI Express (PIPE)	XAUI	GIGE	Serial RapidIO	Basic Single-Width Mode
Number of valid synchronization code groups or ordered sets received to achieve synchronization	4	4	3	127	1 to 256
Number of erroneous code groups received to lose synchronization	17	4	4	3	1 to 64
Number of continuous good code groups received to reduce the error count by one	16	4	4	255	1 to 256

After de-assertion of the `rx_digitalreset` signal in automatic synchronization state machine mode, the word aligner starts looking for the word alignment pattern or synchronization code groups in the received data stream. When the programmed number of valid synchronization code groups or ordered sets is received, the `rx_syncstatus` signal is driven high to indicate that synchronization is acquired.

The rx_syncstatus signal is constantly driven high until the programmed number of erroneous code groups is received without receiving intermediate good groups; after which the rx_syncstatus is driven low. The word aligner indicates loss of synchronization (rx_syncstatus remains low) until the programmed number of valid synchronization code groups are received again.

Manual Alignment Mode Word Aligner in 10-Bit PMA-PCS Interface Mode

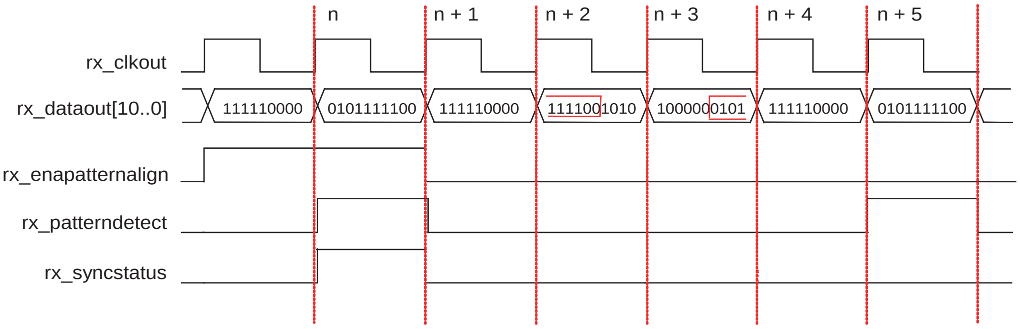
In Basic single-width mode with a 10-bit PMA-PCS interface, you can configure the word aligner in manual alignment mode by selecting the **Use manual word alignment mode** option in the ALTGX MegaWizard Plug-In Manager.

In manual alignment mode, the word aligner operation is controlled by the input signal rx_enapatternalign. The word aligner operation is level-sensitive to the rx_enapatternalign signal. If the rx_enapatternalign signal is held high, the word aligner looks for the programmed 7-bit or 10-bit word alignment pattern in the received data stream. It updates the word boundary if it finds the word alignment pattern in a new word boundary. If the rx_enapatternalign signal is de-asserted low, the word aligner maintains the current word boundary even when it sees the word alignment pattern in a new word boundary.

Two status signals, rx_syncstatus and rx_patterndetect, with the same latency as the datapath, are forwarded to the FPGA fabric to indicate the word aligner status. On receiving the first word alignment pattern after the rx_enapatternalign signal is asserted high, both the rx_syncstatus and rx_patterndetect signals are driven high for one parallel clock cycle. Any word alignment pattern received thereafter in the same word boundary causes only the rx_patterndetect signal to go high for one clock cycle. Any word alignment pattern received thereafter in a different word boundary causes the word aligner to re-align to the new word boundary only if the rx_enapatternalign signal is held high. The word aligner asserts the rx_syncstatus signal for one parallel clock cycle whenever it re-aligns to the new word boundary.

Figure 1–53 shows the manual alignment mode word aligner operation in 10-bit PMA-PCS interface mode. In this example, a /K28.5/ (10'b0101111100) is specified as the word alignment pattern. The word aligner aligns to the /K28.5/ alignment pattern in cycle n because the rx_enapatternalign signal is asserted high. The rx_syncstatus signal goes high for one clock cycle indicating alignment to a new word boundary. The rx_patterndetect signal also goes high for one clock cycle to indicate initial word alignment. At time n + 1, the rx_enapatternalign signal is de-asserted to instruct the word aligner to lock the current word boundary. The alignment pattern is detected again in a new word boundary across cycles n + 2 and n + 3. The word aligner does not align to this new word boundary because the rx_enapatternalign signal is held low. The /K28.5/ word alignment pattern is detected again in the current word boundary during cycle n + 5 causing the rx_patterndetect signal to goes high for one parallel clock cycle.

Figure 1–53. Word Aligner in 10-Bit PMA-PCS Manual Alignment Mode



☞ If the word alignment pattern is known to be unique and does not appear between word boundaries, you can constantly hold the `rx_enapatternalign` signal high because there is no possibility of false word alignment. If there is a possibility of the word alignment pattern occurring across word boundaries, you must control the `rx_enapatternalign` signal to lock the word boundary after the desired word alignment is achieved to avoid re-alignment to an incorrect word boundary.

Bit-Slip Mode Word Aligner in 10-Bit PMA-PCS Interface Mode

In some Basic single-width configurations with 10-bit PMA-PCS interface, you can configure the word aligner in bit-slip mode by selecting the **Use manual bit slipping mode** option in the ALTGX MegaWizard Plug-In Manager.

The word aligner operation for Basic single-width with 10-bit PMA-PCS interface is similar to the word aligner operation in Basic single-width mode with 8-bit PMA-PCS interface. Refer to "Manual Alignment Mode Word Aligner in 8-Bit PMA-PCS Interface Modes" on page 1–66 for word aligner operation in bit-slip mode. The only difference is that the bit-slip word aligner in 10-bit PMA-PCS interface modes allows 7-bit and 10-bit word alignment patterns, whereas the one in 8-bit PMA-PCS interface modes allows only 16-bit word alignment pattern.

Word Aligner in Double-Width Mode

In double-width mode, the PMA-PCS interface is either 16-bit or 20-bit wide. In 16-bit wide PMA-PCS interface modes, the word aligner receives 16-bit wide data from the deserializer. In 20-bit wide PMA-PCS interface modes, the word aligner receives 10-bit wide data from the deserializer. Depending on the configured functional mode, you can configure the word aligner in manual alignment mode or bit-slip mode. The automatic synchronization state machine mode is not supported for word aligner in double-width mode.

Word Aligner in Double-Width Mode with 16-Bit PMA-PCS Interface Modes

The following functional modes support the 16-bit PMA-PCS interface:

- SONET/SDH OC-96
- (OIF) CEI PHY interface
- Basic double-width

Table 1–22 shows the word aligner configurations allowed in functional modes with a 16-bit wide PMA-PCS interface.

Table 1–22. Word Aligner Configurations with a 16-Bit Wide PMA-PCS Interface *(Note 1)*

Functional Mode	Allowed Word Aligner Configurations	Allowed Word Alignment Pattern Length
SONET/SDH OC-96	Manual Alignment	16 bits, 32 bits
Basic double-width	Manual Alignment, Bit-Slip	8 bits, 16 bits, 32 bits

Note to Table 1–22:

(1) The word aligner is bypassed in (OIF) CEI PHY interface mode.

Manual Alignment Mode Word Aligner in 16-Bit PMA-PCS Interface Modes

In manual alignment mode, the word aligner starts looking for the programmed 8-bit, 16-bit, or 32-bit word alignment pattern in the received data stream as soon as `rx_digitalreset` is de-asserted low. It aligns to the first word alignment pattern received irrespective of the logic level driven on the `rx_enapatternalign` signal. Any word alignment pattern received thereafter in a different word boundary does not cause the word aligner to re-align to this new word boundary. After the initial word alignment following de-assertion of the `rx_digitalreset` signal, if a word re-alignment is required, you must use the `rx_enapatternalign` signal.

The word aligner operation is controlled by the input signal `rx_enapatternalign`. The word aligner operation is edge-sensitive to the `rx_enapatternalign` signal. A rising edge on the `rx_enapatternalign` signal triggers the word aligner to look for the word alignment pattern in the received data stream. The word aligner aligns the 16-bit word boundary to the first word alignment pattern received after the rising edge on the `rx_enapatternalign` signal. Any word alignment pattern received thereafter in a different word boundary does not cause the word aligner to re-align to this new word boundary. If another word re-alignment is required, you must de-assert and re-assert the `rx_enapatternalign` signal to create a rising edge on this signal.

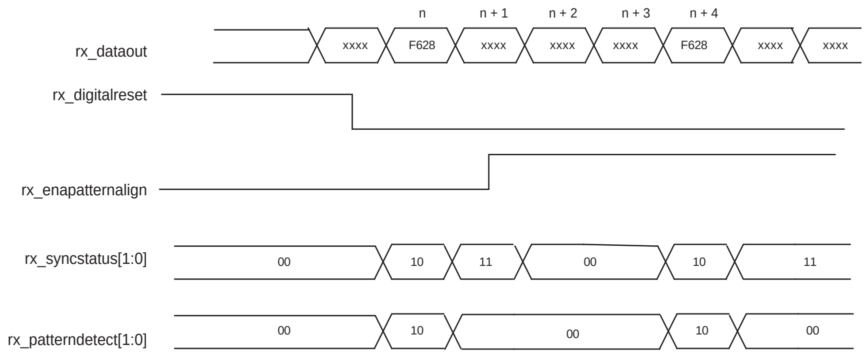
Two status signals, `rx_syncstatus` and `rx_patterndetect`, with the same latency as the datapath, are forwarded to the FPGA fabric to indicate word aligner status.

On receiving the first word alignment pattern, the `rx_patterndetect` signal is driven high for one parallel clock cycle synchronous to the data that matches the MSByte of the word alignment pattern. Any word alignment pattern received thereafter in the same word boundary causes `rx_patterndetect` to go high for one parallel clock cycle.

On receiving the first word alignment pattern, the `rx_syncstatus` signal is constantly driven high until the word aligner sees another rising edge on the `rx_enapatternalign` signal. The rising edge on `rx_enapatternalign` signal re-triggers the word alignment operation.

Figure 1–54 shows the manual alignment mode word aligner operation in 16-bit PMA-PCS interface mode. In this example, a 16'hF628 is specified as the word alignment pattern. The word aligner aligns to the 16'hF628 pattern received in cycle n after de-assertion of `rx_digitalreset`. The `rx_patterndetect[1]` signal is driven high for one parallel clock cycle. The `rx_syncstatus[1]` signal is driven high constantly until cycle n + 2, after which it is driven low because of the rising edge on the `rx_enapatternalign` signal that re-triggers the word aligner operation. The word aligner receives the word alignment pattern again in cycle n + 4, causing the `rx_patterndetect[1]` signal to be driven high for one parallel clock cycle and the `rx_syncstatus[1]` signal to be driven high constantly.

Figure 1–54. Manual Alignment Mode Word Aligner in 16-Bit PMA-PCS Interface Modes



Bit-Slip Mode Word Aligner in 16-Bit PMA-PCS Interface Modes

In some Basic double-width configurations with a 16-bit PMA-PCS interface, you can configure the word aligner in bit-slip mode by selecting the **Use manual bit slipping mode** option in the ALTGX MegaWizard Plug-In Manager.

The word aligner operation for Basic double-width with a 16-bit PMA-PCS interface is similar to the word aligner operation in Basic single-width mode with an 8-bit PMA-PCS interface. Refer to "Word Aligner in Single-Width Mode with 8-Bit PMA-PCS Interface Modes" on page 1–65 for word aligner operation in bit-slip mode. The only difference is that the bit-slip word aligner in 16-bit PMA-PCS interface modes allows 8-bit and 16-bit word alignment patterns, whereas the bit-slip word aligner in 8-bit PMA-PCS interface modes allows only a 16-bit word alignment pattern.

Word Aligner in Double-Width Mode with 20-Bit PMA-PCS Interface Modes

A 20-bit wide PMA-PCS interface is supported only in Basic double-width mode.

Table 1–23 shows the word aligner configurations allowed in functional modes with a 20-bit wide PMA-PCS interface.

Table 1–23. Word Aligner in 20-Bit PMA-PCS Interface Modes

Functional Mode	Allowed Word Aligner Configurations	Allowed Word Alignment Pattern Length
Basic double-width	Manual Alignment, Bit-Slip	7 bits, 10 bits, 20 bits

Manual Alignment Mode Word Aligner in 20-Bit PMA-PCS Interface Modes

The word aligner operation in Basic double-width mode with a 20-bit PMA-PCS interface is similar to the word aligner operation in Basic double-width mode with a 16-bit PMA-PCS interface. Refer to "Word Aligner in Double-Width Mode with 16-Bit PMA-PCS Interface Modes" on page 1–71 for word aligner operation in manual alignment mode. The only difference is that the manual alignment mode word aligner in 20-bit PMA-PCS interface modes allows 7-, 10-, and 20-bit word alignment patterns, whereas the one in 16-bit PMA-PCS interface modes allows only 8-, 16-, and 32-bit word alignment pattern.

Bit-Slip Mode Word Aligner in 20-Bit PMA-PCS Interface Modes

In some Basic single-width configurations with a 20-bit PMA-PCS interface, you can configure the word aligner in bit-slip mode by selecting the **Use manual bit slipping mode** option in the ALTGX MegaWizard Plug-In Manager.

The word aligner operation for Basic double-width with a 20-bit PMA-PCS interface is similar to the word aligner operation in Basic single-width mode with an 8-bit PMA-PCS interface. Refer to "Word Aligner in Single-Width Mode with 8-Bit PMA-PCS Interface Modes" on page 1–65 for word aligner operation in bit-slip mode. The only difference is that the bit-slip word aligner in 20-bit PMA-PCS interface modes allows only 7-, 10-, and 20-bit word alignment patterns, whereas the one in 8-bit PMA-PCS interface modes allows only a 16-bit word alignment pattern.

Table 1–24 summarizes the word aligner options available in Basic Single-Width and Double-Width modes

Table 1–24. Word Aligner Options Available in Basic Single-Width and Double-Width Modes *(Note 1)* (Part 1 of 2)

Functional Mode	PMA-PCS Interface Width	Word Alignment Mode	Word Alignment Pattern Length	rx_enapatternalign Sensitivity	rx_syncstatus Behavior	rx_patterndetect Behavior
Basic Single Width	8-bit	Manual Alignment	16-bit	Rising Edge Sensitive	Asserted high for one parallel clock cycle when the word aligner aligns to a new word boundary.	Asserted high for one parallel clock cycle when the word alignment pattern appears in the current word boundary.
		Bit Slip	16-bit	N/A	N/A	Asserted high for one parallel clock cycle when the word alignment pattern appears in the current word boundary.
	10-bit	Manual Alignment	7-, 10-bit	Level Sensitive	Asserted high for one parallel clock cycle when the word aligner aligns to a new word boundary.	Asserted high for one parallel clock cycle when the word alignment pattern appears in the current word boundary.
		Bit Slip	7-, 10-bit	N/A	N/A	Asserted high for one parallel clock cycle when the word alignment pattern appears in the current word boundary.
		Automatic Synchronization State Machine	7-, 10-bit	N/A	Stays high as long as the synchronization conditions are satisfied.	Asserted high for one parallel clock cycle when the word alignment pattern appears in the current word boundary.

Table 1–24. Word Aligner Options Available in Basic Single-Width and Double-Width Modes *(Note 1)* (Part 2 of 2)

Functional Mode	PMA-PCS Interface Width	Word Alignment Mode	Word Alignment Pattern Length	rx_enapatternalign Sensitivity	rx_syncstatus Behavior	rx_patterndetect Behavior
Basic Double Width	16-bit	Manual Alignment	8-, 16-, 32-bit	Rising Edge Sensitive	Stays high after the word aligner aligns to the word alignment pattern. Goes low on receiving a rising edge on `rx_enapatternalign` until a new word alignment pattern is received.	Asserted high for one parallel clock cycle when the word alignment pattern appears in the current word boundary.
		Bit Slip	8-, 16-, 32-bit	N/A	N/A	Asserted high for one parallel clock cycle when the word alignment pattern appears in the current word boundary.
	20-bit	Manual Alignment	7-, 10-, 20-bit	Rising Edge Sensitive	Stays high after the word aligner aligns to the word alignment pattern. Goes low on receiving a rising edge on `rx_enapatternalign` until a new word alignment pattern is received	Asserted high for one parallel clock cycle when the word alignment pattern appears in the current word boundary.
		Bit Slip	7-, 10-, 20-bit	N/A	N/A	Asserted high for one parallel clock cycle when the word alignment pattern appears in the current word boundary

Note to Table 1–24:

(1) Refer to "Word Aligner in Single-Width Mode" on page 1–65 and "Word Aligner in Double-Width Mode" on page 1–71 for details on word aligner operation.

Programmable Run Length Violation Detection

The programmable run length violation circuit resides in the word aligner block and detects consecutive 1s or 0s in the data. If the data stream exceeds the preset maximum number of consecutive 1s or 0s, the violation is signified by the assertion of the `rx_rlv` signal.

The run length violation status signal on the `rx_rlv` port has lower latency compared to the parallel data on the `rx_dataout` port. The `rx_rlv` signal in each channel is clocked by its parallel recovered clock. The FPGA fabric clock might have phase difference and/or PPM difference (in asynchronous systems) with respect to the recovered clock. To ensure that the FPGA fabric clock can latch the `rx_rlv` signal reliably, the run length violation circuitry asserts the `rx_rlv` signal for a minimum of two recovered clock cycles in single-width modes and a minimum of three recovered clock cycles in double-width modes. The `rx_rlv` signal can be asserted longer, depending on the run length of the received data.

In single-width mode, the run length violation circuit detects up to a run length of 128 (for an 8-bit deserialization factor) or 160 (for a 10-bit deserialization factor). The settings are in increments of 4 or 5 for the 8-bit or 10-bit deserialization factors, respectively.

In double-width mode, the run length violation circuit maximum run length detection is 512 (with a run length increment of 8) and 640 (with a run length increment of 10) for the 16-bit and 20-bit deserialization factors, respectively.

Table 1–25 summarizes the detection capabilities of the run length violation circuit.

Table 1–25. Detection Capabilities of the Run Length Violation Circuit

Mode	PMA-PCS Interface Width	Run Length Violation Detector Range	
		Minimum	**Maximum**
Single-width mode	8-bit	4	128
	10-bit	5	160
Double-width mode	16-bit	8	512
	20-bit	10	640

Receiver Polarity Inversion

The positive and negative signals of a serial differential link are often erroneously swapped during board layout. Solutions like board re-spin or major updates to the PLD logic can be expensive. The receiver polarity inversion feature is provided to correct this situation.

An optional `rx_invpolarity` port is available in all single-width and double-width modes except (OIF) CEI PHY and PCI Express (PIPE) to dynamically enable the receiver polarity inversion feature. In single-width modes, a high value on the `rx_invpolarity` port inverts the polarity of every bit of the 8-bit or 10-bit input data word to the word aligner in the receiver data path. In double-width modes, a high value on the `rx_invpolarity` port inverts the polarity of every bit of the 16-bit or 20-bit input data word to the word aligner in the receiver data path. Because inverting the polarity of each bit has the same effect as swapping the positive and negative signals of the differential link, correct data is seen by the receiver. `rx_invpolarity` is a dynamic signal and can cause initial disparity errors in an 8B/10B encoded link. The downstream system must be able to tolerate these disparity errors.

The generic receiver polarity inversion feature is different from the PCI Express (PIPE) 8B/10B polarity inversion feature. The generic receiver polarity inversion feature inverts the polarity of the data bits at the input of the word aligner and is not available in PCI Express (PIPE) mode. The PIPE 8B/10B polarity inversion feature inverts the polarity of the data bits at the input of the 8B/10B decoder and is available only in PCI Express (PIPE) mode.

Figure 1–55 shows the receiver polarity inversion feature in single-width 10-bit wide data path configurations.

Figure 1–55. Receiver Polarity Inversion in Single-Width Mode

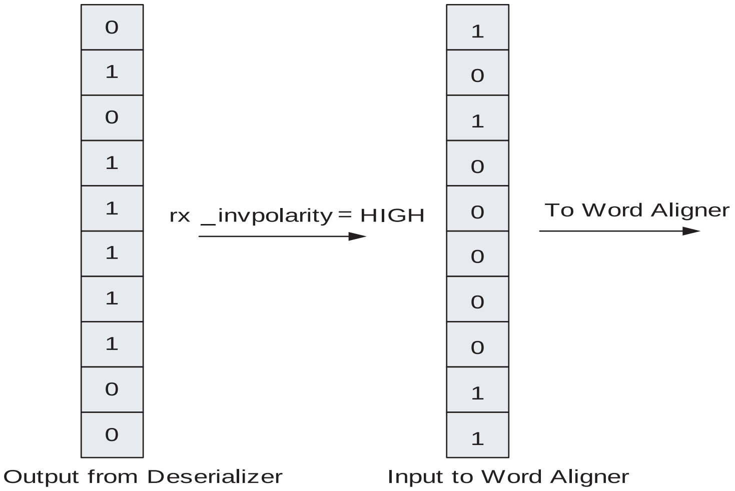
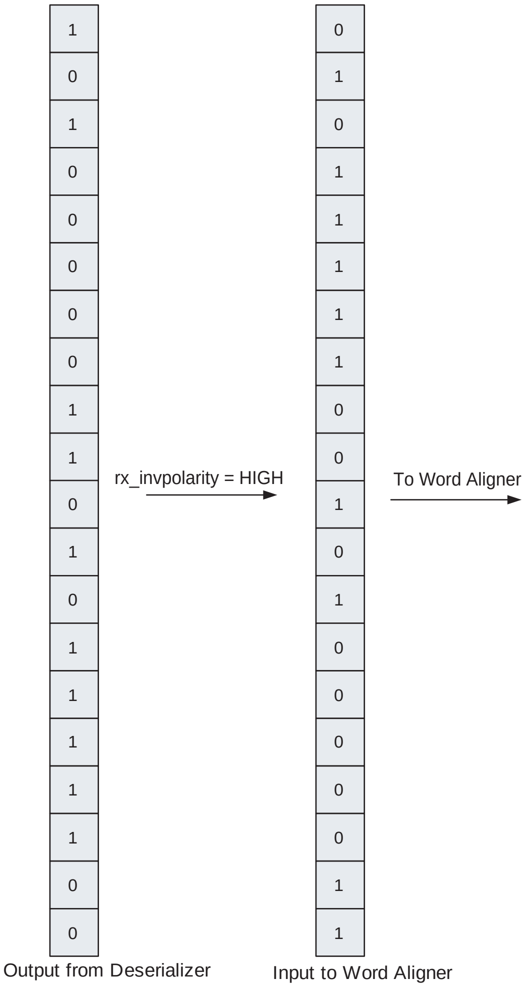


Figure 1–56 shows the receiver polarity inversion feature in double-width 20-bit wide data path configurations.

Figure 1–56. Receiver Polarity Inversion in Double-Width Mode



Receiver Bit Reversal

By default, the Stratix IV GX receiver assumes a LSBit-to-MSBit transmission. If the transmission order is MSBit-to-LSBit, the receiver forwards the bit-flipped version of the parallel data to the FPGA fabric on the `rx_dataout` port. The receiver bit reversal feature is available to correct this situation.

The receiver bit reversal feature is available through the `rx_revbitordwa` port only in Basic single-width and double-width modes with the word aligner configured in bit-slip mode.

- When the `rx_revbitordwa` signal is driven high in Basic single-width mode, the 8-bit or 10-bit data `D[7:0]` or `D[9:0]` at the output of the word aligner gets rewired to `D[0:7]` or `D[0:9]`, respectively.
- When the `rx_revbitordwa` signal is driven high in Basic double-width mode, the 16-bit or 20-bit data `D[15:0]` or `D[19:0]` at the output of the word aligner gets rewired to `D[0:15]` or `D[0:19]`, respectively.

Flipping the parallel data using this feature allows the receiver to forward the correct bit-ordered data to the FPGA fabric on the `rx_dataout` port in the case of MSBit-to-LSBit transmission.

Figure 1–57 shows the receiver bit reversal feature in Basic single-width 10-bit wide data path configurations.

Figure 1–57. Receiver Bit Reversal in Single-Width Mode

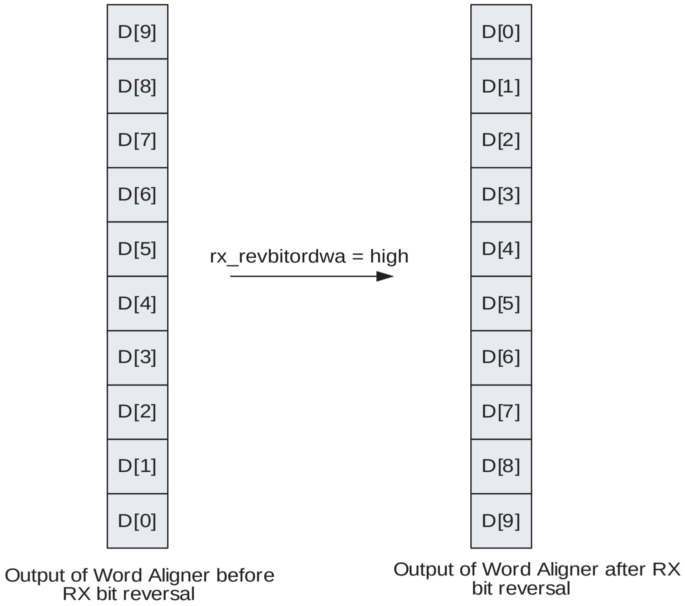
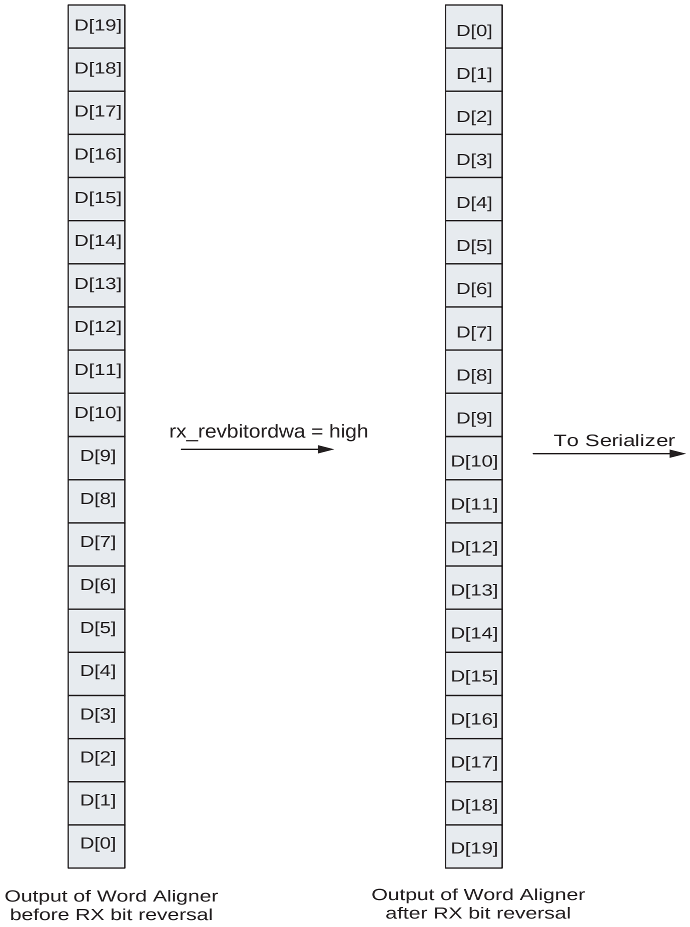


Figure 1–58 shows the receiver bit reversal feature in Basic double-width 20-bit wide data path configurations.

Figure 1–58. Receiver Bit Reversal in Double-Width Mode



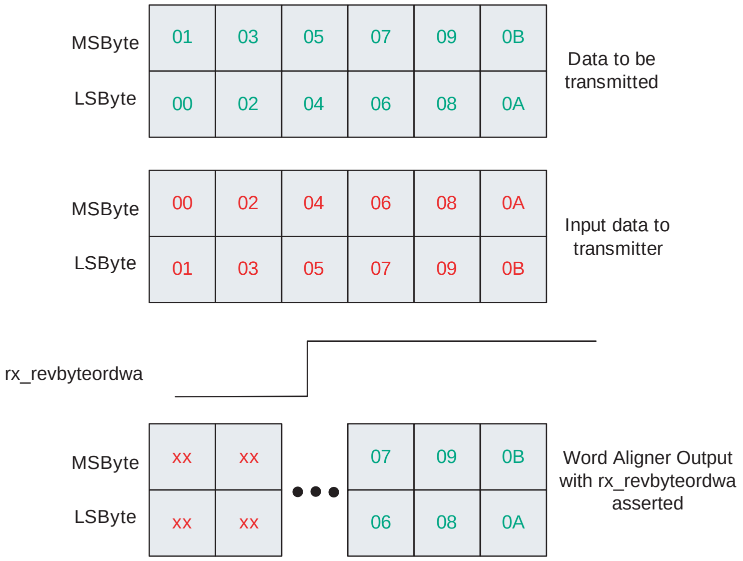
Receiver Byte Reversal in Basic Double-Width Modes

The MSByte and LSByte of the input data to the transmitter may be erroneously swapped. The receiver byte reversal feature is available to correct this situation.

An optional port, `rx_revbyteordwa`, is available only in Basic double-width mode to enable receiver byte reversal. In 8B/10B enabled mode, a high value on `rx_revbyteordwa` swaps the 10-bit MSByte and LSByte of the 20-bit word at the output of the word aligner in the receiver data path. In non-8B/10B enabled mode, a high value on `rx_revbyteordwa` swaps the 8-bit MSByte and LSByte of the 16-bit word at the output of the word aligner in the receiver data path. This compensates for the erroneous swapping at the transmitter and corrects the data received by the downstream systems. `rx_revbyteorderwa` is a dynamic signal and can cause an initial disparity error at the receiver of an 8B/10B encoded link. The downstream system must be able to tolerate this disparity error.

Figure 1–59 shows the receiver byte reversal feature.

Figure 1–59. Receiver Byte Reversal Feature



Deskew FIFO

Code groups received across four lanes in a XAUI link can be misaligned with respect to one another because of skew in the physical medium or differences between the independent clock recoveries per lane. The XAUI protocol allows a maximum skew of 40 UI (12.8 ns) as seen at the receiver of the four lanes.

The XAUI protocol requires the physical layer device to implement a deskew circuitry to align all four channels. To enable the deskew circuitry at the receiver to align the four channels, the transmitter sends a /A/ (/K28.3/) code group simultaneously on all four channels during inter-packet gap (IPG). The skew introduced in the physical medium and the receiver channels can cause the /A/ code groups to be received misaligned with respect to each other.

The deskew operation is performed by the deskew circuitry in XAUI functional mode. Deskew circuitry consists of:

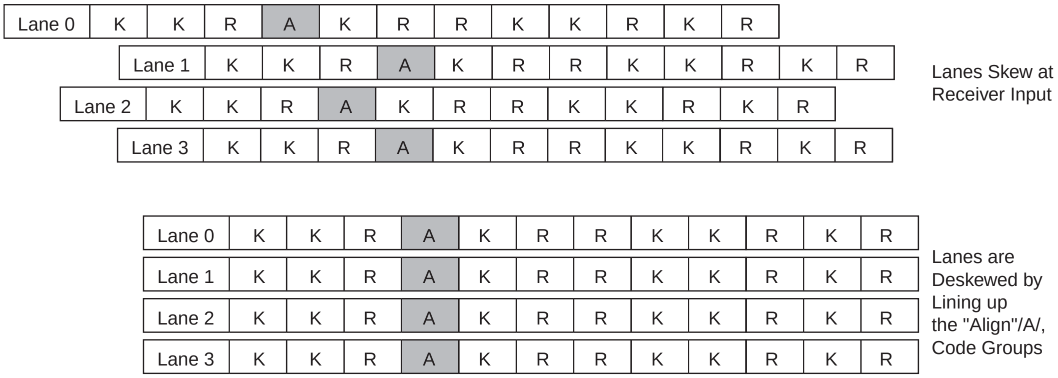
- A 16-word deep deskew FIFO in each of the four channels
- Control logic in the `CMU0 channel` of the transceiver block that controls the deskew FIFO write and read operations in each channel

Deskew circuitry is only available in XAUI mode.

The deskew FIFO in each channel receives data from its word aligner. The deskew operation begins only after link synchronization is achieved on all four channels as indicated by a high level on the `rx_syncstatus` signal from the word aligner in each channel. Until the first /A/ code group is received, the deskew FIFO read and write pointers in each channel are not incremented. After the first /A/ code group is received, the write pointer starts incrementing for each word received but the read pointer is frozen. If the /A/ code group is received on each of the four channels within 10 recovered clock cycles of each other, the read pointer of all four deskew FIFOs is released simultaneously, aligning all four channels.

Figure 1–60 shows lane skew at the receiver input and how the deskew FIFO uses the /A/ code group to align the channels.

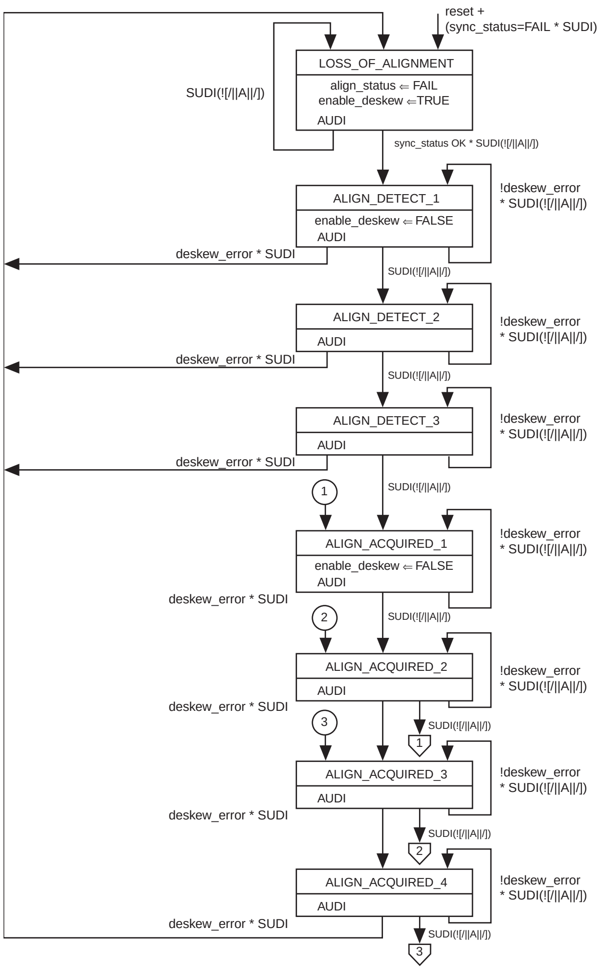
Figure 1–60. Deskew FIFO—Lane Skew at the Receiver Input


Lane 0 K K R A K R R K K R K R
Lane 1 K K R A K R R K K R K R
Lane 2 K K R A K R R K K R K R
Lane 3 K K R A K R R K K R K R
Lanes Skew at Receiver Input
Lane 0 K K R A K R R K K R K R
Lane 1 K K R A K R R K K R K R
Lane 2 K K R A K R R K K R K R
Lane 3 K K R A K R R K K R K R
Lanes are Deskewed by Lining up the "Align"/A/, Code Groups

After alignment of the first ||A|| column, if three additional aligned ||A|| columns are observed at the output of the deskew FIFOs of the four channels, the `rx_channelaligned` signal is asserted high, indicating channel alignment is acquired. After acquiring channel alignment, if four misaligned ||A|| columns are seen at the output of the deskew FIFOs in all four channels with no aligned ||A|| columns in between, the `rx_channelaligned` signal is de-asserted low, indicating loss of channel alignment.

The deskew operation in XAUI functional mode is compliant to the PCS deskew state machine diagram specified in clause 48 of the IEEE P802.3ae, as shown in Figure 1–61.

Figure 1–61. Deskew FIFO Operation in XAUI Functional Mode



Rate Match (Clock Rate Compensation) FIFO

In asynchronous systems, the upstream transmitter and the local receiver can be clocked with independent reference clocks. Frequency differences in the order of a few hundred PPM can corrupt the data when latching from the recovered clock domain (the same clock domain as the upstream transmitter reference clock) to the local receiver reference clock domain.

The rate match FIFO compensates for small clock frequency differences between the upstream transmitter and the local receiver clocks by inserting or removing SKP symbols or ordered-sets from the IPG or idle streams. It deletes SKP symbols or ordered-sets when the upstream transmitter reference clock frequency is higher than the local receiver reference clock frequency. It inserts SKP symbols or ordered-sets when the local receiver reference clock frequency is higher than the upstream transmitter reference clock frequency.

The rate match FIFO consists of a 20-word deep FIFO and necessary logic that controls insertion and deletion of a skip character or ordered-set, depending on the PPM difference.

The rate match FIFO is mandatory and cannot be bypassed in the following functional modes:

- PCI Express (PIPE)
- XAUI
- GIGE

The rate match FIFO is optional in the following functional modes:

- Basic single-width
- Basic double-width

The rate match FIFO receives data from the word aligner (non-XAUI functional modes) or deskew FIFO (XAUI functional mode) in the receiver datapath. It provides the following status signals forwarded to the FPGA fabric:

- `rx_rmfifodatainserted`—indicates insertion of a skip character or ordered-set
- `rx_rmfifodatadeleted`—indicates deletion of a skip character or ordered-set
- `rx_rmfifofull`—indicates rate match FIFO full condition
- `rx_rmfifoempty`—indicates rate match FIFO empty condition

☞ The rate match FIFO status signals are not available in PCI Express (PIPE) mode. These signals are encoded on the `pipestatus[2:0]` signal in PCI Express (PIPE) mode as specified in the PIPE specification.

Rate Match FIFO in PCI Express (PIPE) Mode

In PCI Express (PIPE) mode, the rate match FIFO is capable of compensating up to ± 300 PPM (total 600 PPM) difference between the upstream transmitter and the local receiver. The PCI Express (PIPE) protocol requires the transmitter to send SKP ordered sets during IPGs, adhering to rules listed in the base specification. The SKP ordered set is defined as a /K28.5/ COM symbol followed by three consecutive /K28.0/ SKP symbols groups. The PCI Express (PIPE) protocol requires the receiver to recognize a SKP ordered set as a /K28.5/ COM symbol followed by one-to-five consecutive /K28.0/ SKP symbols.

The rate match FIFO operation is compliant to PCI Express Base Specification 2.0. The rate match operation begins after the synchronization state machine in the word aligner indicates synchronization is acquired by driving the `rx_syncstatus` signal high. The rate match FIFO looks for the SKP ordered set and deletes or inserts SKP symbols as necessary to prevent the rate match FIFO from overflowing or under running.

The rate match FIFO inserts or deletes only one SKP symbol per SKP ordered set received. The rate match FIFO insertion and deletion events are communicated to the FPGA fabric on the `pipestatus[2:0]` port from each channel. The `pipestatus[2:0]` signal is driven to 3'b001 for one clock cycle synchronous to the /K28.5/ COM symbol of the SKP ordered set in which the /K28.0/ SKP symbol is inserted. The `pipestatus[2:0]` signal is driven to 3'b010 for one clock cycle synchronous to the /K28.5/ COM symbol of the SKP ordered set from which the /K28.0/ SKP symbol is deleted.

Figure 1–62 shows an example of rate match deletion in the case where two /K28.0/ SKP symbols are required to be deleted. Only one /K28.0/ SKP symbol is deleted per SKP ordered set received.

Figure 1–62. Rate Match Deletion in PCI Express (PIPE) Mode

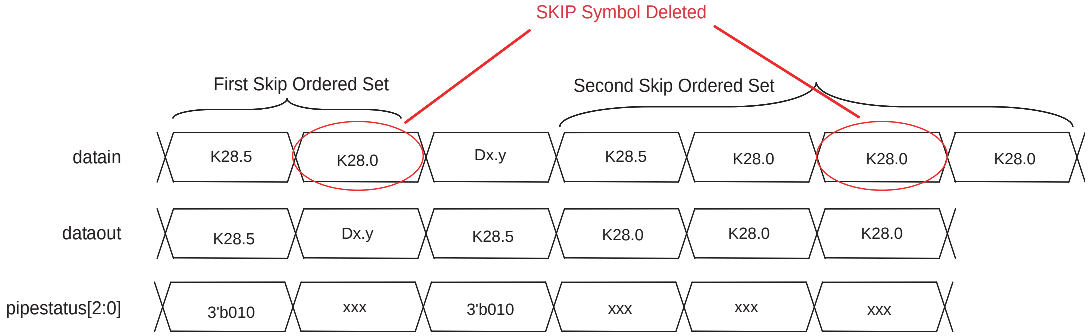
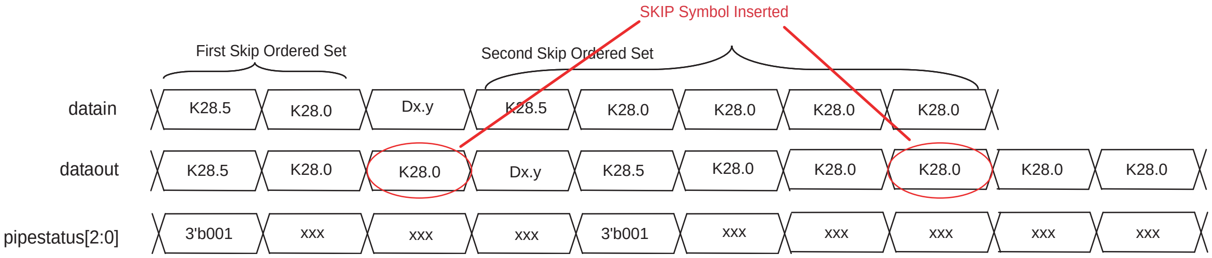


Figure 1–63 shows an example of rate match insertion in the case where two SKP symbols are required to be inserted. Only one /K28.0/ SKP symbol is inserted per SKP ordered set received.

Figure 1–63. Rate Match Insertion in PCI Express (PIPE) Mode

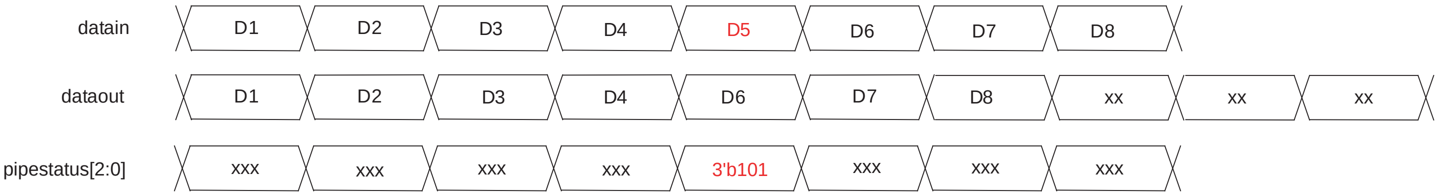


The rate match FIFO full and empty conditions are communicated to the FPGA fabric on `pipestatus[2:0]` port from each channel.

The rate match FIFO in PCI Express (PIPE) mode automatically deletes the data byte that causes the FIFO to go full and drives `pipestatus[2:0] = 3'b101` synchronous to the subsequent data byte.

Figure 1–64 shows the rate match FIFO full condition in PCI Express (PIPE) mode. The rate match FIFO becomes full after receiving data byte D4.

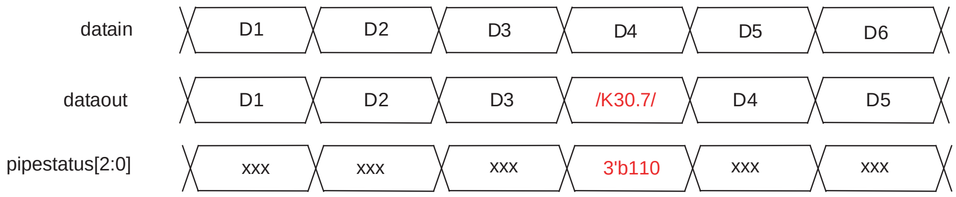
Figure 1–64. Rate Match FIFO Full Condition in PCI Express (PIPE) Mode



The rate match FIFO automatically inserts /K30.7/ (9'h1FE) after the data byte that causes the FIFO to go empty and drives `pipestatus[2:0] = 3'b110` flag synchronous to the inserted /K30.7/ (9'h1FE).

Figure 1–65 shows rate match FIFO empty condition in PCI Express (PIPE) mode. The rate match FIFO becomes empty after reading out data byte D3.

Figure 1–65. Rate Match FIFO Empty Condition in PCI Express (PIPE) Mode



☞ You can configure the rate match FIFO in low-latency mode by turning off the **Enable Rate Match FIFO** option in the ALTGX MegaWizard Plug-In Manager.

Rate Match FIFO in XAUI Mode

In XAUI mode, the rate match FIFO is capable of compensating up to ±100 PPM (total 200 PPM total) difference between the upstream transmitter and the local receiver reference clock. The XAUI protocol requires the transmitter to send /R/ (/K28.0/) code groups simultaneously on all four lanes (denoted as ||R|| column) during inter-packet gaps, adhering to rules listed in the IEEE P802.3ae specification. The rate match FIFO operation in XAUI mode is compliant to the IEEE P802.3ae specification.

The rate match operation begins after:

- The synchronization state machine in the word aligner of all four channels indicates synchronization was acquired by driving its `rx_syncstatus` signal high
- The deskew FIFO block indicates alignment was acquired by driving the `rx_channelaligned` signal high

The rate match FIFO looks for the ||R|| column (simultaneous /R/ code group on all four channels) and deletes or inserts ||R|| column to prevent the rate match FIFO from overflowing or under-running. It can insert or delete as many ||R|| columns as necessary to perform the rate match operation.

Two flags, `rx_rmfifodatadeleted` and `rx_rmfifodatainserted`, that indicate rate match FIFO deletion and insertion events, respectively, are forwarded to the FPGA fabric. If an ||R|| column is deleted, the `rx_rmfifodeleted` flag from each of the four channels goes high for one clock cycle per deleted ||R|| column. If an ||R|| column is inserted, the `rx_rmfifoinserted` flag from each of the four channels goes high for one clock cycle per inserted ||R|| column.

Figure 1–66 shows an example of rate match deletion in the case where three ||R|| columns are required to be deleted.

Figure 1–66. Rate Match Deletion in XAUI Mode

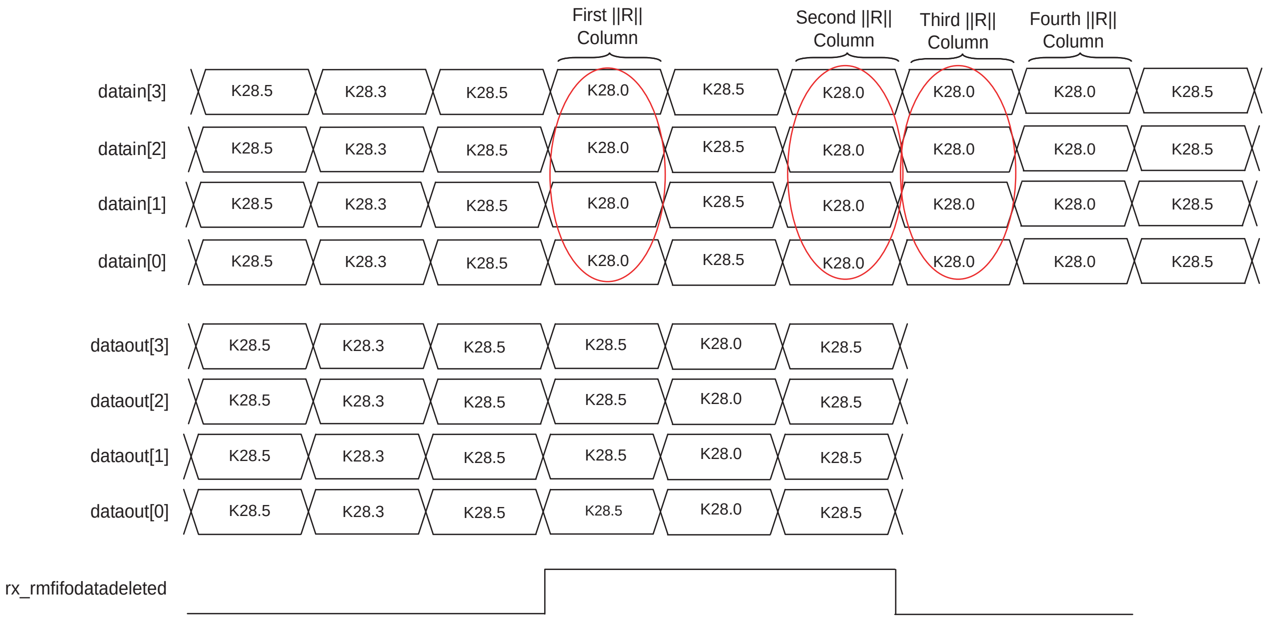
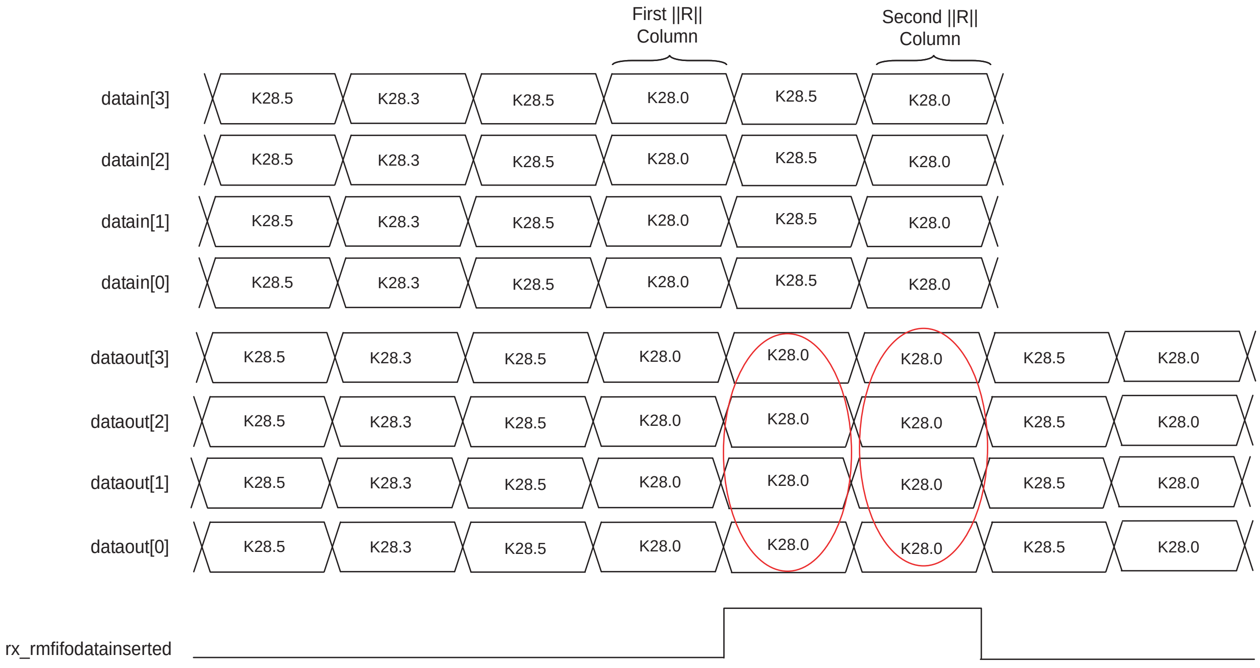

First ||R|| Column
Second ||R|| Column
Third ||R|| Column
Fourth ||R|| Column
datain[3] K28.5 K28.3 K28.5 K28.0 K28.5 K28.0 K28.0 K28.0 K28.5
datain[2] K28.5 K28.3 K28.5 K28.0 K28.5 K28.0 K28.0 K28.0 K28.5
datain[1] K28.5 K28.3 K28.5 K28.0 K28.5 K28.0 K28.0 K28.0 K28.5
datain[0] K28.5 K28.3 K28.5 K28.0 K28.5 K28.0 K28.0 K28.0 K28.5
dataout[3] K28.5 K28.3 K28.5 K28.5 K28.0 K28.5
dataout[2] K28.5 K28.3 K28.5 K28.5 K28.0 K28.5
dataout[1] K28.5 K28.3 K28.5 K28.5 K28.0 K28.5
dataout[0] K28.5 K28.3 K28.5 K28.5 K28.0 K28.5
rx_rmfifodatadeleted

Figure 1–67 shows an example of rate match insertion in the case where two | | R | | columns are required to be inserted.

Figure 1–67. Rate Match Insertion in XAUI Mode


First ||R|| Column
Second ||R|| Column
datain[3] K28.5 K28.3 K28.5 K28.0 K28.5 K28.0
datain[2] K28.5 K28.3 K28.5 K28.0 K28.5 K28.0
datain[1] K28.5 K28.3 K28.5 K28.0 K28.5 K28.0
datain[0] K28.5 K28.3 K28.5 K28.0 K28.5 K28.0
dataout[3] K28.5 K28.3 K28.5 K28.0 K28.0 K28.0 K28.5 K28.0
dataout[2] K28.5 K28.3 K28.5 K28.0 K28.0 K28.0 K28.5 K28.0
dataout[1] K28.5 K28.3 K28.5 K28.0 K28.0 K28.0 K28.5 K28.0
dataout[0] K28.5 K28.3 K28.5 K28.0 K28.0 K28.0 K28.5 K28.0
rx_rmfifodatainserted

Two flags, `rx_rmfifofull` and `rx_rmfifoempty`, are forwarded to the FPGA fabric to indicate rate match FIFO full and empty conditions.

In XAUI mode, the rate match FIFO does not automatically insert or delete code groups to overcome FIFO empty and full conditions, respectively. It asserts the `rx_rmfifofull` and `rx_rmfifoempty` flags for at least three recovered clock cycles to indicate rate match FIFO full and empty conditions, respectively.

☞ In case of rate match FIFO full and empty conditions, you must assert the `rx_digitalreset` signal to reset the receiver PCS blocks.

Rate Match FIFO in GIGE Mode

In GIGE mode, the rate match FIFO is capable of compensating up to ±100 PPM (total 200 PPM total) difference between the upstream transmitter and the local receiver reference clock. The GIGE protocol requires the transmitter to send idle ordered sets /I1/ (/K28.5/D5.6/) and /I2/ (/K28.5/D16.2/) during inter-packet gaps, adhering to rules listed in the IEEE 802.3 specification.

The rate match operation begins after the synchronization state machine in the word aligner indicates synchronization is acquired by driving the `rx_syncstatus` signal high. The rate match FIFO is capable of deleting or inserting the /I2/ (/K28.5/D16.2/) ordered set to prevent the rate match FIFO from overflowing or under-running during normal packet transmission. The rate match FIFO is also capable of deleting or inserting the first two bytes of the /C2/ ordered set (/K28.5/D2.2/Dx.y/Dx.y/) to prevent the rate match FIFO from overflowing or under-running during the auto negotiation phase.

The rate match FIFO can insert or delete as many /I2/ or /C2/ (first two bytes) as necessary to perform the rate match operation.

Two flags, `rx_rmfifodatadeleted` and `rx_rmfifodatainserted`, that indicate rate match FIFO deletion and insertion events, respectively, are forwarded to the FPGA fabric. Both the `rx_rmfifodatadeleted` and `rx_rmfifodatainserted` flags are asserted for two clock cycles for each deleted and inserted /I2/ ordered-set, respectively.

Figure 1–68 shows an example of rate match FIFO deletion in the case where three symbols are required to be deleted. Because the rate match FIFO can only delete /I2/ ordered-set, it deletes two /I2/ ordered-sets (four symbols deleted).

Figure 1–68. Rate Match Deletion in GIGE Mode

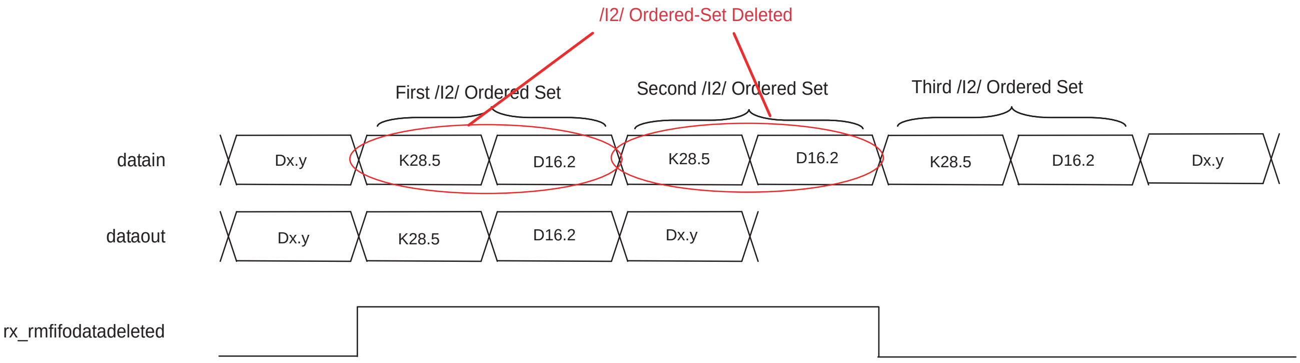
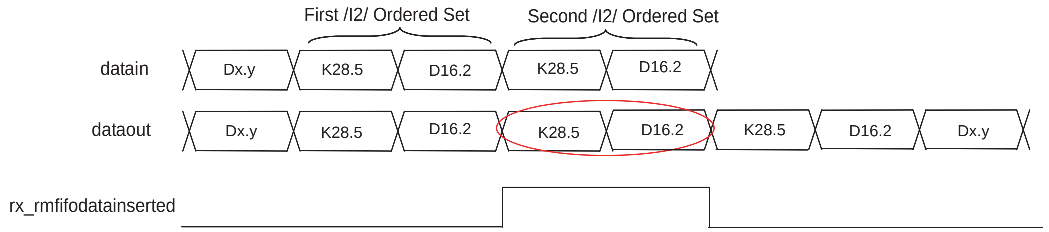


Figure 1–69 shows an example of rate match FIFO insertion in the case where one symbol is required to be inserted. Because the rate match FIFO can only delete /I2/ ordered-set, it inserts one /I2/ ordered-sets (two symbols inserted).

Figure 1–69. Rate Match Insertion in GIGE Mode



Two flags, `rx_rmfifofull` and `rx_rmfifoempty`, are forwarded to the FPGA fabric to indicate rate match FIFO full and empty conditions.

In GIGE mode, the rate match FIFO does not insert or delete code groups automatically to overcome FIFO empty and full conditions, respectively. It asserts the `rx_rmfifofull` and `rx_rmfifoempty` flags for at least two recovered clock cycles to indicate rate match FIFO full and empty conditions, respectively.

☞ In case of rate match FIFO full and empty conditions, you must assert the `rx_digitalreset` signal to reset the receiver PCS blocks.

Rate Match FIFO in Basic Single-Width Mode

In Basic single-width mode, the rate match FIFO is capable of compensating up to ±300 PPM (total 600 PPM total) difference between the upstream transmitter and the local receiver reference clock.

☞ To enable the rate match FIFO in Basic single-width mode, the transceiver channel must have both the transmitter and the receiver channel instantiated. You must select the **Receiver and Transmitter** option in the **What is the operation mode?** field in the ALTGX MegaWizard Plug-In Manager. You must also enable the 8B/10B encoder/decoder in Basic single-width mode with rate match FIFO enabled.

Depending on your proprietary protocol implementation, you can select two 20-bit rate match patterns in the ALTGX MegaWizard Plug-In Manager under the What is the rate match pattern1 and What is the rate match pattern2 fields. Each of the two programmed 20-bit rate match patterns consists of a 10-bit skip pattern and a 10-bit control pattern. You must choose 10-bit code groups that have a neutral disparity as the skip patterns. The rate match FIFO operation begins after the word aligner synchronization status `rx_syncstatus` goes high. When the rate matcher receives either of the two 10-bit control patterns followed by the respective 10-bit skip pattern, it inserts or deletes the 10-bit skip pattern as necessary to avoid the rate match FIFO from overflowing or under running.

The rate match FIFO can delete a maximum of four skip patterns from a cluster, provided there is one skip pattern left in the cluster after deletion. The rate match FIFO can insert a maximum of four skip patterns in a cluster, provided there are no more than five skip patterns in the cluster after insertion. Two flags, `rx_rmfifodatadeleted` and `rx_rmfifodatainserted`, indicate rate match FIFO deletion and insertion events, respectively, are forwarded to the FPGA fabric.

Figure 1–70 shows an example of rate match FIFO deletion in the case where three skip patterns are required to be deleted. In this example, /K28.5/ is the control pattern and neutral disparity /K28.0/ is the skip pattern. The first skip cluster has a /K28.5/ control pattern followed by two /K28.0/ skip patterns. The second skip cluster has a /K28.5/ control pattern followed by four /K28.0/ skip patterns. The rate match FIFO deletes only one /K28.0/ skip pattern from the first skip cluster in order to maintain at least one skip pattern in the cluster after deletion. Two /K28.0/ skip patterns are deleted from the second cluster for a total of three skip patterns deletion requirement.

Figure 1–70. Rate Match Deletion in Basic Single-Width Mode

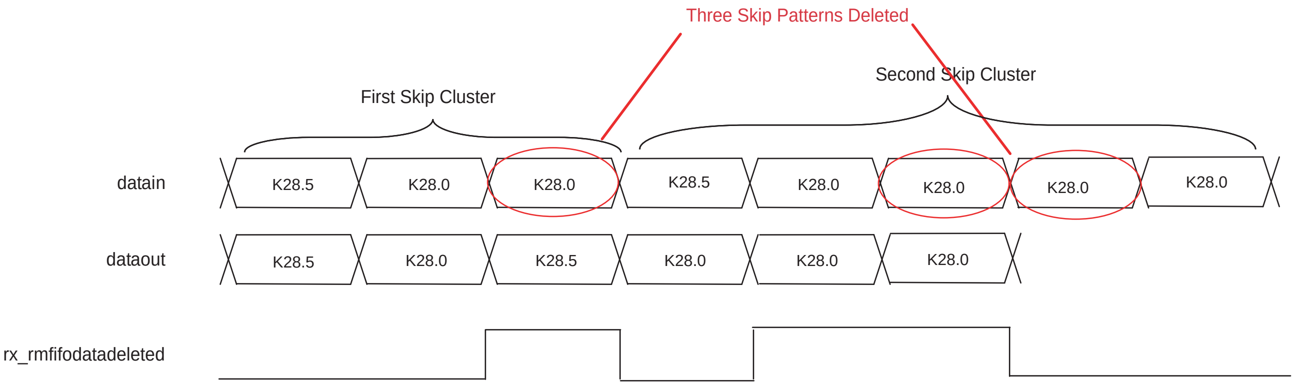
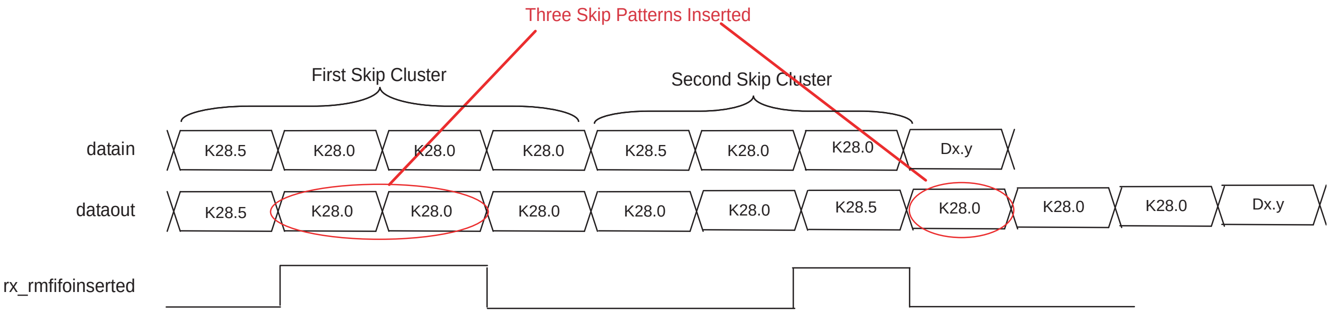


Figure 1–71 shows an example of rate match FIFO insertion in the case where three skip patterns are required to be inserted. In this example, /K28.5/ is the control pattern and neutral disparity /K28.0/ is the skip pattern. The first skip cluster has a /K28.5/ control pattern followed by three /K28.0/ skip patterns. The second skip cluster has a /K28.5/ control pattern followed by one /K28.0/ skip pattern. The rate match FIFO inserts only two /K28.0/ skip patterns into the first skip cluster in order to maintain a maximum of five skip patterns in the cluster after insertion. One /K28.0/ skip pattern is inserted into the second cluster for a total of three skip patterns insertion requirement.

Figure 1–71. Rate Match Insertion in Basic Single-Width Mode

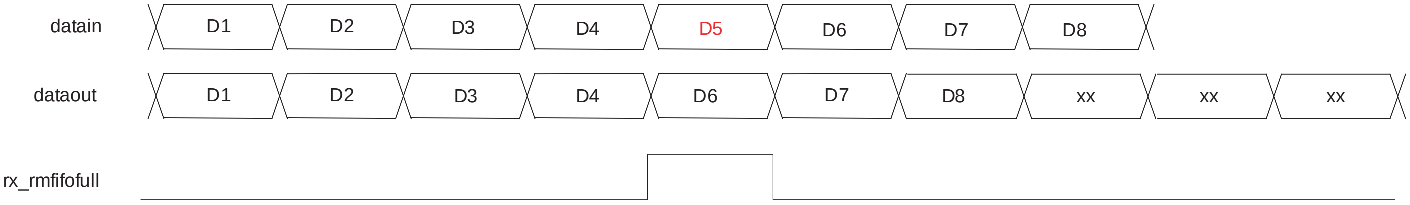


Two flags, `rx_rmfifofull` and `rx_rmfifoempty`, are forwarded to the FPGA fabric to indicate rate match FIFO full and empty conditions.

The rate match FIFO in Basic single-width mode automatically deletes the data byte that causes the FIFO to go full and asserts the `rx_rmfifofull` flag synchronous to the subsequent data byte.

Figure 1–72 shows the rate match FIFO full condition in Basic single-width mode. The rate match FIFO becomes full after receiving data byte D4.

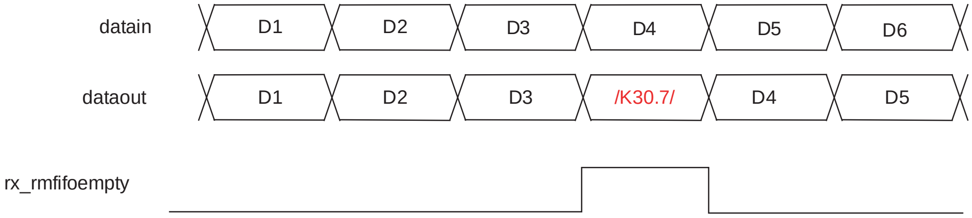
Figure 1–72. Rate Match FIFO Full Condition in Basic Single-Width Mode



The rate match FIFO automatically inserts /K30.7/ (9'h1FE) after the data byte that causes the FIFO to go empty and asserts the `rx_fifoempty` flag synchronous to the inserted /K30.7/ (9'h1FE).

Figure 1–73 shows rate match FIFO empty condition in Basic single-width mode. The rate match FIFO becomes empty after reading out data byte D3.

Figure 1–73. Rate Match FIFO Empty Condition in Basic Single-Width Mode



Rate Match FIFO in Basic Double-Width Mode

In Basic double-width mode, the rate match FIFO is capable of compensating up to ±300 PPM (total 600 PPM total) difference between the upstream transmitter and the local receiver reference clock.

☞ To enable the rate match FIFO in Basic double-width mode, the transceiver channel must have both the transmitter and the receiver channel instantiated. You must select the **Receiver and Transmitter** option in the **What is the operation mode?** field in the ALTGX MegaWizard Plug-In Manager. You must also enable the 8B/10B encoder/decoder in Basic double-width mode with rate match FIFO enabled.

Depending on your proprietary protocol implementation, you can select two 20-bit rate match patterns in the ALTGX MegaWizard Plug-In Manager under the What is the rate match pattern1 and What is the rate match pattern2 fields. Each of the two programmed 20-bit rate match patterns consists of a 10-bit skip pattern and a 10-bit control pattern. You must choose 10-bit code groups that have a neutral disparity as the skip patterns. The rate match FIFO operation begins after the word aligner synchronization status `rx_syncstatus` goes high. When the rate matcher receives either of the two 10-bit control patterns followed by the respective 10-bit skip pattern, it inserts or deletes a pair of 10-bit skip patterns as necessary to avoid the rate match FIFO from overflowing or under running.

The rate match FIFO can delete as many pairs of skip patterns from a cluster necessary to avoid the rate match FIFO from overflowing. The rate match FIFO can delete a pair of skip patterns only if the two 10-bit skip patterns appear in the same clock cycle on the LSByte and MSByte of the 20-bit word. If the two skip patterns appear straddled on the MSByte of a clock cycle and the LSByte of the next clock cycle, the rate match FIFO cannot delete the pair of skip patterns. The rate match FIFO can insert as many pairs of skip patterns into a cluster necessary to avoid the rate match FIFO from under running. The 10-bit skip pattern can appear on MSByte or LSByte or both of the 20-bit word.

Two flags, `rx_rmfifodatadeleted` and `rx_rmfifodatainserted`, that indicate rate match FIFO deletion and insertion events, respectively, are forwarded to the FPGA fabric.

Figure 1–74 shows an example of rate match FIFO deletion in the case where three skip patterns are required to be deleted. In this example, /K28.5/ is the control pattern and neutral disparity /K28.0/ is the skip pattern. The first skip cluster has a /K28.5/ control pattern in the LSByte and /K28.0/ skip pattern in the MSByte of a clock cycle followed by one /K28.0/ skip pattern in the LSByte of the next clock cycle. The rate match FIFO cannot delete the two skip patterns in this skip cluster because they do not appear in the same clock cycle. The second skip cluster has a /K28.5/ control pattern in the MSByte of a clock cycle followed by two pairs of /K28.0/ skip patterns in the next two cycles. The rate match FIFO deletes both pairs of /K28.0/ skip patterns (for a total of four skip patterns deleted) from the second skip cluster to meet the three skip pattern deletion requirement.

Figure 1–74. Rate Match Deletion in Basic Double-Width Mode

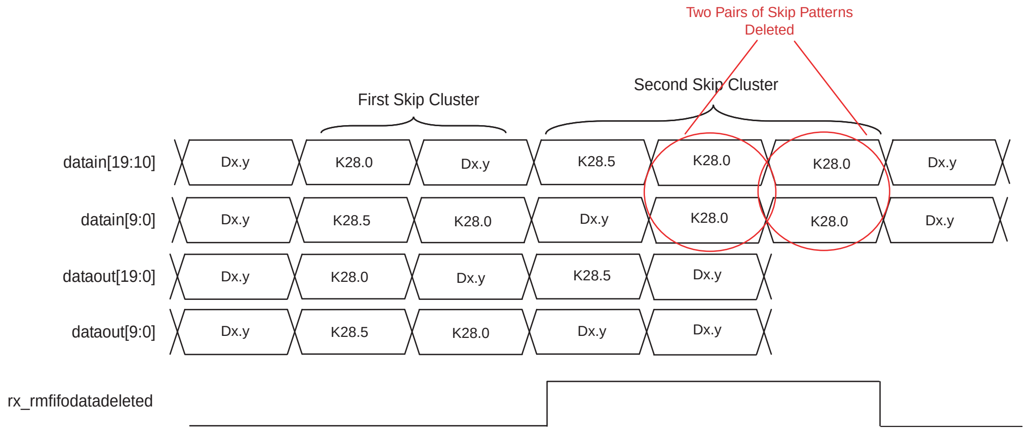
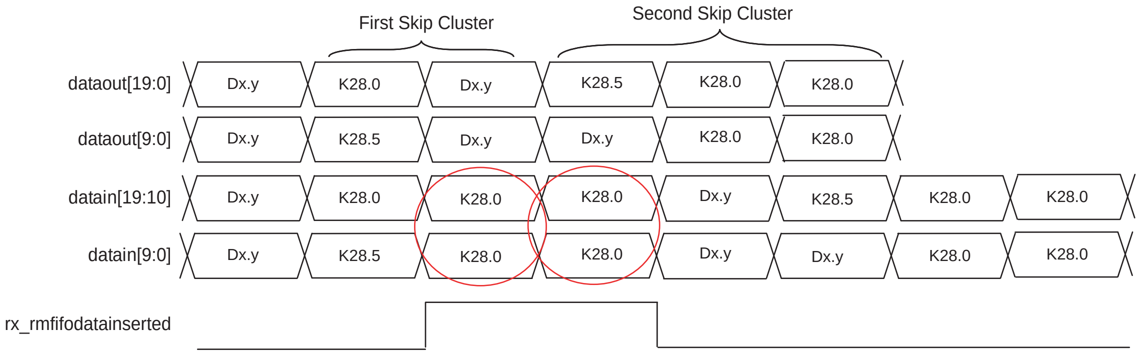


Figure 1–75 shows an example of rate match FIFO insertion in the case where three skip patterns are required to be inserted. In this example, /K28.5/ is the control pattern and neutral disparity /K28.0/ is the skip pattern. The first skip cluster has a /K28.5/ control pattern in the LSByte and /K28.0/ skip pattern in the MSByte of a clock cycle followed by one /K28.0/ skip pattern in the LSByte of the next clock cycle. The rate match FIFO inserts pairs of skip patterns in this skip cluster to meet the three skip pattern insertion requirement.

Figure 1–75. Rate Match Insertion in Basic Double-Width Mode

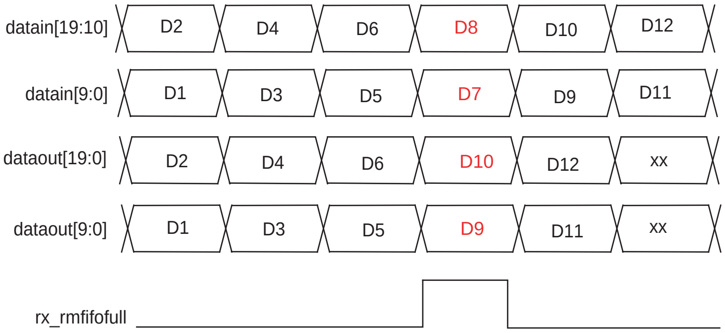


Two flags, `rx_rmfifofull` and `rx_rmfifoempty`, are forwarded to the FPGA fabric to indicate rate match FIFO full and empty conditions.

The rate match FIFO in Basic double-width mode automatically deletes the pair of data byte that causes the FIFO to go full and asserts the `rx_rmfifofull` flag synchronous to the subsequent pair of data bytes.

Figure 1–76 shows the rate match FIFO full condition in Basic double-width mode. The rate match FIFO becomes full after receiving the 20-bit word D5D6.

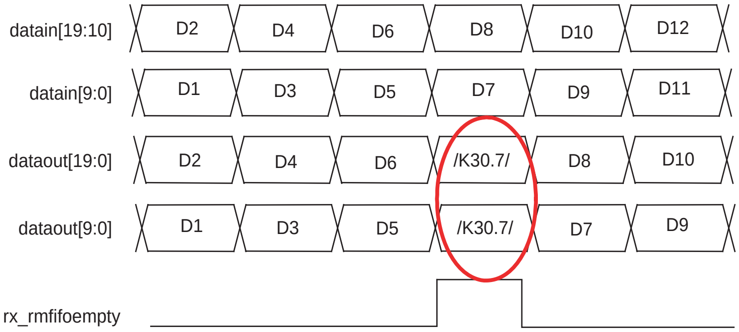
Figure 1–76. Rate Match FIFO Full Condition in Basic Double-Width Mode



The rate match FIFO automatically inserts a pair of /K30.7/ ({9'h1FE,9'h1FE}) after the data byte that causes the FIFO to go empty and asserts the `rx_fifoempty` flag synchronous to the inserted pair of /K30.7/ ({9'h1FE,9'h1FE}).

Figure 1–77 shows rate match FIFO empty condition in Basic double-width mode. The rate match FIFO becomes empty after reading out the 20-bit word D5D6.

Figure 1–77. Rate Match FIFO Empty Condition in Basic Double-Width Mode



8B/10B Decoder

Protocols like PCI Express (PIPE), XAUI, GIGE, and Serial RapidIO require the serial data sent over the link to be 8B/10B encoded to maintain the DC balance in the serial data transmitted. These protocols require the receiver PCS logic to implement an 8B/10B decoder to decode the data before forwarding it to the upper layers for packet processing.

The Stratix IV GX receiver channel PCS datapath implements the 8B/10B decoder after the rate matcher. In functional modes with rate matcher enabled, the 8B/10B decoder receives data from the rate matcher. In functional modes with rate matcher disabled, the 8B/10B decoder receives data from the word aligner.

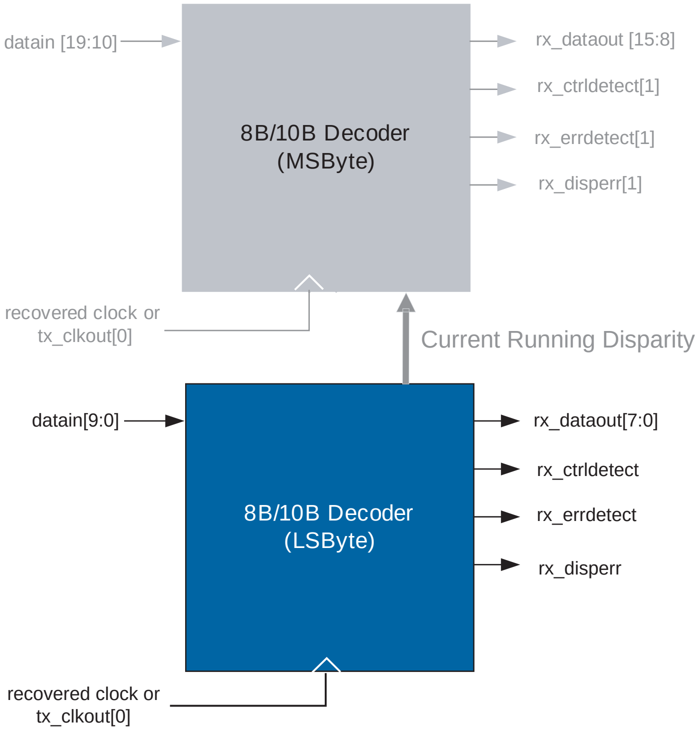
The 8B/10B decoder operates in two modes:

- Single-width mode
- Double-width mode

8B/10B Decoder in Single-Width Mode

Figure 1–78 shows the block diagram of the 8B/10B decoder in single-width mode.

Figure 1–78. 8B/10B Decoder in Single-Width Mode



In single-width mode, the 8B/10B decoder receives 10-bit data from the rate matcher or word aligner (when rate matcher is disabled) and decodes it into an 8-bit data + 1-bit control identifier. The decoded data is fed to the byte deserializer or the receiver phase compensation FIFO (if byte deserializer is disabled).

☞ The 8B/10B decoder is compliant to Clause 36 in the IEEE802.3 specification.

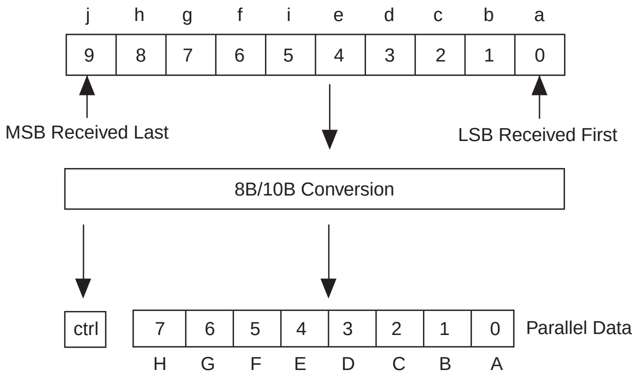
The 8B/10B decoder operates in single-width mode in the following functional modes:

- PCI Express (PIPE)
- XAUI
- GIGE
- Serial RapidIO
- Basic single-width

For PCI Express (PIPE), XAUI, GIGE, and Serial RapidIO functional modes, the ALTGX MegaWizard Plug-In Manager forces selection of the 8B/10B decoder in the receiver datapath. In Basic single-width mode, it allows you to enable or disable the 8B/10B decoder depending on your proprietary protocol implementation.

Figure 1–79 shows a 10-bit code group decoded into an 8-bit data and a 1-bit control identifier by the 8B/10B decoder in single-width mode.

Figure 1–79. 8B/10B Decoder in Single-Width Mode

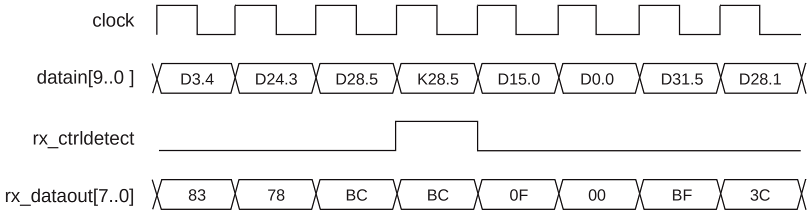


Control Code Group Detection

The 8B/10B decoder indicates whether the decoded 8-bit code group is a data or control code group on the `rx_ctrldetect` port. If the received 10-bit code group is one of the 12 control code groups (/Kx.y/) specified in the IEEE802.3 specification, the `rx_ctrldetect` signal is driven high. If the received 10-bit code group is a data code group (/Dx.y/), the `rx_ctrldetect` signal is driven low.

Figure 1–80 illustrates the 8B/10B decoder decoding the received 10-bit /K28.5/ control code group into an 8-bit data code group (8'hBC) driven on the `rx_dataout` port. `rx_ctrldetect` is asserted high synchronous with 8'hBC on the `rx_dataout` port, indicating that it is a control code group. The rest of the codes received are data code groups /Dx.y/.

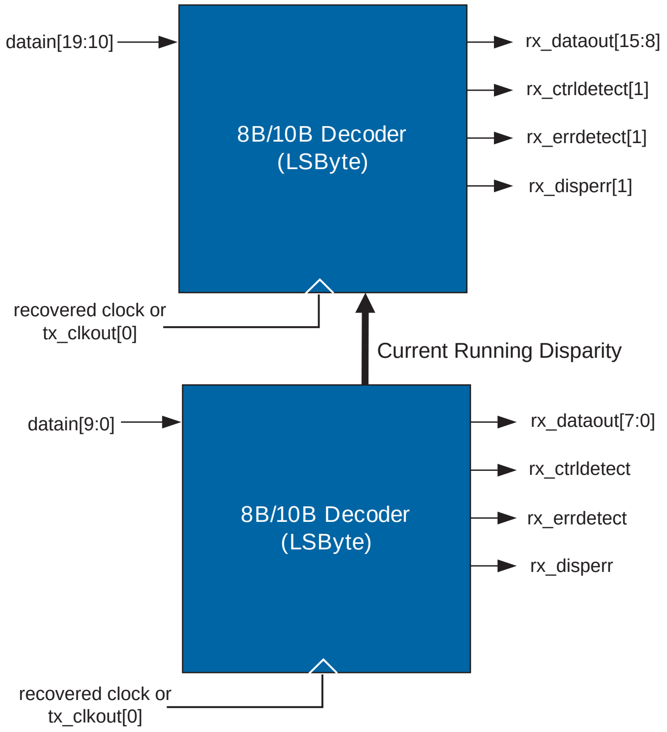
Figure 1–80. 8B/10B Decoder in Control Code Group Detection



8B/10B Decoder in Double-Width Mode

Figure 1–81 shows the block diagram of the 8B/10B decoder in double-width mode.

Figure 1–81. 8B/10B Decoder in Double-Width Mode



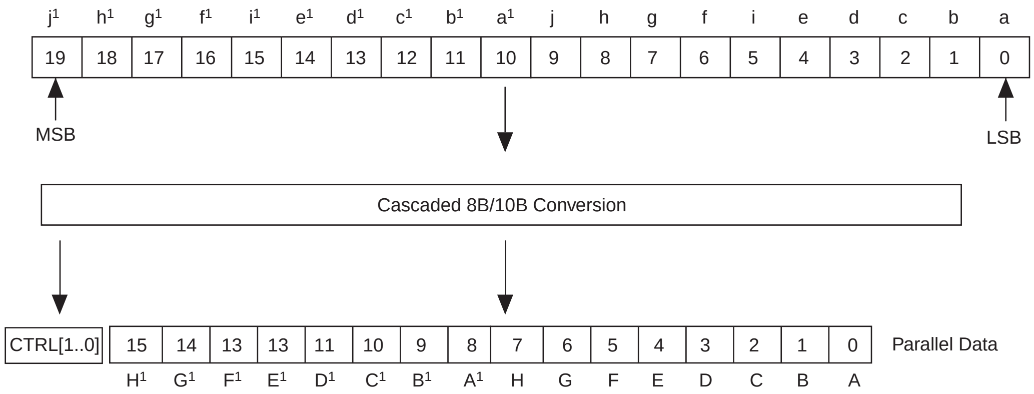
In double-width mode, two 8B/10B decoders are cascaded for decoding the 20-bit encoded data, as shown in Figure 1–82. The 10-bit LSByte of the received 20-bit encoded data is decoded first and the ending running disparity is forwarded to the 8B/10B decoder responsible for decoding the 10-bit MSByte. The cascaded 8B/10B decoder decodes the 20-bit encoded data into 16-bit data + 2-bit control identifier. The MSB and the LSB of the 2-bit control identifier corresponds to the MSByte and LSByte of the 16-bit decoded data code group. The decoded data is fed to the byte deserializer or the receiver phase compensation FIFO (if byte deserializer is disabled).

☞ Each of the two cascaded 8B/10B decoders is compliant to Clause 36 in the IEEE802.3 specification.

The 8B/10B decoder operates in double-width mode only in Basic double-width functional mode. You can enable or disable the 8B/10B decoder depending on your proprietary protocol implementation.

Figure 1–82 shows a 20-bit code group decoded into 16-bit data and 2-bit control identifier by the 8B/10B decoder in double-width mode.

Figure 1–82. 8B/10 Decoder in 20-Bit Double-Width Mode

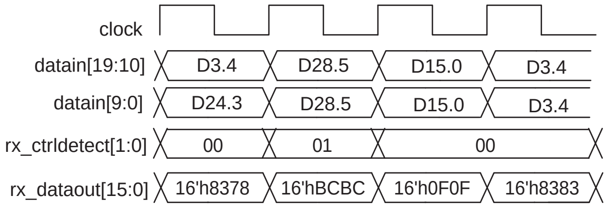

j^1 h^1 g^1 f^1 i^1 e^1 d^1 c^1 b^1 a^1 j h g f i e d c b a
19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
MSB
LSB
Cascaded 8B/10B Conversion
CTRL[1..0]
15 14 13 13 11 10 9 8 7 6 5 4 3 2 1 0
Parallel Data
H^1 G^1 F^1 E^1 D^1 C^1 B^1 A^1 H G F E D C B A

Control Code Group Detection

The cascaded 8B/10B decoder indicates whether the decoded 16-bit code group is a data or control code group on the 2-bit `rx_ctrldetect[1:0]` port. The `rx_ctrldetect[0]` signal is driven high or low depending on whether decoded data on the `rx_dataout[7:0]` port (LSByte) is a control or data code group, respectively. The `rx_ctrldetect[1]` signals are driven high or low depending on whether decoded data on the `rx_dataout[15:8]` port (MSByte) is a control or data code group, respectively.

Figure 1–83 shows the 8B/10B decoding of the received 10-bit /K28.5/ control code group into 8-bit data code group (8'hBC) driven on the `rx_dataout` port. The `rx_ctrldetect` is asserted high synchronous with 8'hBC on the `rx_dataout` port, indicating that it is a control code group. The rest of the codes received are data code groups /Dx.y/.

Figure 1–83. 8B/10B Decoder 10-Bit Control Code Group



Byte Deserializer

The FPGA fabric-transceiver interface frequency has an upper limit of 250 MHz. In functional modes that have a receiver PCS frequency greater than 250 MHz, the parallel received data and status signals cannot be forwarded directly to the FPGA fabric because it violates the upper limit of the 250 MHz FPGA fabric-transceiver interface frequency. In such configurations, the byte deserializer is required to reduce the FPGA fabric-transceiver interface frequency to half while doubling the parallel data width. For example, at 3.2 Gbps data rate with a deserialization factor of 10, the receiver PCS datapath runs at 320 MHz. The 10-bit parallel received data and status signals at 320 MHz cannot be forwarded to the FPGA fabric because it violates the upper limit of 250 MHz. The byte serializer converts the 10-bit parallel received data at 320 MHz into 20-bit parallel data at 160 MHz before forwarding to the FPGA fabric.

☞ The byte deserializer is required in configurations that exceed the FPGA fabric-transceiver interface clock upper frequency limit. It is optional in configurations that do not exceed the FPGA fabric-transceiver interface clock upper frequency limit.

The byte deserializer operates in two modes:

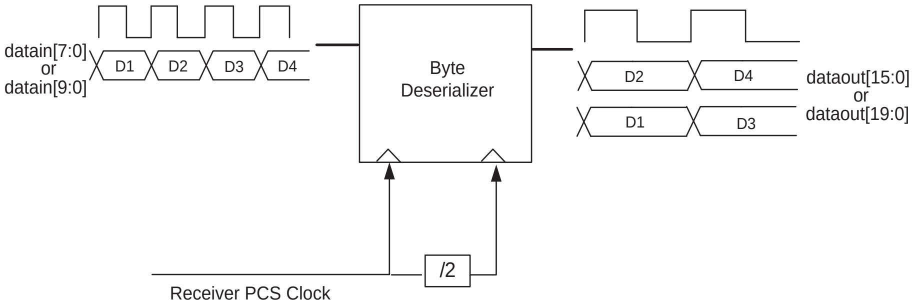
- Single-width mode
- Double-width mode

Byte Deserializer in Single-Width Mode

In single-width mode, the byte deserializer receives 8-bit wide data from the 8B/10B decoder or 10-bit wide data from the word aligner (if the 8B/10B decoder is disabled) and deserializes it into 16-bit or 20-bit wide data at half the speed.

Figure 1–84 shows the block diagram of the byte deserializer in single-width mode.

Figure 1–84. Byte Deserializer in Single-Width Mode

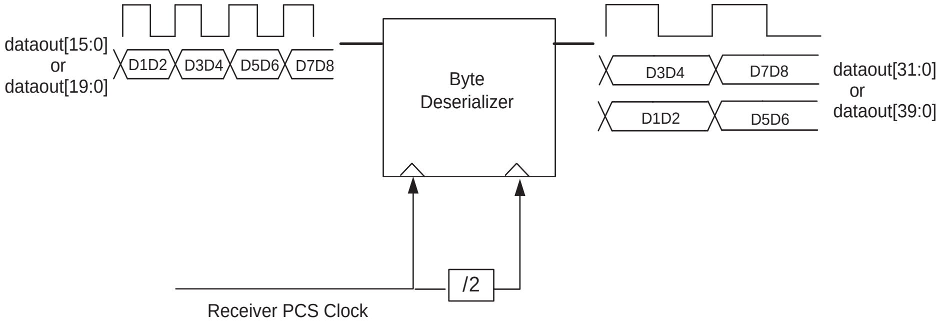


Byte Deserializer in Double-Width Mode

In double-width mode, the byte deserializer receives 16-bit wide data from the 8B/10B decoder or 20-bit wide data from the word aligner (if the 8B/10B decoder is disabled) and deserializes it into 32-bit or 40-bit wide data at half the speed.

Figure 1–85 shows the block diagram of the byte deserializer in double-width mode

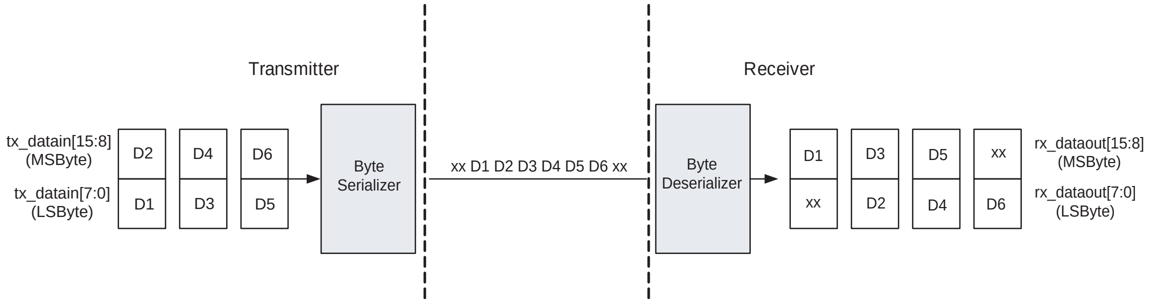
Figure 1–85. Byte Deserializer in Double-Width Mode



Byte Ordering Block

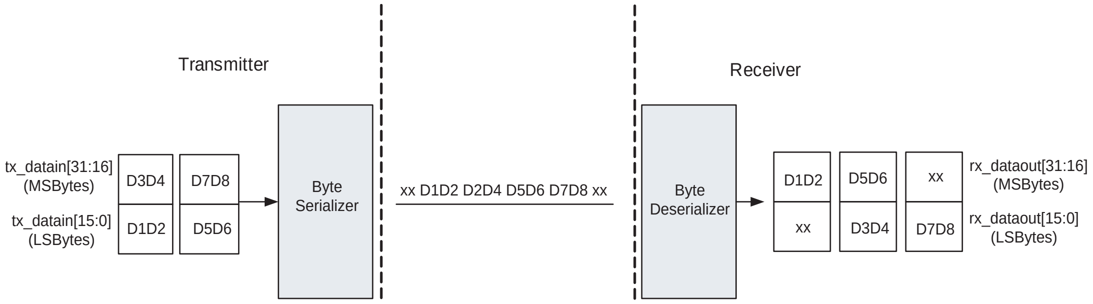
In single-width modes with the 16-bit or 20-bit FPGA fabric-transceiver interface, the byte deserializer receives one data byte (8- or 10-bit) and deserializes it into two data bytes (16- or 20-bit). Depending on when the receiver PCS logic comes out of reset, the byte ordering at the output of the byte deserializer may or may not match the original byte ordering of the transmitted data. The byte misalignment resulting from byte deserialization is unpredictable because it depends on which byte is being received by the byte deserializer when it comes out of reset. Figure 1–86 shows a scenario in which the MSByte and LSByte of the two-byte transmitter data appears straddled across two word boundaries after getting byte deserialized at the receiver.

Figure 1–86. MSByte and LSByte of the Two-Bit Transmitter Data Straddled Across Two Word Boundaries



In double-width modes with the 32-bit or 40-bit FPGA fabric-transceiver interface, the byte deserializer receives two data bytes (16- or 20-bit) and deserializers it into four data bytes (32- or 40-bit). Figure 1–87 shows a scenario in which the two MSBytes and LSBytes of the four-byte transmitter data appears straddled across two word boundaries after getting byte deserialized at the receiver.

Figure 1–87. MSByte and LSByte of the Four-Bit Transmitter Data Straddled Across Two Word Boundaries



The Stratix IV GX transceivers have an optional byte ordering block in the receiver data path that you can use to restore proper byte ordering before forwarding the data to the FPGA fabric. The byte ordering block looks for the user-programmed byte ordering pattern in the byte-deserialized data. You must select a byte ordering pattern that you know appears at the LSByte(s) position of the parallel transmitter data. If the byte ordering block finds the programmed byte ordering pattern in the MSByte(s) position of the byte-deserialized data, it inserts the appropriate number of user-programmed PAD bytes to push the byte ordering pattern to the LSByte(s) position, thereby restoring proper byte ordering.

Byte Ordering Block in Single-Width Modes

The byte ordering block is available in the following single-width functional modes:

- SONET/SDH OC-48
- Basic single-width mode with:
  - 16-bit FPGA fabric-transceiver interface
  - No 8B/10B decoder (8-bit PMA-PCS interface)
  - Word aligner in manual alignment mode
- Basic single-width mode with:
  - 16-bit FPGA fabric-transceiver interface
  - 8B/10B decoder
  - Word aligner in automatic synchronization state machine mode

☞ Refer to "Basic Single-Width Mode Configurations" on page 1–110 for more information about configurations that allow the byte ordering block in the receiver datapath.

The Quartus II software automatically configures the byte ordering pattern and byte ordering PAD pattern for SONET/SDH OC-48 functional mode. Refer to "OC-48 Byte Ordering" on page 1–162 for more information.

In Basic single-width mode, you can program a custom byte ordering pattern and byte ordering PAD pattern in the ALTGX MegaWizard Plug-In Manager. Table 1–26 shows the byte ordering pattern length allowed in Basic single-width mode.

Table 1–26. Byte Ordering Pattern Length in Basic Single-Width Mode

Functional Mode	Byte Ordering Pattern Length	Byte Ordering PAD Pattern Length
Basic single-width mode with: ■ 16-bit FPGA fabric-transceiver interface ■ no 8B/10B decoder ■ Word aligner in manual alignment mode	8-bit	8-bit
Basic single-width mode with: ■ 16-bit FPGA fabric-transceiver interface ■ 8B/10B decoder ■ Word aligner in automatic synchronization state machine mode	9-bit *(1)*	9-bit

Note to Table 1–26:

(1) If a /Kx.y/ control code group is selected as the byte ordering pattern, the MSB of the 9-bit byte ordering pattern must be 1'b1. If a /Dx.y/ data code group is selected as the byte ordering pattern, the MSB of the 9-bit byte ordering pattern must be 1'b0. The least significant 8 bits must be the 8B/10B decoded version of the code group used for byte ordering.

Byte Ordering Block in Double-Width Modes

The byte ordering block is available in the following double-width functional modes:

- Basic double-width mode with
  - 32-bit FPGA fabric-transceiver interface
  - No 8B/10B decoder (16-bit PMA-PCS interface)
  - Word aligner in manual alignment mode
- Basic double-width mode with
  - 32-bit FPGA fabric-transceiver interface
  - 8B/10B decoder (20-bit PMA-PCS interface)
  - Word aligner in manual alignment mode
- Basic double-width mode with
  - 40-bit FPGA fabric-transceiver interface
  - No 8B/10B decoder (20-bit PMA-PCS interface)
  - Word aligner in manual alignment mode

☞ For more information on configurations that allow byte ordering block in the receiver datapath, refer to "Basic Double-Width Mode Configurations" on page 1–112.

In Basic double-width modes, you can program a custom byte ordering pattern and byte ordering PAD pattern in the ALTGX MegaWizard Plug-In Manager. Table 1–27 shows the byte ordering pattern length allowed in Basic double-width mode.

Table 1–27. Byte Ordering Pattern Length in Basic Double-Width Mode

Functional Mode	Byte Ordering Pattern Length	Byte Ordering PAD Pattern Length
Basic double-width mode with: ■ 32-bit FPGA fabric-transceiver interface ■ No 8B/10B decoder (16-bit PMA-PCS interface) ■ Word aligner in manual alignment mode	16-bit, 8-bit	8-bit
Basic double-width mode with: ■ 32-bit FPGA fabric-transceiver interface ■ 8B/10B decoder (20-bit PMA-PCS interface) ■ Word aligner in manual alignment mode	18-bit, 9-bit *(1)*	9-bit
Basic double-width mode with: ■ 40-bit FPGA fabric-transceiver interface ■ No 8B/10B decoder (20-bit PMA-PCS interface) ■ Word aligner in manual alignment mode	20-bit, 10-bit	10-bit

Note to Table 1–27:

(1) The 18-bit byte ordering pattern `D[17:0]` consists of MSByte `D[17:9]` and LSByte `D[8:0]`. `D[17]` corresponds to `rx_ctrldetect[1]` and `D[16:9]` corresponds to `rx_dataout[15:8]`. Similarly `D[9]` corresponds to `rx_ctrldetect[0]` and `D[7:0]` corresponds to `rx_dataout[7:0]`.

The byte ordering block modes of operation in both single-width and double-width modes:

- Word-alignment-based byte ordering

- User-controlled byte ordering

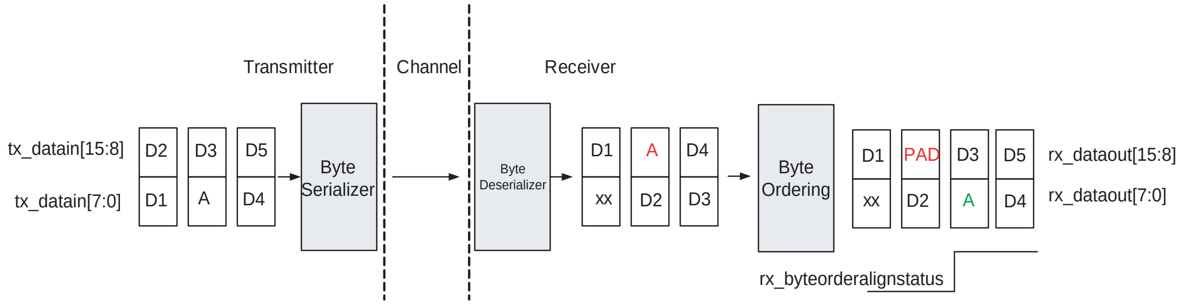
Word-Alignment-Based Byte Ordering

In word-alignment-based byte ordering, the byte ordering block starts looking for the byte ordering pattern in the byte-deserialized data every time it sees a rising edge on the rx_syncstatus signal. After a rising edge on the rx_syncstatus signal, if the byte ordering block finds the first data byte that matches the programmed byte ordering pattern in the MSByte position of the byte-deserialized data, it inserts one programmed PAD pattern to push the byte ordering pattern in the LSByte position. If the byte ordering blocks finds the first data byte that matches the programmed byte ordering pattern in the LSByte position of the byte-deserialized data, it considers the data to be byte ordered and does not insert any PAD pattern. In either case, the byte ordering block asserts the rx_byteorderalignstatus signal.

☞ You can choose word-alignment-based byte ordering by selecting the sync status signal from the word aligner option in the **What do you want the byte ordering to be based on?** field in the ALTGX MegaWizard Plug-In Manager.

Figure 1–88 shows an example of the byte ordering operation in single-width modes. In this example, A is the programmed byte ordering pattern and PAD is the programmed PAD pattern. The byte deserialized data places the byte ordering pattern A in the MSByte position resulting in incorrect byte ordering. Assuming that a rising edge on the rx_syncstatus signal had occurred before the byte ordering block sees the byte ordering pattern A in the MSByte position, the byte ordering block inserts a PAD byte and pushes the byte ordering pattern A in the LSByte position. The data at the output of the byte ordering block has correct byte ordering as reflected on the rx_byteorderalignstatus signal.

Figure 1–88. Byte Ordering in Single-Width Modes



If the byte ordering block sees another rising edge on the rx_syncstatus signal from the word aligner, it de-asserts the rx_byteorderalignstatus signal and repeats the byte ordering operation as previously discussed.

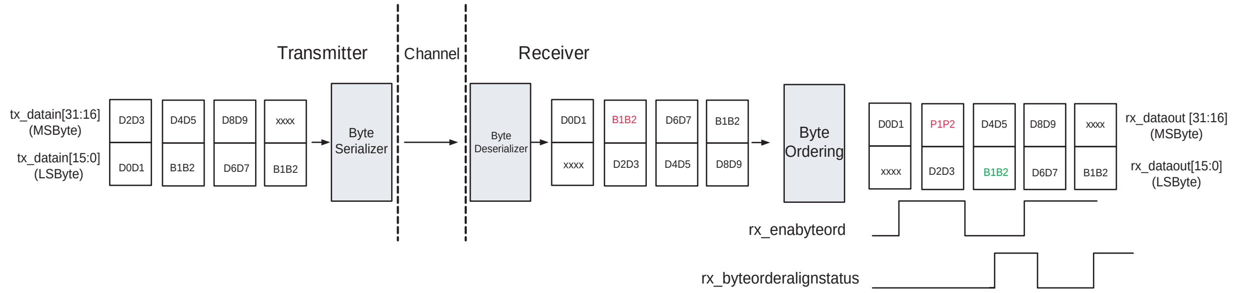
User-Controlled Byte Ordering

Unlike word-alignment-based byte ordering, user-controlled byte ordering provides control to the user logic to restore correct byte ordering at the receiver. When enabled, an rx_enabyteord port is available that you can use to trigger the byte ordering operation. A rising edge on the rx_enabyteord port triggers the byte ordering block. After a rising edge on the rx_enabyteord signal, if the byte ordering block finds the first data byte that matches the programmed byte ordering pattern in the

MSByte position of the byte-deserialized data, it inserts one programmed PAD pattern to push the byte ordering pattern in the LSByte position. If the byte ordering blocks finds the first data byte that matches the programmed byte ordering pattern in the LSByte position of the byte-deserialized data, it considers the data to be byte ordered and does not insert any PAD byte. In either case, the byte ordering block asserts the `rx_byteorderalignstatus` signal.

Figure 1–89 shows user-controlled byte ordering in Basic double-width Mode.

Figure 1–89. User-Controlled Byte Ordering in Basic Double-Width Mode



Receiver Phase Compensation FIFO

The receiver phase compensation FIFO in each channel ensures reliable transfer of data and status signals between the receiver channel and the FPGA fabric. The receiver phase compensation FIFO compensates for the phase difference between the parallel receiver PCS clock (FIFO write clock) and the FPGA fabric clock (FIFO read clock).

The receiver phase compensation FIFO operates in one of the following two modes:

- Low latency mode—The Quartus II software automatically configures the receiver phase compensation FIFO in low latency mode in all functional modes. In this mode, the FIFO is four words deep and the latency through the FIFO is 2-to-3 parallel clock cycles (pending characterization).
- High latency mode—In this mode, the FIFO is eight words deep and the latency through the FIFO is 4-to-5 parallel clock cycles (pending characterization).

The receiver phase compensation FIFO write clock source varies with the receiver channel configuration. Table 1–28 shows the receiver phase compensation FIFO write clock source in different configurations.

Table 1–28. Receiver Phase Compensation FIFO Write Clock Source (Part 1 of 2)

Configuration	Receiver Phase Compensation FIFO Write Clock	
	Without Byte Serializer	**With Byte Serializer**
Non-bonded channel configuration with rate matcher	Parallel transmitter PCS clock from the local clock divider in the associated channel (`tx_clkout`)	Divide-by-two version of the parallel transmitter PCS clock from the local clock divider in the associated channel (`tx_clkout`)
Non-bonded channel configuration without rate matcher	Parallel recovered clock from the receiver PMA in the associated channel (`rx_clkout`)	Divide-by-two version of the parallel recovered clock from the receiver PMA in the associated channel (`rx_clkout`)

Table 1–28. Receiver Phase Compensation FIFO Write Clock Source (Part 2 of 2)

	Receiver Phase Compensation FIFO Write Clock	
Configuration	**Without Byte Serializer**	**With Byte Serializer**
×4 bonded channel configuration	Parallel transmitter PCS clock from the central clock divider in the CMU0 of the associated transceiver block (`coreclkout`)	Divide-by-two version of the parallel transmitter PCS clock from the central clock divider in CMU0 of the associated transceiver block (`coreclkout`)
×8 bonded channel configuration	Parallel transmitter PCS clock from the central clock divider in CMU0 of the master transceiver block (`coreclkout` from master transceiver block)	Divide-by-two version of the parallel transmitter PCS clock from the central clock divider in CMU0 of the master transceiver block (`coreclkout` from master transceiver block)

The receiver phase compensation FIFO read clock source varies depending on whether or not you instantiate the `rx_coreclk` port in the ALTGX MegaWizard Plug-In Manager. Table 1–29 shows the receiver phase compensation FIFO read clock source in different configurations.

Table 1–29. Receiver Phase Compensation FIFO Read Clock Source

	Receiver Phase Compensation FIFO Read Clock	
Configuration	**rx_coreclk Port Not Instantiated**	**rx_coreclk Port Instantiated *(1)***
Non-bonded channel configuration with rate matcher	FPGA fabric clock driven by the clock signal on the `tx_clkout` port	FPGA fabric clock driven by the clock signal on the `rx_coreclk` port
Non-bonded channel configuration without rate matcher	FPGA fabric clock driven by the clock signal on the `rx_clkout` port	FPGA fabric clock driven by the clock signal on the `rx_coreclk` port
×4 bonded channel configuration	FPGA fabric clock driven by the clock signal on the `coreclkout` port	FPGA fabric clock driven by the clock signal on the `rx_coreclk` port
×8 bonded channel configuration	FPGA fabric clock driven by the clock signal on the `coreclkout` port	FPGA fabric clock driven by the clock signal on the `rx_coreclk` port

Note to Table 1–29:

(1) The clock signal driven on the `rx_coreclk` port must have 0 PPM frequency difference with respect to the receiver phase compensation FIFO write clock.

Receiver Phase Compensation FIFO Error Flag

An optional `rx_phase_comp_fifo_error` port is available in all functional modes to indicate a receiver phase compensation FIFO overrun or underflow condition. The `rx_phase_comp_fifo_error` signal is asserted high when the phase compensation FIFO gets either full or empty. This feature is useful to verify a phase compensation FIFO overrun or underflow condition as a probable cause of link errors.

Offset Cancellation in the Receiver Buffer and Receiver CDR

Offset Cancellation

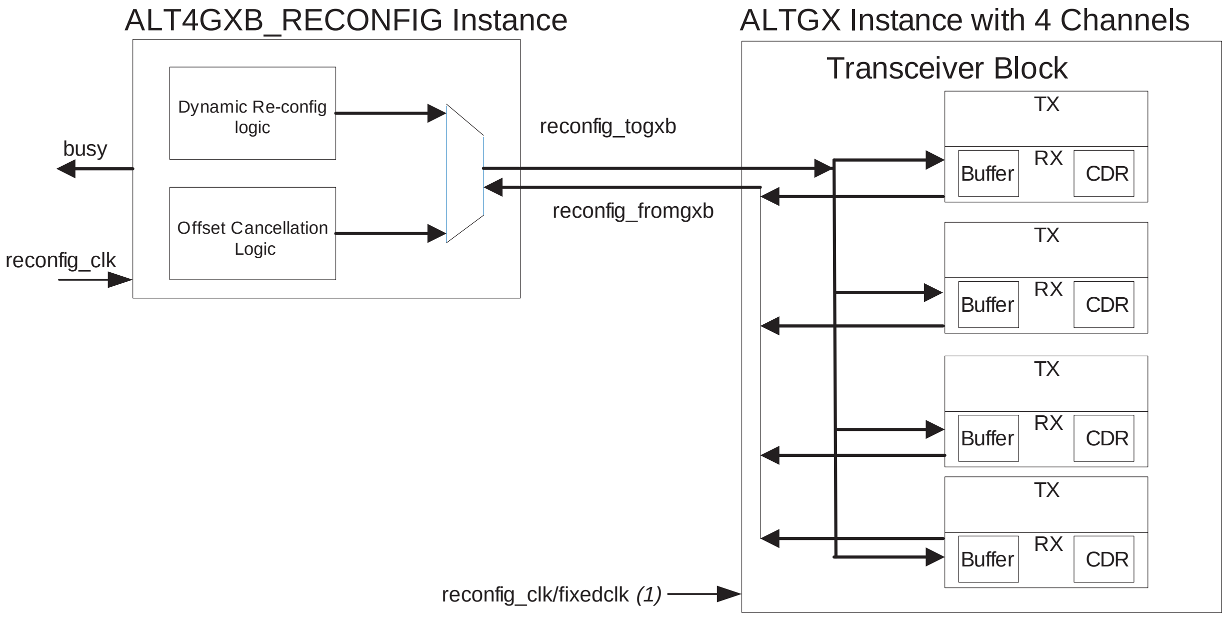
As silicon progresses towards smaller process nodes, the performance of circuits at these smaller nodes depends more on the process variations. These process variations result in analog voltages that can be offset from required ranges. Offset cancellation logic corrects these offsets. The receiver buffer and receiver CDR require offset cancellation.

Offset cancellation is executed automatically once each time a Stratix IV device is powered on. The control logic for offset cancellation is integrated into the ALT4GXB_RECONFIG megafunction. To use this logic, you need to enable the offset cancellation option in the ALT4GXB_RECONFIG MegaWizard Plug-In Manager GUI. Additionally, the `reconfig_fromgxb` and `reconfig_togxb` busses and the necessary clocks need to be connected between the ALTGX instance and the ALT4GXB_RECONFIG instance.

☞ For additional information about offset cancellation control logic connectivity, refer to the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*.

Offset cancellation logic requires a separate clock. In PCI Express (PIPE) mode, you must connect the clock input to the `fixedclk` port provided by the ALTGX MegaWizard Plug-In Manager. The frequency of this clock input must be 125 MHz. For all other functional modes, connect the clock input to the `reconfig_clk` port provided by the ALTGX MegaWizard Plug-In Manager. The frequency of the clock connected to the `reconfig_clk` port must be within the range of 37.5 to 50 MHz. Figure 1–90 shows the interface of the offset cancellation control logic (ALT4GXB_RECONFIG instance) and ALTGX instance.

Figure 1–90. Interface of Offset Cancellation Control Logic to ALTGX



Note to Figure 1–90:

(1) The `fixedclk` port is only applicable for PCI Express (PIPE) functional mode.

Offset Cancellation Process

The offset cancellation process begins by disconnecting the path from the receiver input buffer to the receiver CDR. It then sets the receiver CDR into a fixed set of dividers to guarantee a VCO clock rate that is within the range necessary to provide the proper offset cancellation. Subsequently, the offset cancellation process goes through various states and culminates in the offset cancellation of the receiver buffer and the receiver CDR.

After offset cancellation is complete, the user's divider settings are restored. Then, the reconfiguration block sends and receives data to the ALT4GXB via the `reconfig_togxb` and `reconfig_fromgxb` busses. These busses are connected by the user between ALT4GXB_RECONFIG and ALT4GXB instances. The de-assertion of the "busy" signal from the offset cancellation control logic indicates that the offset cancellation process is complete.

Due to the offset cancellation process, the transceiver reset sequence has changed. Refer to the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook* for more information on the offset cancellation process.

Functional Modes

You can configure Stratix IV GX transceivers in one of the following functional modes using the ALTGX MegaWizard Plug-In Manager:

- Basic
  - Basic single-width at 600 Mbps to 3.75 Gbps
  - Basic double-width at 1 Gbps to 8.5 Gbps

- PCI Express (PIPE) (Gen1 at 2.5 Gbps and Gen2 at 5 Gbps)
- XAUI (3.125 Gbps up to HiGig at 3.75 Gbps)
- GIGE (1.25 Gbps)
- Serial RapidIO (1.25 Gbps, 2.5 Gbps, 3.125 Gbps)
- SONET/SDH (OC-12, OC-48)
- (OIF) CEI PHY Interface (>3.135 Gbps to 6.375 Gbps)
- SDI (HD at 1.485/1.4835 Gbps, 3G at 2.97/2.967 Gbps)

Basic Functional Mode

The Stratix IV GX transceiver datapath is extremely flexible in Basic functional mode. To configure the transceiver in Basic functional mode, you must select Basic in the **Which protocol will you be using?** option of the ALTGX MegaWizard Plug-In Manager.

The Basic functional mode can be further sub-divided into the following two functional modes:

- Basic single-width mode
- Basic double-width mode

You can configure the transceiver in Basic single-width mode by selecting **Single** in the **What is the deserializer block width?** option in the ALTGX MegaWizard Plug-In Manager. You can configure the transceiver in Basic double-width mode by selecting **Double** in the **What is the deserializer block width?** option in the ALTGX MegaWizard Plug-In Manager.

Table 1–30 shows the PCS-PMA interface widths and data rates supported in Basic single-width and double-width modes.

Table 1–30. PCS-PMA Interface Widths and Data Rates in Basic Single-Width and Double-Width Modes

Basic Functional Mode	Supported Data Rate Range *(1)*	PMA-PCS Interface Width
Basic single-width mode	600 Mbps to 3.75 Gbps	8-bit
		10-bit
Basic double-width mode	1 Gbps to 8.5 Gbps	16-bit
		20-bit

Note to Table 1–30:

(1) The data rate range supported in Basic single-width and double-width modes varies depending on whether or not you use the byte serializer/deserializer. Refer to "Basic Single-Width Mode Configurations" on page 1–110 and "Basic Double-Width Mode Configurations" on page 1–112 for more information.

Low Latency PCS Datapath

The ALTGX MegaWizard Plug-In Manager provides an **Enable low latency PCS mode** option when configured in Basic single-width or double-width mode. If you select this option, the following transmitter and receiver channel PCS blocks are truly bypassed to yield a low latency PCS datapath:

- 8B/10B encoder and decoder

- Word aligner
- Deskew FIFO
- Rate match (clock rate compensation) FIFO
- Byte ordering

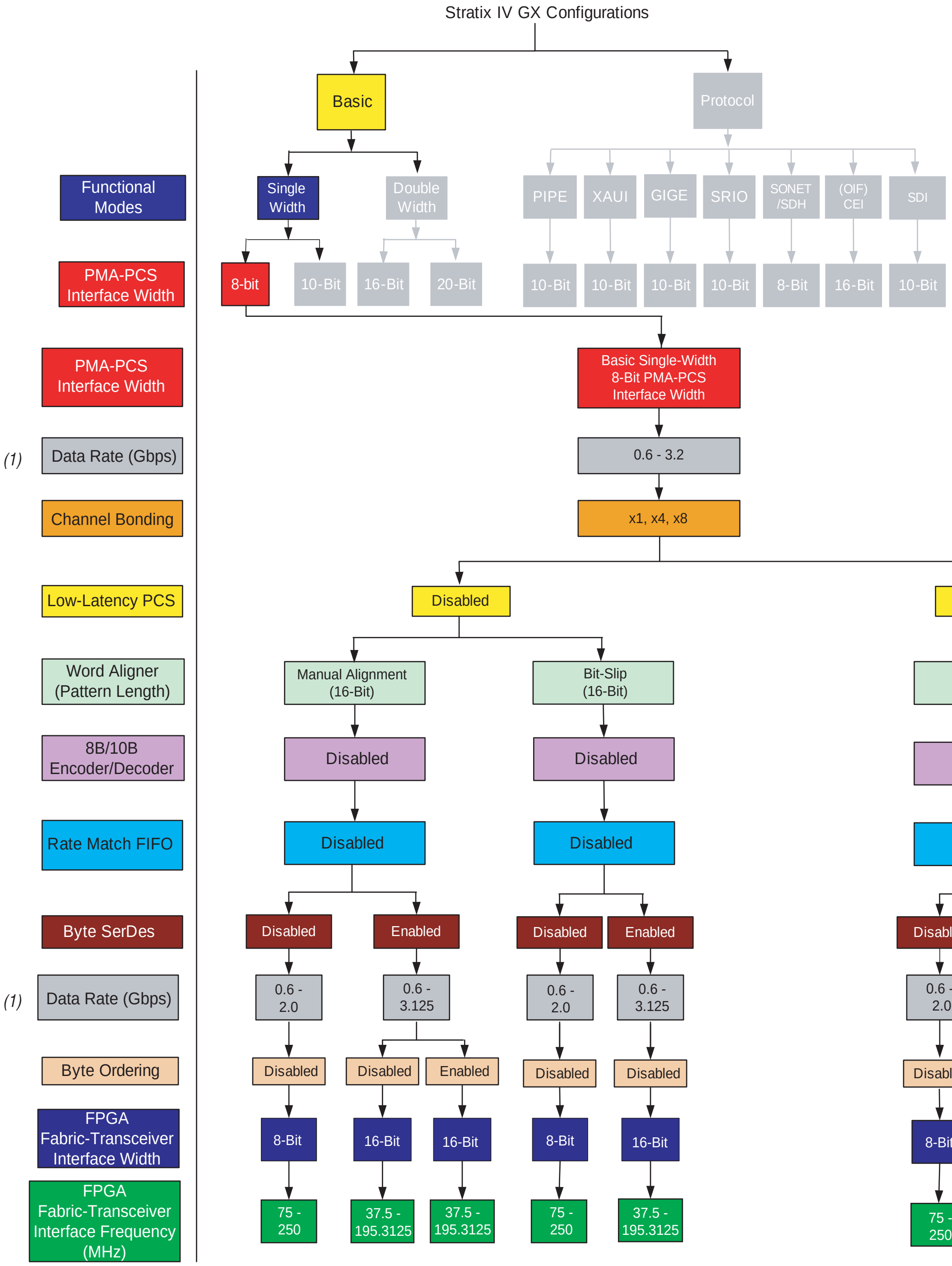
In low latency PCS modes, the transmitter and receiver phase compensation FIFOs are always enabled. Depending on the targeted data rate, the byte serializer and deserializer blocks can be optionally bypassed. Refer to "Basic Single-Width Mode Configurations" on page 1–110 and "Basic Double-Width Mode Configurations" on page 1–112 for more information.

☞ The PCS latency in Basic single-width and double-width modes with and without the low latency PCS mode option is pending characterization.

Basic Single-Width Mode Configurations

Figure 1–91 shows Stratix IV GX transceiver configurations allowed in Basic single-width functional mode with an 8-bit wide PMA-PCS interface.

Figure 1–91. Transceiver Configurations in Basic Single-Width Mode with an 8-Bit Wide PMA-PCS Interface

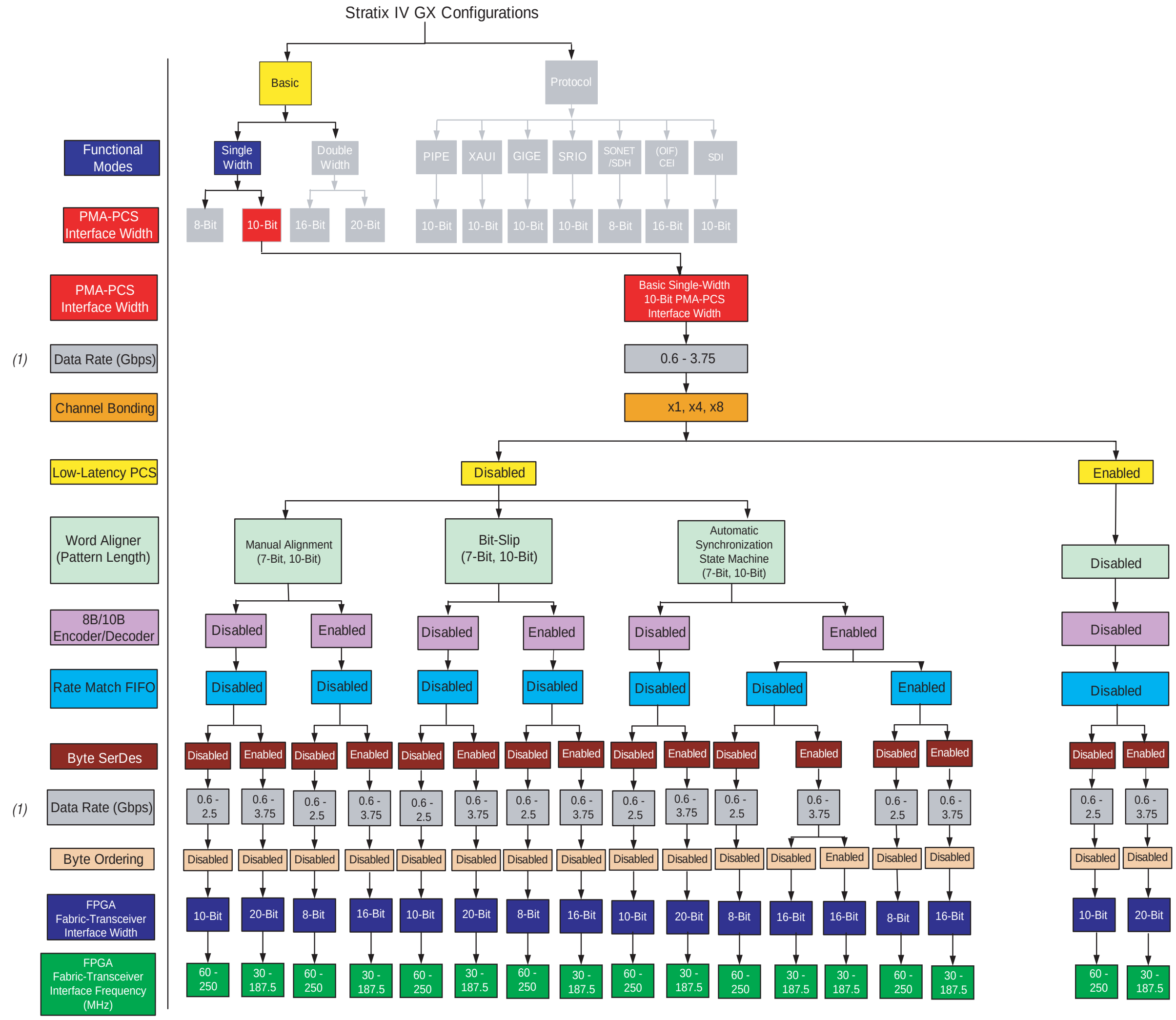


Note to Figure 1–91:

(1) The maximum data rate specification shown in Figure 1–91 are valid only for the -2 (fastest) speed grade devices. Refer to *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook* for data rate specifications for other speed grades offered.

Figure 1–92 shows Stratix IV GX transceiver configurations allowed in Basic single-width functional mode with a 10-bit wide PMA-PCS interface.

Figure 1–92. Transceiver Configurations in Basic Single-Width Mode with a 10-Bit Wide PMA-PCS Interface



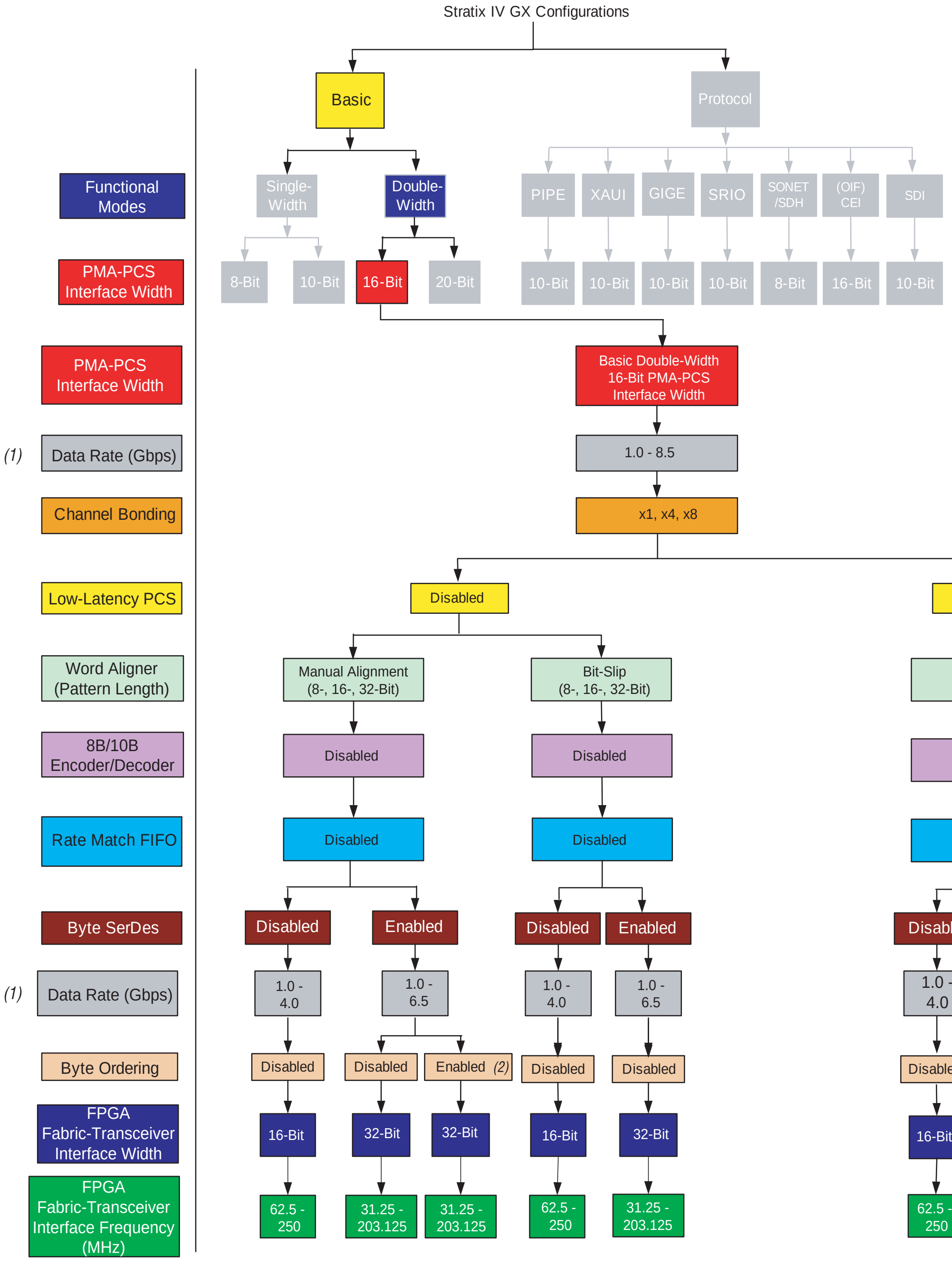
Note to Figure 1–92:

(1) The maximum data rate specification shown in Figure 1–92 are valid only for the -2 (fastest) speed grade devices. Refer to *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook* for data rate specifications for other speed grades offered.

Basic Double-Width Mode Configurations

Figure 1–93 shows Stratix IV GX transceiver configurations allowed in Basic double-width functional mode with a 16-bit wide PMA-PCS interface.

Figure 1–93. Transceiver Configurations in Basic Double-Width Mode with a 16-Bit Wide PMA-PCS Interface



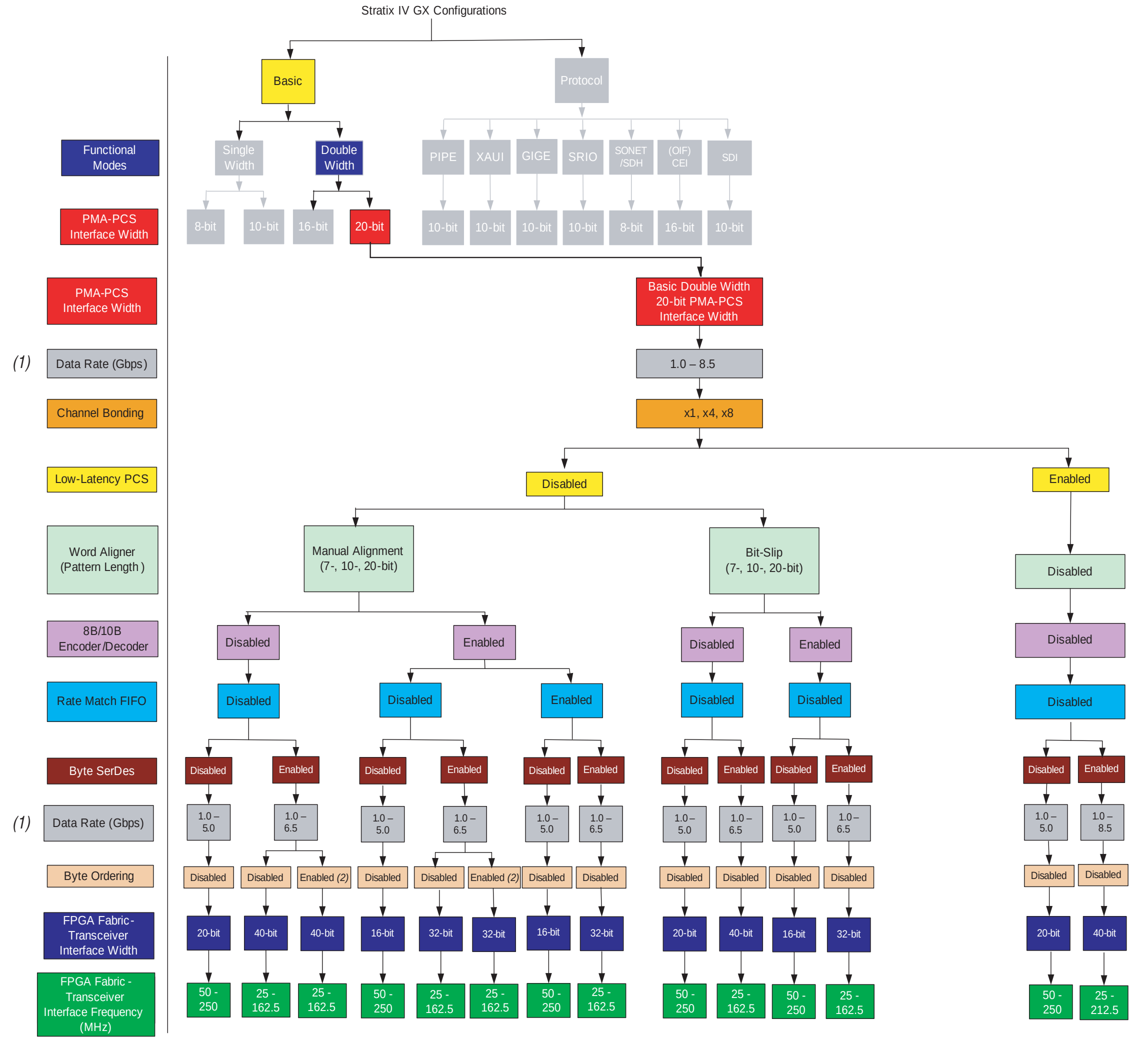
Note to Figure 1–93:

(1) The maximum data rate specification shown in Figure 1–93 are valid only for the -2 (fastest) speed grade devices. Refer to *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook* for data rate specifications for other speed grades offered.

(2) The byte ordering block is available only if you select the word alignment pattern length of 16 or 32 bits.

Figure 1–94 shows Stratix IV GX transceiver configurations allowed in Basic double-width functional mode with a 20-bit wide PMA-PCS interface.

Figure 1–94. Transceiver Configurations in Basic Double-Width Mode with a 20-Bit Wide PMA-PCS Interface



Note to Figure 1–94:

(1) The maximum data rate specification shown in Figure 1–94 are valid only for the -2 (fastest) speed grade devices. Refer to *DC and Switching Characteristics* chapter in volume 4 of the *Stratix IV Device Handbook* for data rate specifications for other speed grades offered.

(2) The byte ordering block is available only if you select the word alignment pattern length of 20 bits.

PCI Express (PIPE) Mode

Intel Corporation has developed a PHY interface for the PCI Express Architecture (PIPE) specification to enable implementation of a PCI Express-compliant physical layer device. The PIPE specification also defines a standard interface between the physical layer device and the media access control layer (MAC). Version 2.00 of the PIPE specification provides implementation details for a PCI Express-compliant physical layer device at both Gen1 (2.5 GT/s) and Gen2 (5 GT/s) signaling rates.

To implement a Version 2.00 PIPE compliant PHY, you must configure the Stratix IV GX transceivers in PCI Express (PIPE) functional mode. Stratix IV GX devices have built-in PCI Express hard IP blocks that you can use to implement the PHY-MAC layer, Data Link layer, and Transaction layer of the PCI Express protocol stack. You can also bypass the PCI Express hard IP blocks and implement the PHY-MAC layer, Data Link layer, and Transaction layer in the FGPA fabric using a Soft IP. If you enable the PCI Express hard IP blocks, the Stratix IV GX transceivers interface with these hard IP blocks. Otherwise, the Stratix IV GX transceivers interface with the FPGA fabric.

You can configure the Stratix IV GX transceivers in PCI Express (PIPE) functional mode using one of the following two methods:

- ALTGX MegaWizard Plug-In Manager if you do not use the PCI Express hard IP block
- PCI Express Compiler if you use the PCI Express hard IP block

☞ Description of PCI Express hard IP architecture and PCI Express (PIPE) mode configurations allowed when using PCI Express hard IP block are out of the scope of this chapter. For more information about the PCI Express hard IP block, refer to the *PCI Express Compiler User Guide.*

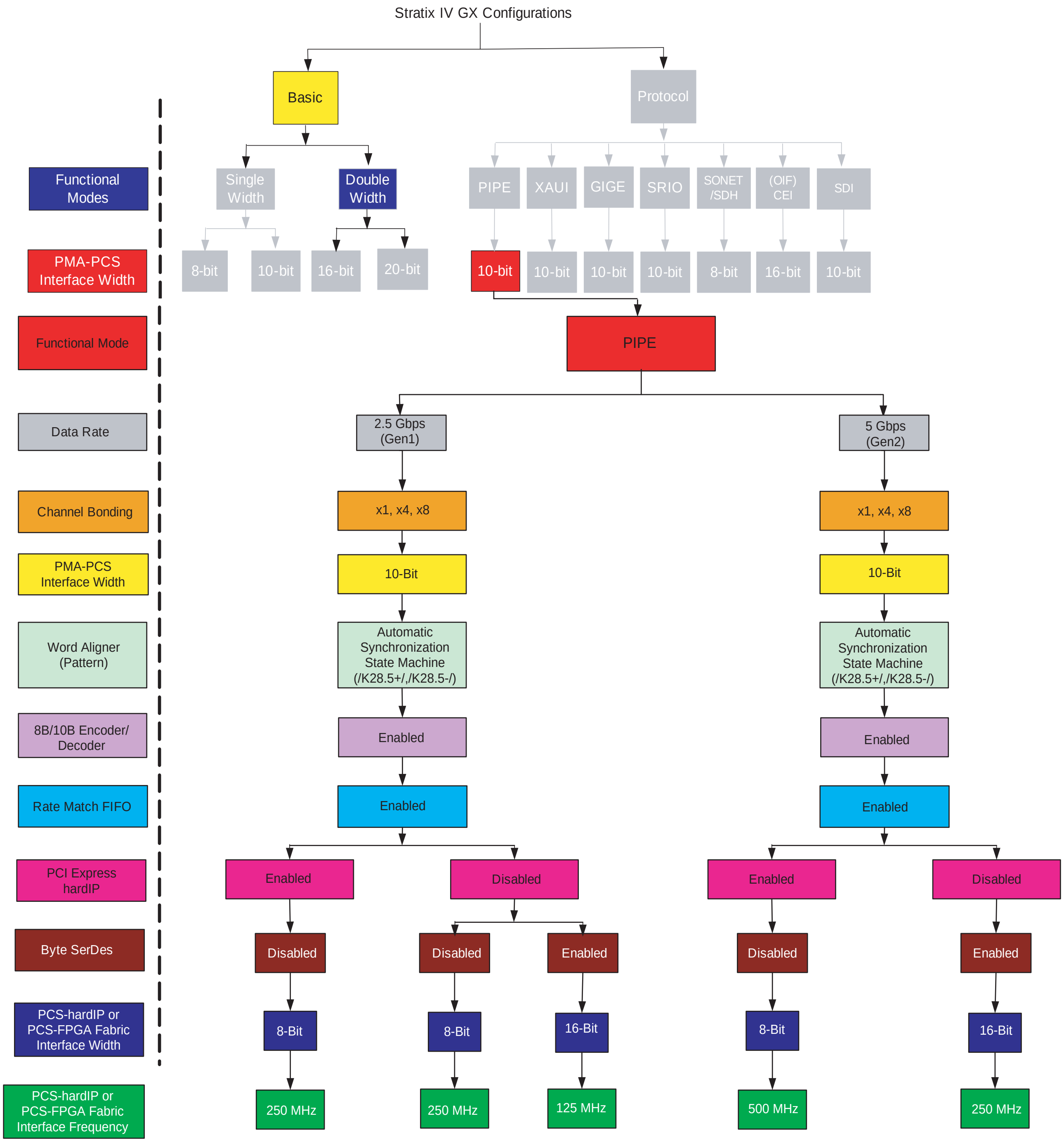
PCI Express (PIPE) Mode Configurations

Stratix IV GX transceivers support both Gen1 (2.5 Gbps) and Gen2 (5 Gbps) data rates in PCI Express (PIPE) functional mode. When configured for Gen2 (5 Gbps) data rate, the Stratix IV GX transceivers allow dynamic switching between Gen2 (5 Gbps) and Gen1 (2.5 Gbps) signaling rates. The dynamic switch capability between the two PCI Express (PIPE) signaling rates is very critical for speed negotiation during link training.

Stratix IV GX transceivers support ×1, ×4, and ×8 lane configurations in PIPE functional mode at both 2.5 Gbps and 5 Gbps data rates. In PCI Express (PIPE) ×1 configuration, the PCS and PMA blocks of each channel are clocked and reset independently. PCI Express (PIPE) ×4 and ×8 configurations support channel bonding for four-lane and eight-lane PCI Express (PIPE) links. In these bonded channel configurations, the PCS and PMA blocks of all bonded channels share common clock and reset signals.

Figure 1–95 shows the Stratix IV GX transceiver configurations allowed in PCI Express (PIPE) functional mode.

Figure 1–95. Stratix IV GX Transceivers in PCI Express (PIPE) Functional Mode



PCI Express (PIPE) Mode Datapath

Figure 1–96 shows the Stratix IV GX transceiver datapath when configured in PCI Express (PIPE) functional mode.

Figure 1–96. Stratix IV GX Transceiver Datapath in PCI Express (PIPE) x1 Mode

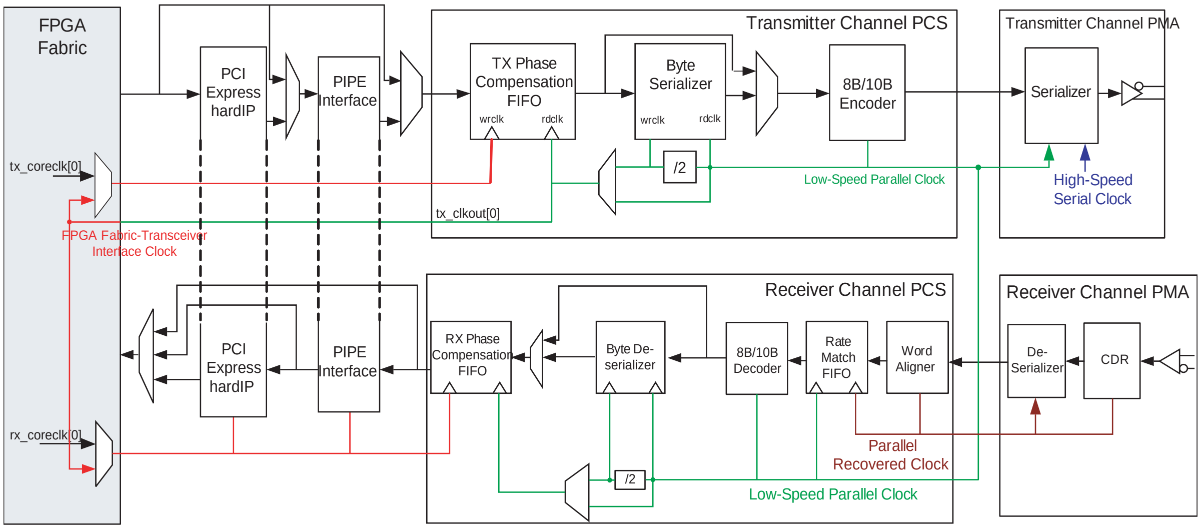


Table 1–31 shows the transceiver datapath clock frequencies in PCI Express (PIPE) functional mode configured using the ALTGX MegaWizard Plug-In Manager.

Table 1–31. Stratix IV GX Transceiver Datapath Clock Frequencies in PCI Express (PIPE) Mode

Functional Mode	Data Rate	High-Speed Serial Clock Frequency	Parallel Recovered Clock and Low-Speed Parallel Clock Frequency	FPGA Fabric-Transceiver Interface Clock Frequency	
				Without Byte Serializer/ Deserializer (8-Bit Wide)	With Byte Serializer/ Deserializer (16-Bit Wide)
PCI Express (PIPE) ×1, ×4, ×8 (Gen1)	2.5 Gbps	1.25 GHz	250 MHz	250 MHz	125 MHz
PCI Express (PIPE) ×1, ×4, ×8 (Gen2)	5 Gbps	2.5 GHz	500 MHz	N/A *(1)*	250 MHz

Note to Table 1–31:

(1) In PCI Express (PIPE) functional mode at Gen2 (5 Gbps) data rate, the byte serializer/deserializer cannot be bypassed.

The transceiver datapath clocking varies between non-bonded (×1) and bonded (×4 and ×8) configurations in PCI Express (PIPE) mode.

For more information about transceiver datapath clocking in different PCI Express (PIPE) configurations, refer to the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*.

The transmitter datapath in PCI Express (PIPE) mode consists of:

- PIPE interface
- Transmitter phase compensation FIFO

- Optional byte serializer (enabled for 16-bit and disabled for 8-bit FPGA fabric-transceiver interface)
- 8B/10B encoder
- 10:1 serializer
- Transmitter buffer with receiver detect circuitry

The receiver datapath in PCI Express (PIPE) mode consists of:

- Receiver buffer with signal detect circuitry
- 1:10 deserializer
- Word aligner that implements PCI Express-compliant synchronization state machine
- Optional rate match FIFO (clock rate compensation) that can tolerate up to 600 PPM frequency difference
- 8B/10B decoder
- Optional byte deserializer (enabled for 16-bit and disabled for 8-bit FPGA fabric-transceiver interface)
- Receiver phase compensation FIFO
- PIPE interface

Table 1–32 shows features supported in PCI Express (PIPE) functional mode for 2.5 Gbps and 5 Gbps data rate configurations.

Table 1–32. Supported Features in PCI Express (PIPE) Mode

Feature	2.5 Gbps (Gen1)	5 Gbps (Gen2)
×1, ×4, ×8 link configurations	✓	✓
PCI Express-compliant synchronization state machine	✓	✓
±300 PPM (total 600 PPM) clock rate compensation	✓	✓
8-bit FPGA fabric-Transceiver interface	✓	—
16-bit FPGA fabric-Transceiver interface	✓	✓
Transmitter buffer electrical idle	✓	✓
Receiver Detection	✓	✓
8B/10B encoder disparity control when transmitting compliance pattern	✓	✓
Power state management	✓	✓
Receiver status encoding	✓	✓
Dynamic switch between 2.5 Gbps and 5 Gbps signaling rate	—	✓
Dynamically selectable transmitter margining for differential output voltage control	—	✓
Dynamically selectable transmitter buffer de-emphasis of -3.5 db and -6 dB	—	✓
Dynamically selectable full-swing and half-swing transmitter output voltage levels	—	✓

PCI Express (PIPE) Interface

In PCI Express (PIPE) mode, each channel has a PIPE interface block that transfers data, control, and status signals between the PHY-MAC layer and the transceiver channel PCS and PMA blocks. The PIPE interface block is compliant to version 2.00 of the PCI Express (PIPE) specification. If you use the PCI Express hard IP block, the PHY-MAC layer is implemented in the hard IP block. Otherwise, the PHY-MAC layer may be implemented using Soft IP in the FPGA fabric.

☞ The PIPE interface block is only used in PCI Express (PIPE) mode and cannot be bypassed.

Besides transferring data, control, and status signals between the PHY-MAC layer and the transceiver, the PIPE interface block implements the following functions required in a PIPE-compliant physical layer device:

- Force the transmitter buffer in electrical idle state
- Initiate receiver detect sequence
- 8B/10B encoder disparity control when transmitting compliance pattern
- Manage PIPE power states
- Indicate completion of various PHY functions like receiver detection and power state transitions on the `pipephydonestatus` signal
- Encode receiver status and error conditions on the `pipestatus[2:0]` signal as specified in the PIPE specification

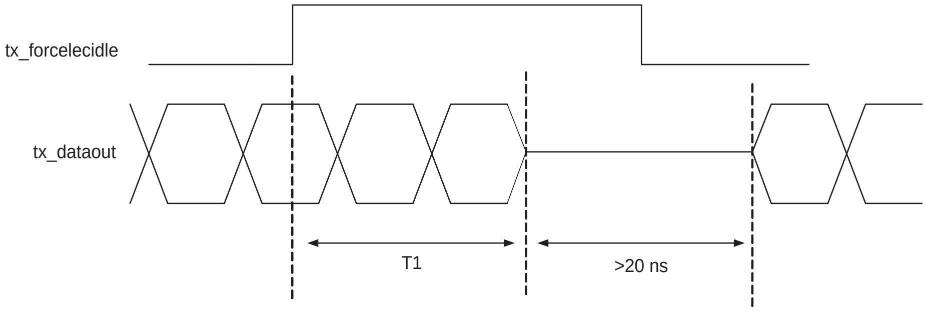
Transmitter Buffer Electrical Idle

When the input signal `tx_forceelecidle` is asserted high, the PIPE interface block puts the transmitter buffer in that channel in electrical idle state. During electrical idle, the transmitter buffer differential and common mode output voltage levels are compliant to the PCI Express Base Specification 2.0 for both PCI Express Gen1 and Gen2 data rates.

Figure 1–97 shows the relationship between assertion of the `tx_forceelecidle` signal and the transmitter buffer output on the `tx_dataout` port. Time T1 taken from the assertion of the `tx_forceelecidle` signal to the transmitter buffer reaching electrical idle voltage levels is pending characterization. Once in electrical idle state, the PCI Express (PIPE) protocol requires the transmitter buffer to stay in electrical idle for a minimum of 20 ns for both Gen1 and Gen2 data rates.

☞ The minimum period of time for which the `tx_forceelecidle` signal must be asserted high such that the transmitter buffer stays in electrical idle state for at least 20 ns is pending characterization.

Figure 1–97. Transmitter Buffer Electrical Idle State



The PCI Express (PIPE) specification requires the transmitter buffer to be in electrical idle in certain power states. Refer to Table 1–31 for more information on the `tx_forceelecidle` signal levels required in different PCI Express (PIPE) power states.

Receiver Detection

During the detect substate of the LTSSM state machine, the PCI Express (PIPE) protocol requires the transmitter channel to perform a receiver detect sequence to detect if a receiver is present at the far end of each lane. The PCI Express (PIPE) specification specifies receiver detect operation to be performed during the P1 power state.

The PIPE interface block in Stratix IV GX transceivers provides an input signal `tx_detectrxloopback` for the receiver detect operation. When the input signal `tx_detectrxloopback` is asserted high in the P1 power state, the PIPE interface block sends a command signal to the transmitter buffer in that channel to initiate a receiver detect sequence. In the P1 power state, the transmitter buffer must always be in electrical idle state. On receiving this command signal, the receiver detect circuitry creates a step voltage at the output of the transmitter buffer. If an active receiver (that complies to the PCI Express [PIPE] input impedance requirements) is present at the far end, the time constant of the step voltage on the trace is higher compared to when the receiver is not present. The receiver detect circuitry monitors the time constant of the step signal seen on the trace to decide if a receiver was detected or not. The receiver detect circuitry monitor needs a 125-MHz clock for operation that you must drive on the `fixedclk` port.

☞ For the receiver detect circuitry to function reliably, the AC-coupling capacitor on the serial link and the receiver termination values used in your system must be compliant to the PCI Express (PIPE) Base Specification 2.0.

Receiver detect circuitry communicates the status of the receiver detect operation to the PIPE interface block. If a far-end receiver is successfully detected, the PIPE interface block asserts `pipephydonestatus` for one clock cycle and synchronously drives the `pipestatus[2:0]` signal to 3'b011. If a far-end receiver is not detected, the PIPE interface block asserts `pipephydonestatus` for one clock cycle and synchronously drives the `pipestatus[2:0]` signal to 3'b000.

Figure 1–98 and 1–121 show receiver detect operation where a receiver was successfully detected and where a receiver was not detected, respectively.

Figure 1–98. Receiver Detect, Successfully Detected

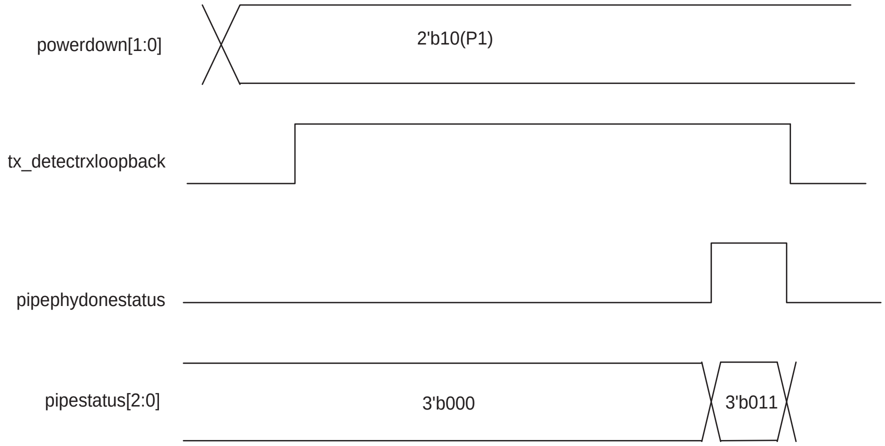
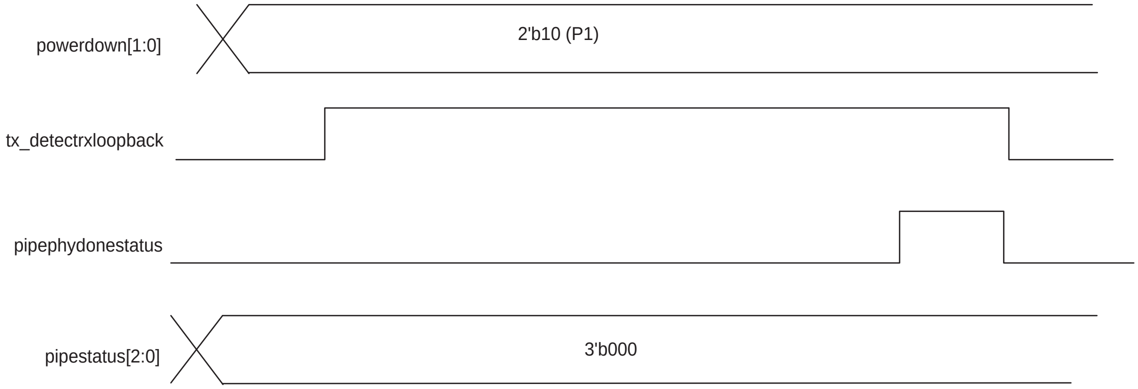


Figure 1–99. Receiver Detect, Unsuccessfully Detected



Compliance Pattern Transmission Support

The LTSSM state machine can enter the polling.compliance substate where the transmitter is required to transmit a compliance pattern as specified in the PCI Express (PIPE) Base Specification 2.0. The polling.compliance substate is intended to assess if the transmitter is electrically compliant with the PCI Express (PIPE) voltage and timing specifications.

The compliance pattern is a repeating sequence of the following four code groups:

- /K28.5/
- /D21.5/
- /K28.5/
- /D10.2/

PCI Express (PIPE) protocol requires the first /K28.5/ code group of the compliance pattern to be encoded with negative current disparity. To satisfy this requirement, the PIPE interface block provides an input signal, `tx_forcedispcompliance`. A high level on `tx_forcedispcompliance` forces the associated parallel transmitter data on the `tx_datain` port to transmit with negative current running disparity.

- For 8-bit transceiver channel width configurations, you must drive `tx_forcedispcompliance` high in the same parallel clock cycle as the first /K28.5/ of the compliance pattern on the `tx_datain` port.
- For 16-bit transceiver channel width configurations, you must drive only the LSB of `tx_forcedispcompliance[1:0]` high in the same parallel clock cycle as /K28.5/D21.5/ of the compliance pattern on the `tx_datain` port.

Figure 1–100 and 1–122 show the required level on the `tx_forcedispcompliance` signal while transmitting the compliance pattern in 8-bit and 16-bit channel width configurations, respectively.

Figure 1–100. Compliance Pattern Transmission Support, 8-Bit Wide Channel Configurations

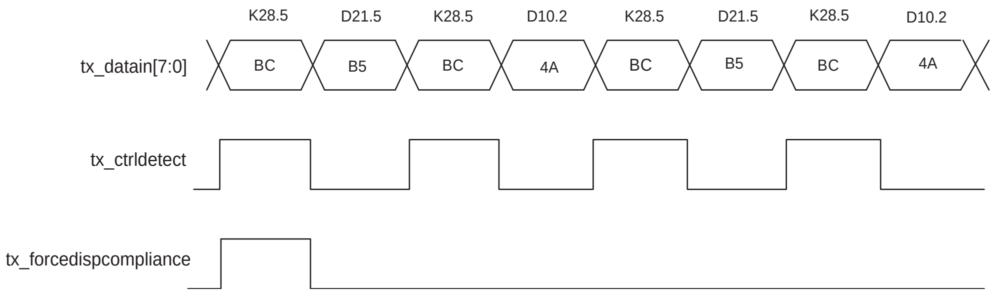
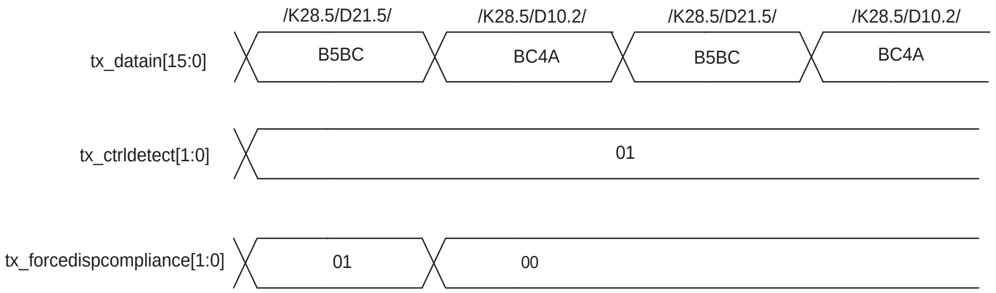


Figure 1–101. Compliance Pattern Transmission Support, 16-Bit Wide Channel Configurations



Power State Management

The PCI Express (PIPE) specification defines four power states, namely P0, P0s, P1, and P2, that the physical layer device must support to minimize power consumption.

- P0 is the normal operation state during which packet data is transferred on the PCI Express (PIPE) link.
- P0s, P1, and P2 are low-power states into which the physical layer must transition as directed by the PHY-MAC layer to minimize power consumption.

The PCI Express (PIPE) specification provides the mapping of these power states to the LTSSM states specified in the PCI Express (PIPE) Base Specification 2.0. The PHY-MAC layer is responsible for implementing the mapping logic between the LTSSM states and the four power states in the PCI Express (PIPE)-compliant PHY.

The PIPE interface in Stratix IV GX transceivers provides an input port, `powerdn[1:0]`, for each transceiver channel configured in PCI Express (PIPE) mode. Table 1–35 shows mapping between the logic levels driven on the `powerdn[1:0]` port and the resulting power state that the PIPE interface block puts the transceiver channel into.

Table 1–33. Power State Functions and Descriptions

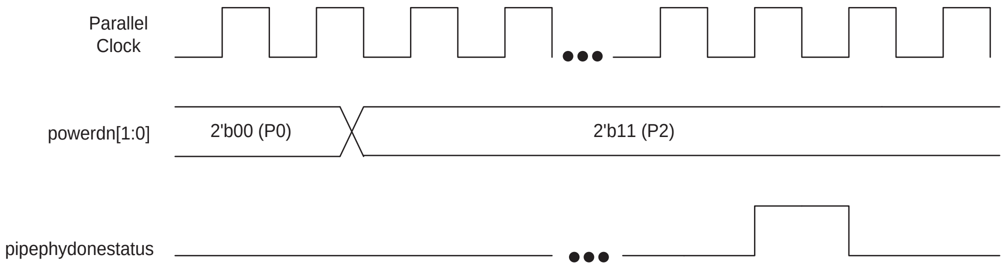
Power State	powerdn	Function	Description
P0	2'b00	Transmits normal data, transmits Electrical Idle, or enters into loopback mode	Normal operation mode
P0s	2'b01	Only transmits Electrical Idle	Low recovery time saving state
P1	2'b10	Transmitter buffer is powered down and can do a receiver detect while in this state	High recovery time power saving state
P2	2'b11	Transmits Electrical Idle or a beacon to wake up the downstream receiver	Lowest power saving state

☞ When transitioning from the P0 power state to lower power states (P0s, P1, and P2), the PCI Express (PIPE) specification requires the physical layer device to implement power saving measures. Stratix IV GX transceivers do not implement these power saving measures except putting the transmitter buffer in Electrical Idle in the lower power states.

The PIPE interface block indicates successful power state transition by asserting the `pipephydonestatus` signal for one parallel clock cycle as specified in the PCI Express (PIPE) specification. The PHY-MAC layer must not request any further power state transition until the `pipephydonestatus` signal has indicated the completion of the current power state transition request.

Figure 1–102 shows an example waveform for a transition from P0 to P2 power state.

Figure 1–102. Power State Transition from P0 to P2



The PCI Express (PIPE) specification allows the PIPE interface to perform protocol functions like receiver detect, loopback, and beacon transmission in specified power states only. This requires the PHY-MAC layer to drive the `tx_detectrxloopback` and `tx_forceelecidle` signals appropriately in each power state to perform these functions. Table 1–34 summarizes the logic levels that the PHY-MAC layer must drive on the `tx_detectrxloopback` and `tx_forceelecidle` signals in each power state.

Table 1–34. Logic Levels for the PHY-MAC Layer

Power State	tx_detectrxloopback	tx_forceelecidle
P0	0: normal mode 1: data path in loopback mode	0: Must be deasserted 1: Illegal mode
P0s	Don't care	0: Illegal mode 1: Must be asserted in this state
P1	0: Electrical Idle 1: receiver detect	0: Illegal mode 1: Must be asserted in this state
P2	Don't care	Deasserted in this state for sending beacon. Otherwise asserted.

Receiver Status

The PCI Express (PIPE) specification requires the PHY to encode the receiver status on a 3-bit `RxStatus[2:0]` signal. This status signal is used by the PHY-MAC layer for its operation.

The PIPE interface block receives status signals from the transceiver channel PCS and PMA blocks and encodes the status on the 3-bit output signal `pipestatus[2:0]` to the FPGA fabric. The encoding of the status signals on `pipestatus[2:0]` is compliant with the PCI Express (PIPE) specification and listed in Table 1–35.

Table 1–35. Encoding of the Status Signals on pipestatus[2:0]

pipestatus[2:0]	Description	Error Condition Priority
3'b000	Received data OK	N/A
3'b001	One SKP symbol added	5
3'b010	One SKP symbol deleted	6
3'b011	Receiver detected	N/A
3'b100	8B/10B decode error	1
3'b101	Elastic buffer (rate match FIFO) overflow	2
3'b110	Elastic buffer (rate match FIFO) underflow	3
3'b111	Received disparity error	4

Two or more of the error conditions, for example, 8B/10B decode error (code group violation), rate match FIFO overflow or underflow, and receiver disparity error, can occur simultaneously. The PIPE interface follows the priority listed in Table 1–35 while encoding the receiver status on the `pipestatus[2:0]` port. For example, if the PIPE interface receives an 8B/10B decode error and disparity error for the same symbol, it drives 3'b100 on the `pipestatus[2:0]` signal.

Fast Recovery Mode

The PCI Express Base specification fast training sequences (FTS) are used for bit and byte synchronization to transition from L0s to L0 (PIPE P0s to P0) power states. When transitioning from L0s to L0 power state, the PCI Express Base Specification requires the physical layer device to acquire bit and byte synchronization after receiving a maximum of 255 FTS (~4 us at Gen1 data rate and ~2 us at Gen2 data rate).

If the Stratix IV GX receiver CDR is configured in Automatic Lock mode, the receiver cannot meet the PCI Express specification of acquiring bit and byte synchronization within 4 us (Gen1 data rate) or 2 us (Gen2 data rate) due to the signal detect and PPM detector time. To meet this specification, each Stratix IV GX transceiver has a built-in Fast Recovery circuitry that you can optionally enable.

☞ To enable the Fast Recovery circuitry, select the **Enable fast recovery mode** option in the ALTGX MegaWizard Plug-In Manager.

If you enable the **Fast Recovery mode** option, the Fast Recovery circuitry controls the receiver CDR `rx_locktorefclk` and `rx_locktodata` signals to force the receiver CDR in LTR or LTD modes. It relies on the Electrical Idle Ordered Sets (EIOS), NFTS sequences received in L0 power state, and the signal detect signal from the receiver input buffer to control the receiver CDR lock mode.

☞ The Fast Recovery circuitry is self-operational and does not require any control inputs from the user. When enabled, the `rx_locktorefclk` and `rx_locktodata` ports are not available in the ALTGX MegaWizard Plug-In Manager.

Electrical Idle Inference

The PCI Express (PIPE) protocol allows inferring the electrical idle condition at the receiver instead of detecting the electrical idle condition using analog circuitry. Clause 4.2.4.3 in PCI Express (PIPE) Base Specification 2.0 specifies conditions to infer electrical idle at the receiver in various sub-states of the LTSSM state machine.

In all PCI Express (PIPE) modes (×1, ×4, and ×8), each receiver channel PCS has an optional Electrical Idle Inference module designed to implement the electrical idle inference conditions specified in the PCI Express (PIPE) Base Specification 2.0. You can enable the Electrical Idle Inference module by selecting the **Enable electrical idle inference functionality** option in the ALTGX MegaWizard Plug-In manager.

If enabled, this module infers electrical idle depending on the logic level driven on the `rx_elecidleinfersel[2:0]` input signal. The Electrical Idle Inference module in each receiver channel indicates whether the electrical idle condition is inferred or not on the `pipeelecidle` signal of that channel. The Electrical Idle Interface module drives the `pipeelecidle` signal high if it infers an electrical idle condition; otherwise, it drives it low.

Table 1–36 shows electrical idle inference conditions specified in the PCI Express (PIPE) Base Specification 2.0 and implemented in the Electrical Idle Inference module to infer electrical idle in various substates of the LTSSM state machine. For the Electrical Idle Inference Module to correctly infer an electrical idle condition in each LTSSM substate, you must drive the `rx_elecidleinfersel[2:0]` signal appropriately, as shown in Table 1–36.

Table 1–36. Electrical Idle Inference Conditions (Part 1 of 2)

LTSSM State	Gen1 (2.5 Gbps)	Gen2 (5 Gbps)	rx_elecidleinfersel[2:0]
L0	Absence of skip ordered set in 128 μs window	Absence of skip ordered set in 128 μs window	3'b100
Recovery.RcvrCfg	Absence of TS1 or TS2 ordered set in 1280 UI interval	Absence of TS1 or TS2 ordered set in 1280 UI interval	3'b101

Table 1–36. Electrical Idle Inference Conditions (Part 2 of 2)

LTSSM State	Gen1 (2.5 Gbps)	Gen2 (5 Gbps)	rx_elecidleinfersel[2:0]
Recovery.Speed when successful speed negotiation = 1'b1	Absence of TS1 or TS2 ordered set in 1280 UI interval	Absence of TS1 or TS2 ordered set in 1280 UI window	3'b101
Recovery.Speed when successful speed negotiation = 1'b0	Absence of an exit from Electrical Idle in 2000 UI interval	Absence of an exit from Electrical Idle in 16000 UI interval	3'b110
Loopback.Active (as slave)	Absence of an exit from Electrical Idle in 128 µs window	N/A	3'b111

In the Recovery.Speed substate of the LTSSM state machine with unsuccessful speed negotiation (`rx_elecidleinfersel[2:0]` = *3'b110*), the PCI Express (PIPE) Base Specification requires the receiver to infer an electrical idle condition (`pipeelecidle` = high) if absence of an exit from Electrical Idle is detected in a 2000 UI interval for Gen1 data rate and 16000 UI interval for Gen2 data rate. The electrical idle inference module detects an absence of exit from Electrical Idle if four /K28.5/ COM code groups are not received in the specified interval.

In other words, when configured for Gen1 data rate and `rx_elecidleinfersel[2:0]` = 3'b110, the Electrical Idle inference module asserts `pipeelecidle` high if it does not receive four /K28.5/ COM code groups in a 2000 UI interval. When configured for Gen1 data rate and `rx_elecidleinfersel[2:0]` = 3'b111 in Loopback.Active substate of the LTSSM state machine, the Electrical Idle inference module asserts `pipeelecidle` high if it does not receive four /K28.5/ COM code groups in a 128 µs interval.

When configured for Gen2 data rate and `rx_elecidleinfersel[2:0]` = 3'b110, the Electrical Idle inference module asserts `pipeelecidle` high if it does not receive four /K28.5/ COM code groups in a 16000 UI interval.

☞ The Electrical Idle inference module does not have the capability to detect electrical idle exit condition based on reception of the electrical idle exit ordered set (EIEOS), as specified in the PCI Express (PIPE) Base Specification.

If you select the **Enable Electrical Idle Inference Functionality** option in the ALTGX MegaWizard Plug-In Manager and drive `rx_elecidleinfersel[2:0] = 3'b0xx`, the Electrical Idle Inference Block uses the EIOS detection from the Fast Recovery circuitry to drive the `pipeelecidle` signal

If you do not select the **Enable electrical idle inference functionality** option in the ALTGX MegaWizard Plug-In manager, the Electrical Idle inference module is disabled. In this case, the `rx_signaldetect` signal from the signal detect circuitry in the receiver buffer is inverted and driven as the `pipeelecidle` signal.

PCI Express Gen2 (5 Gbps) Support

The PCI Express (PIPE) functional mode supports the following additional features when configured for 5 Gbps data rate:

- Dynamic switch between 2.5 Gbps and 5 Gbps signaling rate

- Dynamically selectable transmitter margining for differential output voltage control
- Dynamically selectable transmitter buffer de-emphasis of -3.5 db and -6 dB
- Dynamically selectable full-swing and half-swing transmitter output voltage levels

Dynamic Switch Between Gen1 (2.5 Gbps) and Gen2 (5 Gbps) Signaling Rate

During link training, the upstream and downstream PCI Express (PIPE) ports negotiate the speed (2.5 Gbps or 5 Gbps) at which the link operates. Because the upstream and downstream PCI Express (PIPE) ports do not know the speed capabilities of their link partner, the PCI Express (PIPE) protocol requires each port to start with a Gen1 (2.5 Gbps) signaling rate. One of the ports capable of supporting Gen2 (5 Gbps) signaling rate might initiate a speed change request by entering the Recovery state of the LTSSM. In the Recovery state, each port advertises its speed capabilities by transmitting training sequences as specified in the PCI Express Base (PIPE) Specification 2.0. If both ports are capable of operating at the Gen2 (5 Gbps) signaling rate, the PHY-MAC layer instructs the physical layer device to operate at Gen2 (5 Gbps) signaling rate.

To support speed negotiation during link training, the PCI Express (PIPE) specification requires a PCI Express (PIPE)-compliant physical layer device to provide an input signal (`Rate`*)* to the PHY-MAC layer. When this input signal is driven low, the physical layer device must operate at Gen1 (2.5 Gbps) signaling rate; when driven high, it must operate at Gen2 (5 Gbps) signaling rate. The PCI Express (PIPE) specification allows the PHY-MAC layer to initiate a signaling rate switch only in power states P0 and P1 with the transmitter buffer in Electrical Idle state. The PCI Express (PIPE) specification allows the physical layer device to implement the signaling rate switch using any of the following approaches:

- Change the transceiver datapath clock frequency keeping the transceiver interface width constant
- Change the transceiver interface width between 8-bit and 16-bit keeping the transceiver clock frequency constant

When configured in PCI Express (PIPE) functional mode at Gen2 (5 Gbps) data rate, the ALTGX MegaWizard Plug-In Manager provides an input signal, `rateswitch`. The `rateswitch` signal is functionally equivalent to the `Rate` signal specified in the PCI Express (PIPE) specification. The PHY-MAC layer can use the `rateswitch` signal to instruct the Stratix IV GX device to operate at either Gen1 (2.5 Gbps) or Gen2 (5 Gbps) data rate, depending on the negotiated speed between the upstream and downstream ports. A low-to-high transition on the `rateswitch` signal initiates a data rate switch from Gen1 (2.5 Gbps) to Gen2 (5 Gbps). A high-to-low transition on the `rateswitch` signal initiates a data rate switch from Gen2 (5 Gbps) to Gen1 (2.5 Gbps). The signaling rate switch between Gen1 (2.5 Gbps) and Gen2 (5 Gbps) is achieved by changing the transceiver datapath clock frequency between 250 MHz and 500 MHz, while maintaining a constant transceiver interface width of 16-bit.

The dedicated PCI Express (PIPE) rate switch circuitry performs the dynamic switch between Gen1 (2.5 Gbps) and Gen2 (5 Gbps) signaling rate. The PCI Express (PIPE) rate switch circuitry consists of:

- PCI Express (PIPE) rate switch controller

- PCI Express (PIPE) clock switch circuitry

PCI Express (PIPE) Rate Switch Controller

The `rateswitch` signal serves as the input signal to the PCI Express (PIPE) rate switch controller. On seeing a transition on the `rateswitch` signal from the PHY-MAC layer, the PCI Express (PIPE) rate switch controller performs the following operations:

- Controls the PCI Express (PIPE) clock switch circuitry to switch between Gen1 (2.5 Gbps) and Gen2 (5 Gbps) signaling rate depending on the level on the `rateswitch` signal
- Disables and resets the transmitter and receiver phase compensation FIFO pointers until the PCI Express (PIPE) clock switchover circuitry indicates successful rate switch completion
- Communicates completion of rate switch to the PCI Express (PIPE) interface module, which in turn communicates completion of the rate switch to the PHY-MAC layer on the `pipephydonestatus` signal

PCI Express (PIPE) rate switch controller location:

- In PCI Express (PIPE) ×1 mode, the PCI Express (PIPE) rate switch controller is located in the transceiver PCS of each channel.
- In PCI Express (PIPE) ×4 mode, the PCI Express (PIPE) rate switch controller is located in `CMU0 Channel` within the transceiver block.
- In PCI Express (PIPE) ×8 mode, the PCI Express (PIPE) rate switch controller is located in `CMU0_Channel` within the master transceiver block.

☞ When operating at Gen 2 data rate, asserting the `rx_digitalreset` signal causes the PCI Express rate switch circuitry to switch the transceiver to Gen 1 data rate.

PCI Express (PIPE) Clock Switch Circuitry

When the PHY-MAC layer instructs a rate switch between Gen1 (2.5 Gbps) and Gen2 (5 Gbps) signaling rates, both the transmitter high-speed serial and low-speed parallel clock and the CDR recovered clock must switch to support the instructed data rate. Stratix IV GX transceivers have dedicated PCI Express (PIPE) clock switch circuitry located in the following blocks:

- Local clock divider in transmitter PMA of each transceiver channel
- CMU0 clock divider in `CMU0_Channel` of each transceiver block
- Receiver CDR in receiver PMA of each transceiver channel

PCI Express (PIPE) transmitter high-speed serial and low-speed parallel clock switch occurs:

- In PCI Express (PIPE) ×1 mode, the `CMU_PLL` clock switch occurs in the local clock divider in each transceiver channel.
- In PCI Express (PIPE) ×4 mode, the `CMU_PLL` clock switch occurs in the CMU0 clock divider in the `CMU0_Channel` within the transceiver block.
- In PCI Express (PIPE) ×8 mode, the `CMU_PLL` clock switch occurs in the CMU0 clock divider in the `CMU0_Channel` within the master transceiver block.

In PCI Express (PIPE) ×1, ×4, and ×8 modes, the recovered clock switch happens in the receiver CDR of each transceiver channel.

Table 1–37 lists the locations of the PCI Express (PIPE) rate switch controller and the PCI Express (PIPE) clock switch circuitry in PCI Express (PIPE) ×1, ×2, ×4, and ×8 modes.

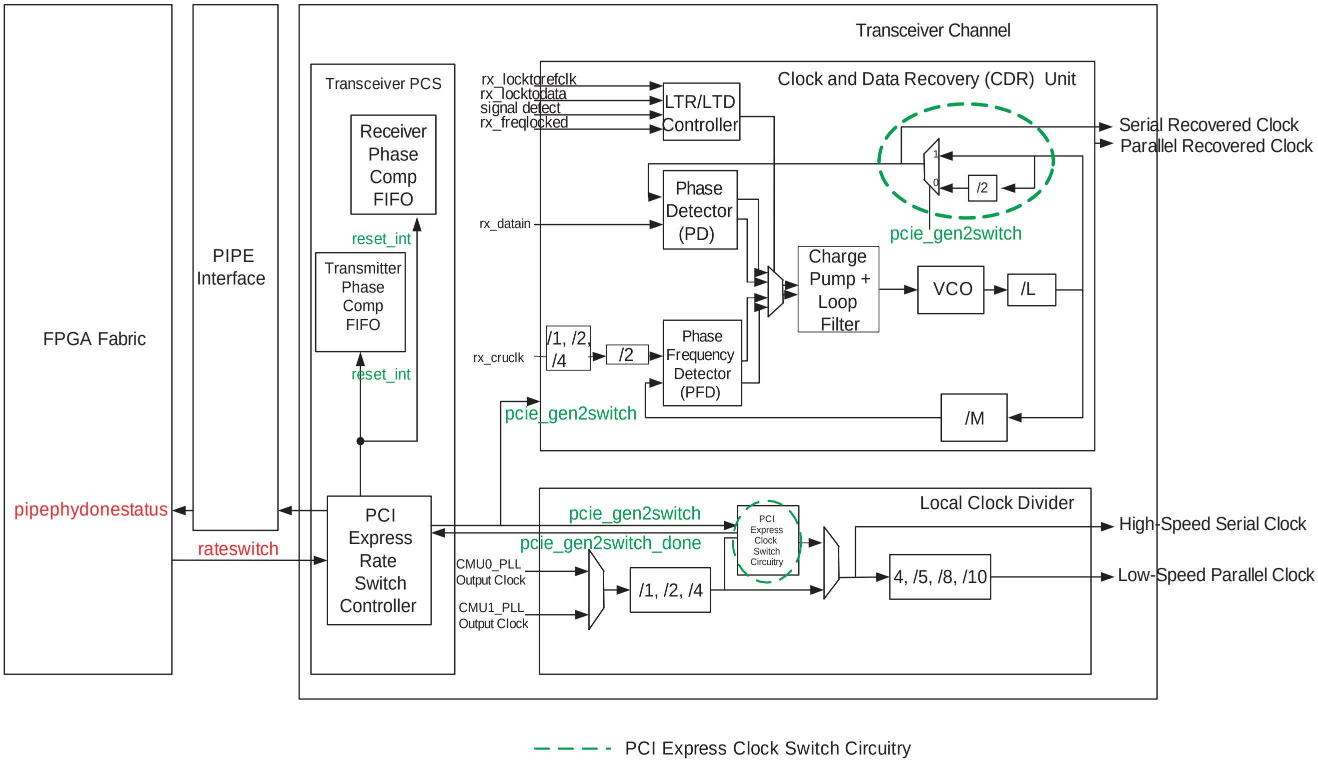
Table 1–37. PCI Express (PIPE) Rate Switch Controller and Clock Switch Circuitry

Channel Bonding Option	Location of PCI Express (PIPE) Rate Switch Controller Module	Location of PCI Express (PIPE) Clock Switch Circuitry	
		Transmitter High-Speed Serial and Low-Speed Parallel Clock Switch Circuitry	**Recovered Clock Switch Circuitry**
×1	Individual channel PCS block	Local clock divider in transmitter PMA of each channel	CDR block in receiver PMA of each channel
×4	`CMU0 Channel`	CMU0 clock divider in `CMU0_Channel`	CDR block in receiver PMA of each channel
×8	`CMU0 Channel` of the master transceiver block	CMU0 clock divider in `CMU0_Channel` of the master transceiver block	CDR block in receiver PMA of each channel

Dynamic Switch between Gen1 (2.5 Gbps) and Gen2 (5 Gbps) Signaling Rates in PCI Express (PIPE) x1 Mode

Figure 1–103 shows the PCI Express (PIPE) rate switch circuitry in PCI Express (PIPE) ×1 mode configured at Gen2 (5 Gbps) data rate.

Figure 1–103. Dynamic Switch Signaling in PCI Express (PIPE) x1 Mode



In PCI Express (PIPE) ×1 mode configured at Gen2 (5 Gbps) data rate, when the PCI Express (PIPE) rate switch controller sees a transition on the `rateswitch` signal, it sends control signal `pcie_gen2switch` to the PCI Express (PIPE) clock switch circuitry in the local clock divider block and the receiver CDR to switch to the instructed signaling rate. A low-to-high transition on the `rateswitch` signal initiates a Gen1 (2.5 Gbps) to Gen2 (5 Gbps) signaling rate switch. A high-to-low transition on the `rateswitch` signal initiates a Gen2 (5 Gbps) to Gen1 (2.5 Gbps) signaling rate switch.

Table 1–38 shows transceiver clock frequencies when switching between 2.5 Gbps and 5 Gbps signaling rates.

Table 1–38. Transceiver Clock Frequencies Signaling Rates in PCI Express (PIPE) x1 Mode

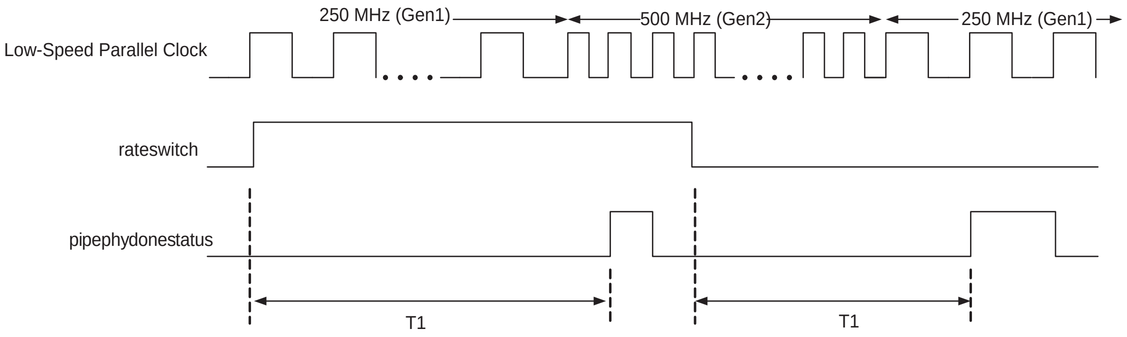
Transceiver Clocks	Gen1 (2.5 Gbps) to Gen2 (5 Gbps) Switch (Low-to-High Transition on the rateswitch Signal)	Gen2 (5 Gbps) to Gen1 (2.5 Gbps) Switch (High-to-Low Transition on the rateswitch Signal)
High-Speed Serial Clock	1.25 GHz to 2.5 GHz	2.5 GHz to 1.25 GHz
Low-Speed Parallel Clock	250 MHz to 500 MHz	500 MHz to 250 MHz
Serial Recovered Clock	1.25 GHz to 2.5 GHz	2.5 GHz to 1.25 GHz
Parallel Recovered Clock	250 MHz to 500 MHz	500 MHz to 250 MHz
FPGA Fabric-Transceiver Interface Clock	125 MHz to 250 MHz	250 MHz to 125 MHz

The PCI Express (PIPE) clock switch circuitry in the local clock divider block performs the clock switch between 250 MHz and 500 MHz on the low-speed parallel clock when switching between Gen1 (2.5 Gbps) and Gen2 (5 Gbps) signaling rates. It indicates successful completion of clock switch on the `pcie_gen2switchdone` signal to the PCI Express (PIPE) rate switch controller. The PCI Express (PIPE) rate switch controller forwards the clock switch completion status to the PIPE interface block. The PCI Express (PIPE) interface block communicates the clock switch completion status to the PHY-MAC layer by asserting the `pipephydonestatus` signal for one parallel clock cycle.

Figure 1–104 shows low-speed parallel clock switch between Gen1 (250 MHz) and Gen2 (500 MHz) in response to the change in logic level on the `rateswitch` signal. The rate switch completion is shown marked with a one clock cycle assertion of the `pipephydonestatus` signal.

☞ Time T1 from a transition on the `rateswitch` signal to the assertion of `pipephydonestatus` is pending characterization.

Figure 1–104. Low-Speed Parallel Clock Switching in PCI Express (PIPE) x1 Mode

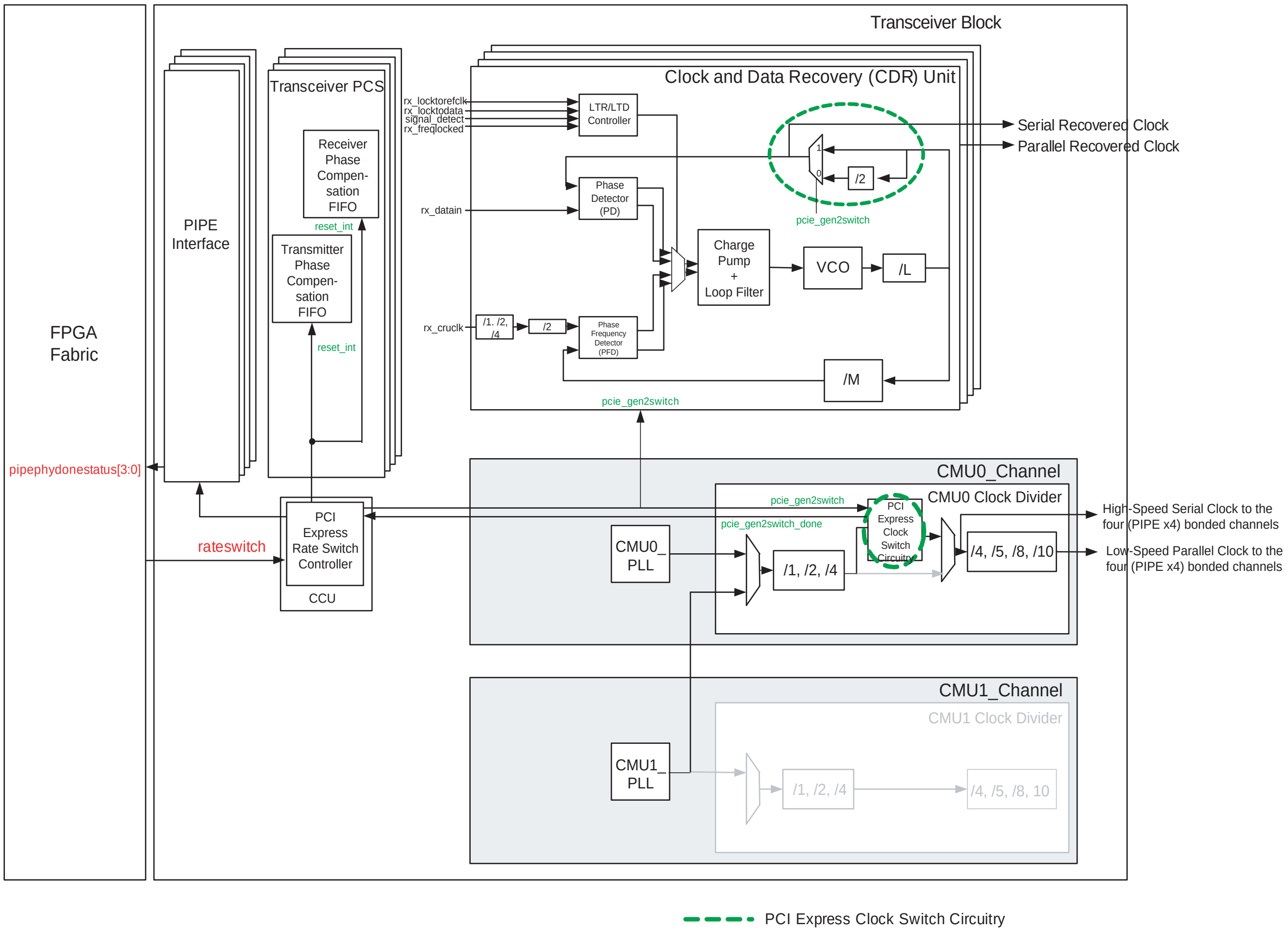


As a result of the signaling rate switch between Gen1 (2.5 Gbps) and Gen2 (5 Gbps), the FPGA fabric-transceiver interface clock switches between 125 MHz and 250 MHz. The FPGA fabric-transceiver interface clock clocks the read side and the write side of the transmitter phase compensation FIFO and the receiver phase compensation FIFO, respectively. It is also routed to the FPGA fabric on a global or regional clock resource and looped back to clock the write port and read port of the transmitter phase compensation FIFO and the receiver phase compensation FIFO, respectively. Due to the routing delay between the write and read clock of the transmitter and receiver phase compensation FIFOs, the write pointers and read pointers might collide during a rate switch between 125 MHz and 250 MHz. To avoid collision of the phase compensation FIFO pointers, the PCI Express (PIPE) rate switch controller automatically disables and resets the pointers during clock switch. When the PCI Express (PIPE) clock switch circuitry in the local clock divider indicates successful clock switch completion, the PCI Express (PIPE) rate switch controller releases the phase compensation FIFO pointer resets.

Dynamic Switch between Gen1 (2.5 Gbps) and Gen2 (5 Gbps) Signaling Rates in PCI Express (PIPE) ×4 Mode

Figure 1–105 shows the PCI Express rate switch circuitry in PCI Express (PIPE) ×4 mode configured at Gen2 (5 Gbps) data rate.

Figure 1–105. Dynamic Switch Signaling in PCI Express (PIPE) x4 Mode



In PCI Express (PIPE) ×4 mode configured at Gen2 (5 Gbps) data rate, when the PCI Express (PIPE) rate switch controller sees a transition on the `rateswitch` signal, it sends control signal `pcie_gen2switch` to the PCI Express (PIPE) clock switch circuitry in the CMU0 clock divider block and the receiver CDR to switch to the instructed signaling rate. A low-to-high transition on the `rateswitch` signal initiates a Gen1 (2.5 Gbps) to Gen2 (5 Gbps) signaling rate switch. A high-to-low transition on the `rateswitch` signal initiates a Gen2 (5 Gbps) to Gen1 (2.5 Gbps) signaling rate switch.

Table 1–39 shows the transceiver clock frequencies when switching between 2.5 Gbps and 5 Gbps signaling rates.

Table 1–39. Transceiver Clock Frequencies Signaling Rates in PCI Express (PIPE) x4 Mode (Part 1 of 2)

Transceiver Clocks	Gen1 (2.5 Gbps) to Gen2 (5 Gbps) Switch (Low-to-High Transition on the rateswitch Signal)	Gen2 (5 Gbps) to Gen1 (2.5 Gbps) Switch (High-to-Low Transition on the rateswitch Signal)
High-Speed Serial Clock	1.25 GHz to 2.5 GHz	2.5 GHz to 1.25 GHz
Low-Speed Parallel Clock	250 MHz to 500 MHz	500 MHz to 250 MHz
Serial Recovered Clock	1.25 GHz to 2.5 GHz	2.5 GHz to 1.25 GHz

Table 1–39. Transceiver Clock Frequencies Signaling Rates in PCI Express (PIPE) x4 Mode (Part 2 of 2)

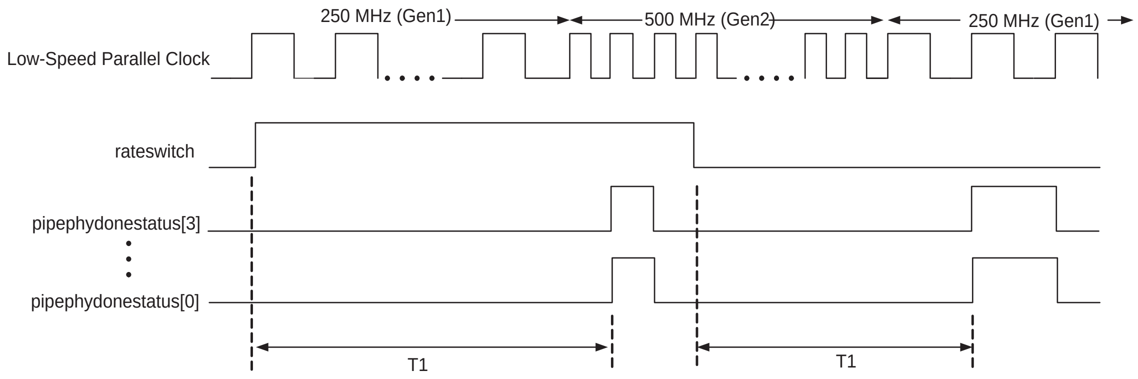
Transceiver Clocks	Gen1 (2.5 Gbps) to Gen2 (5 Gbps) Switch (Low-to-High Transition on the rateswitch Signal)	Gen2 (5 Gbps) to Gen1 (2.5 Gbps) Switch (High-to-Low Transition on the rateswitch Signal)
Parallel Recovered Clock	250 MHz to 500 MHz	500 MHz to 250 MHz
FPGA Fabric-Transceiver Interface Clock	125 MHz to 250 MHz	250 MHz to 125 MHz

The PCI Express (PIPE) clock switch circuitry in the CMU0 clock divider block performs the clock switch between 250 MHz and 500 MHz on the low-speed parallel clock when switching between Gen1 (2.5 Gbps) and Gen2 (5 Gbps) signaling rates. It indicates successful completion of clock switch on the `pcie_gen2switchdone` signal to the PCI Express (PIPE) rate switch controller. The PCI Express (PIPE) rate switch controller forwards the clock switch completion status to the PIPE interface block. The PIPE interface block communicates the clock switch completion status to the PHY-MAC layer by asserting the `pipephydonestatus` signal of all bonded channels for one parallel clock cycle.

Figure 1–106 shows low-speed parallel clock switch between Gen1 (250 MHz) and Gen2 (500 MHz) in response to the change in logic level on the `rateswitch` signal. The rate switch completion is shown marked with a one clock cycle assertion of the `pipephydonestatus` signal of all bonded channels.

Time T1 from a transition on the `rateswitch` signal to the assertion of `pipephydonestatus` is pending characterization.

Figure 1–106. Low-Speed Parallel Clock Switching in PCI Express (PIPE) x4 Mode

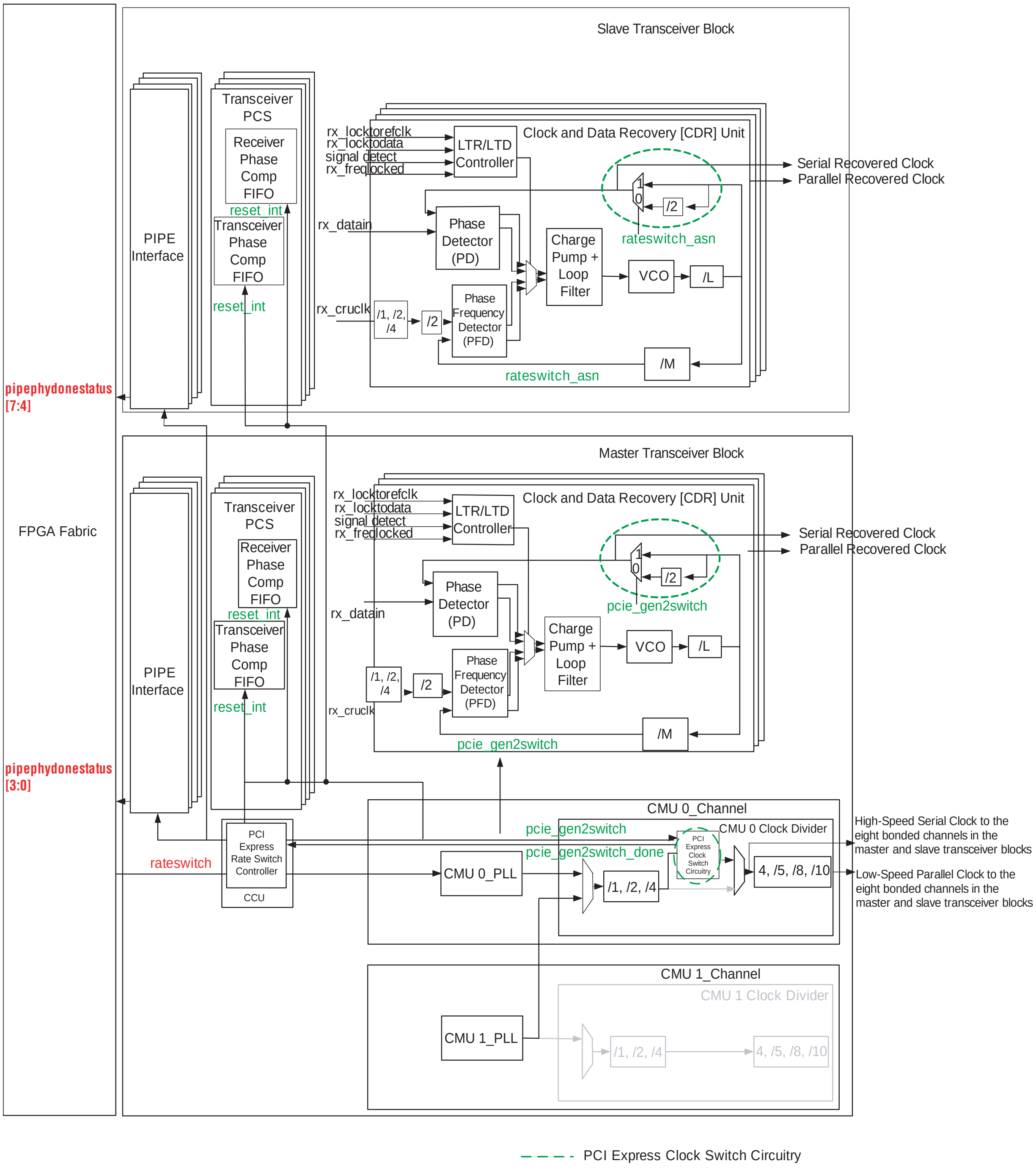


As a result of the signaling rate switch between Gen1 (2.5 Gbps) and Gen2 (5 Gbps), the FPGA fabric-transceiver interface clock switches between 125 MHz and 250 MHz. The FPGA fabric-transceiver interface clock clocks the read side and write side of the transmitter phase compensation FIFO and the receiver phase compensation FIFO of all bonded channels, respectively. It is also routed to the FPGA fabric on a global or regional clock resource and looped back to clock the write port and read port of the transmitter phase compensation FIFO and the receiver phase compensation FIFO, respectively. Due to the routing delay between the write and read clock of the transmitter and receiver phase compensation FIFOs, the write pointers and read pointers might collide during a rate switch between 125 MHz and 250 MHz. To avoid collision of the phase compensation FIFO pointers, the PCI Express (PIPE) rate switch controller automatically disables and resets the phase compensation FIFO pointers of all bonded channels during clock switch. When the PCI Express (PIPE) clock switch circuitry in the local clock divider indicates successful clock switch completion, the PCI Express (PIPE) rate switch controller releases the phase compensation FIFO pointer resets.

Dynamic Switch between Gen1 (2.5 Gbps) and Gen2 (5 Gbps) Signaling Rates in PCI Express (PIPE) ×8 Mode

Figure 1–107 shows the PCI Express (PIPE) rate switch circuitry in PCI Express (PIPE) ×8 mode configured at Gen2 (5 Gbps) data rate.

Figure 1–107. Dynamic Switch Signaling in PCI Express (PIPE) x8 Mode



In PCI Express (PIPE) ×8 mode configured at 5 Gbps data rate, when the PCI Express (PIPE) rate switch controller sees a transition on the `rateswitch` signal, it sends control signal `pcie_gen2switch` to the PCI Express (PIPE) clock switch circuitry in the CMU0 clock divider of the master transceiver block and the receiver CDR in all eight bonded channels to switch to the instructed signaling rate. A low-to-high transition on the `rateswitch` signal initiates a Gen1 (2.5 Gbps) to Gen2 (5 Gbps) signaling rate switch. A high-to-low transition on the `rateswitch` signal initiates a Gen2 (5 Gbps) to Gen1 (2.5 Gbps) signaling rate switch.

Table 1–40 shows the transceiver clock frequencies when switching between 2.5 Gbps and 5 Gbps signaling rates.

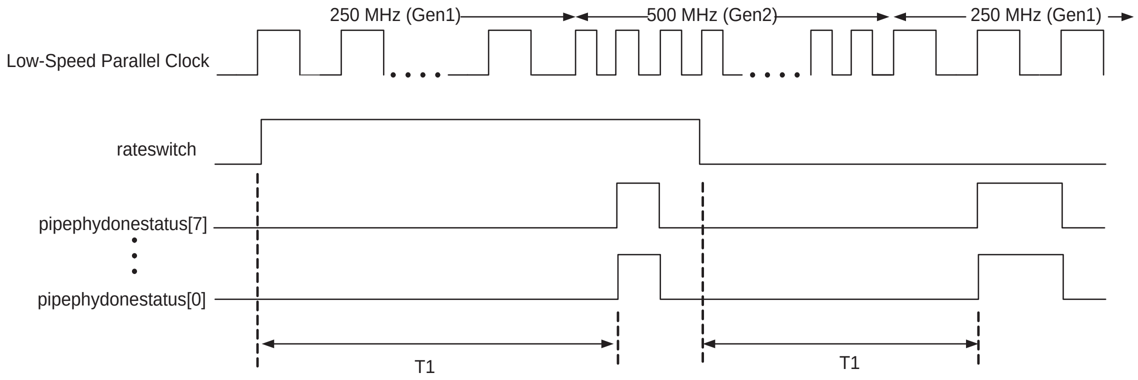
Table 1–40. Transceiver Clock Frequencies Signaling Rates in PCI Express (PIPE) ×8 Mode

Transceiver Clocks	Gen1 (2.5 Gbps) to Gen 2 (5 Gbps) Switch (Low-to-High Transition on the rateswitch Signal)	Gen2 (5 Gbps) to Gen1 (2.5 Gbps) Switch (High-to-Low Transition on the rateswitch Signal)
High-Speed Serial Clock	1.25 GHz to 2.5 GHz	2.5 GHz to 1.25 GHz
Low-Speed Parallel Clock	250 MHz to 500 MHz	500 MHz to 250 MHz
Serial Recovered Clock	1.25 GHz to 2.5 GHz	2.5 GHz to 1.25 GHz
Parallel Recovered Clock	250 MHz to 500 MHz	500 MHz to 250 MHz
FPGA Fabric-Transceiver Interface Clock	125 MHz to 250 MHz	250 MHz to 125 MHz

The PCI Express (PIPE) clock switch circuitry in the CMU0 clock divider of the master transceiver block performs the clock switch between 250 MHz and 500 MHz on the low-speed parallel clock when switching between Gen1 (2.5 Gbps) and Gen2 (5 Gbps) signaling rates. It indicates successful completion of clock switch on the `pcie_gen2switchdone` signal to the PCI Express (PIPE) rate switch controller. The PCI Express (PIPE) rate switch controller forwards the clock switch completion status to the PIPE interface block. The PIPE interface block communicates the clock switch completion status to the PHY-MAC layer by asserting the `pipephydonestatus` signal of all eight bonded channels for one parallel clock cycle.

Figure 1–108 shows low-speed parallel clock switch between Gen1 (250 MHz) and Gen2 (500 MHz) in response to the change in logic level on the `rateswitch` signal. The rate switch completion is shown marked with a one clock cycle assertion of the `pipephydonestatus` signal of all eight bonded channels.

Figure 1–108. Low-Speed Parallel Clock Switching in PCI Express (PIPE) ×8 Mode *(Note 1)*



Note to Figure 1–108:

(1) Time T1 from a transition on the `rateswitch` signal to the assertion of `pipephydonestatus` is pending characterization.

As a result of the signaling rate switch between Gen1 (2.5 Gbps) and Gen2 (5 Gbps), the FPGA fabric-transceiver interface clock switches between 125 MHz and 250 MHz. The FPGA fabric-transceiver interface clock clocks the read side and write side of the transmitter phase compensation FIFO and the receiver phase compensation FIFO of all eight bonded channels, respectively. It is also routed to the FPGA fabric on a global or regional clock resource and looped back to clock the write port and read port of the transmitter phase compensation FIFO and the receiver phase compensation FIFO, respectively. Due to the routing delay between the write and read clock of the transmitter and receiver phase compensation FIFOs, the write pointers and read pointers might collide during a rate switch between 125 MHz and 250 MHz. To avoid collision of the phase compensation FIFO pointers, the PCI Express (PIPE) rate switch controller automatically disables and resets the phase compensation FIFO pointers of all eight bonded channels during clock switch. When the PCI Express (PIPE) clock switch circuitry in the local clock divider indicates successful clock switch completion, the PCI Express (PIPE) rate switch controller releases the phase compensation FIFO pointer resets.

PCI Express Cold Reset Requirements

The PCI Express Base Specification 2.0 defines the following three types of conventional resets to the PCI Express system components:

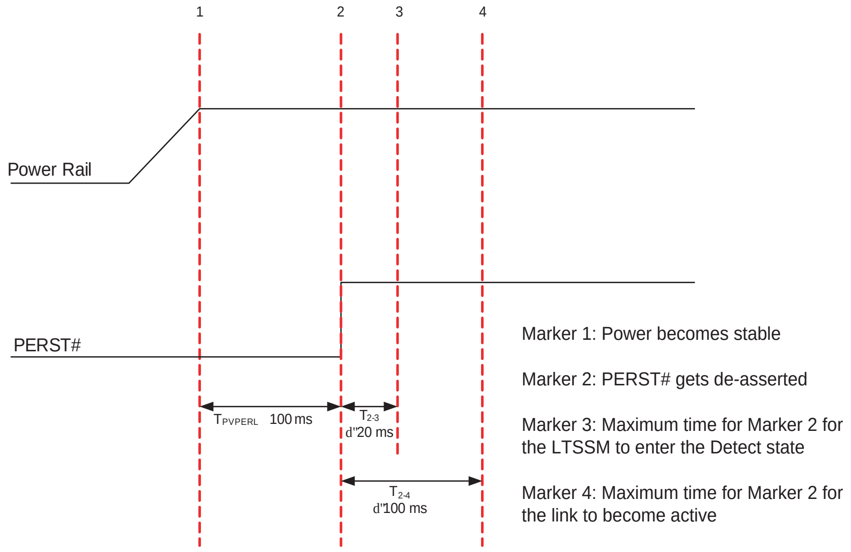
- Cold Reset—fundamental reset after power up
- Warm Reset—fundamental reset without removal and re-application of power
- Hot Reset—In-band conventional reset initiated by higher layer by setting the Hot Reset bit in the TS1 or TS2 training sequences

The fundamental reset is provided by the system to the component or adapter card via the auxiliary signal PERST#. The PCI Express Base Specification 2.0 specifies that the PERST# must be kept asserted for a minimum of 100 ms (TPVPERL) after the system power becomes stable in a cold reset situation. Additionally, all system components must enter the LTSSM Detect state within 20 ms and the link must become active within 100 ms after de-assertion of the PERST# signal. This implies that each PCI Express system component must become active within 200 ms after the power becomes stable.

☞ The link being active is interpreted as the physical layer device coming out of Electrical Idle in L0 state of the LTSSM state machine.

Figure 1–109 shows the PCI Express cold reset timing requirements.

Figure 1–109. PCI Express Cold Reset Requirements


1
2
3
4
Power Rail
PERST#
Marker 1: Power becomes stable
Marker 2: PERST# gets de-asserted
Marker 3: Maximum time for Marker 2 for the LTSSM to enter the Detect state
Marker 4: Maximum time for Marker 2 for the link to become active
T_{PVPERL} 100 ms
T_{2-3}
d"20 ms
T_{2-4}
d"100 ms

Time taken by a PCI Express port implemented using the Stratix IV GX device to go from power up to link active state is stated below:

- Power On Reset—begins after power rails become stable. Typically takes 12 ms
- FPGA Configuration/Programming—begins after power on reset. Configuration time depends on the FPGA density
- Time taken from de-assertion of PERST# to link active—typically takes 40 ms (pending characterization and verification of the PCI Express soft IP and hard IP)

To meet the PCI Express specification of 200 ms from power on to link active, the Stratix IV GX device configuration time must be less than 148 ms (200 ms - 12 ms for power on reset - 40 ms for link to become active after PERST# de-assertion)

Table 1–41 shows typical configuration times for Stratix IV GX devices when configured using Fast Passive Parallel (FPP) configuration scheme at 125 MHz.

Table 1–41. Typical Configuration Times for Stratix IV GX Devices Configured with Fast Passive Parallel (Part 1 of 2)

Stratix IV GX Device	Configuration Time (ms)
EP4SGX70	45
EP4SGX110	45
EP4SGX230	92
EP4SGX290	123
EP4SGX360	123

Table 1–41. Typical Configuration Times for Stratix IV GX Devices Configured with Fast Passive Parallel (Part 2 of 2)

Stratix IV GX Device	Configuration Time (ms)
EP4SGX530	*(1)*

Note to Table 1–41:

(1) Pending characterization.

For more information about FPP configuration scheme, refer to the *Configuration, Design Security, Remote System Upgrades with Stratix IV Devices* chapter in volume 1 in the *Stratix IV Device Handbook.*

Most flash memories available in the market can run up to 100 MHz. To configure the Stratix IV GX devices at 125 MHz, Altera recommends using a MAX II device to convert the 16-bit flash memory output at 62.5 MHz to 8-bit configuration data input to the Stratix IV GX devices at 125 MHz.

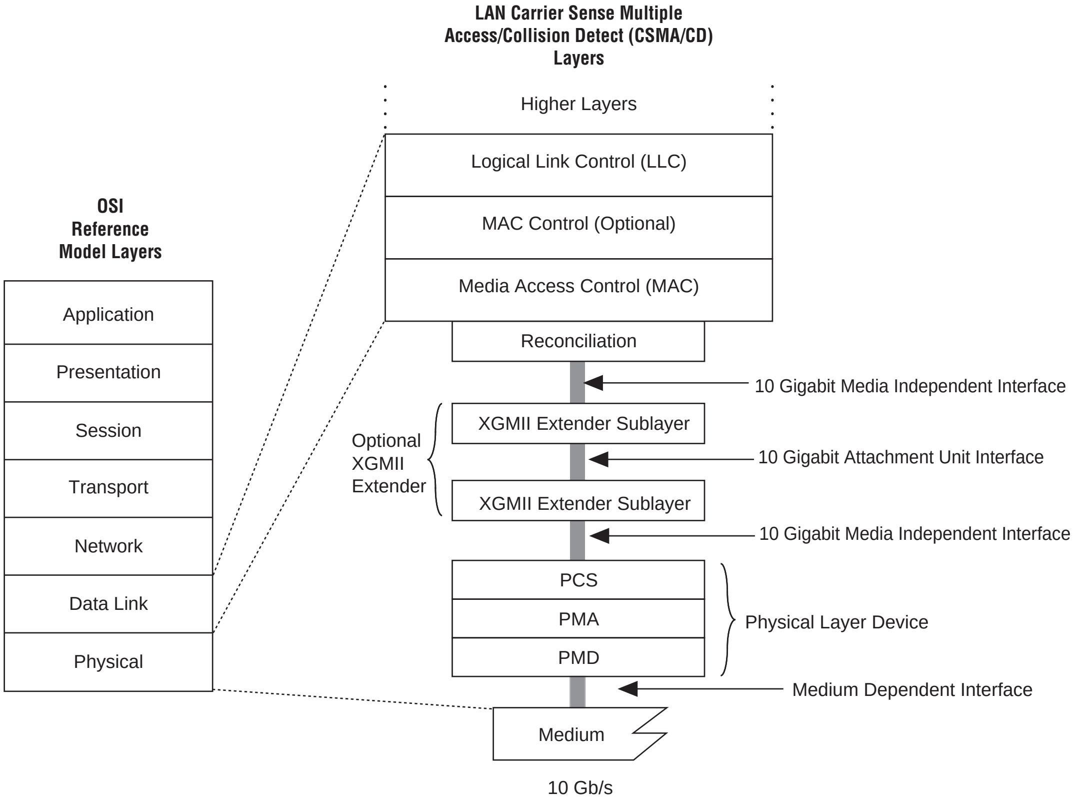
XAUI Mode

The XAUI is an optional, self-managed interface that you can insert between the reconciliation sublayer and the PHY layer to transparently extend the physical reach of the XGMII.

XAUI addresses several physical limitations of the XGMII. XGMII signaling is based on the HSTL Class 1 single-ended I/O standard, which has an electrical distance limitation of approximately 7 cm. Because XAUI uses a low-voltage differential signaling method, the electrical limitation is increased to approximately 50 cm. Another advantage of XAUI is simplification of backplane and board trace routing. XGMII is composed of 32 transmit channels, 32 receive channels, 1 transmit clock, 1 receive clock, 4 transmitter control characters, and 4 receive control characters for a 74-pin wide interface in total. XAUI, on the other hand, only consists of 4 differential transmitter channels and 4 differential receiver channels for a 16-pin wide interface in total. This reduction in pin count significantly simplifies the routing process in the layout design.

Figure 1–110 shows the relationships between the XGMII and XAUI layers.

Figure 1–110. XAUI and XGMII Layers


LAN Carrier Sense Multiple Access/Collision Detect (CSMA/CD) Layers
Higher Layers
Logical Link Control (LLC)
MAC Control (Optional)
Media Access Control (MAC)
Reconciliation
10 Gigabit Media Independent Interface
OSI Reference Model Layers
Application
Presentation
Session
Transport
Network
Data Link
Physical
Optional XGMII Extender
XGMII Extender Sublayer
10 Gigabit Attachment Unit Interface
XGMII Extender Sublayer
10 Gigabit Media Independent Interface
PCS
PMA
PMD
Physical Layer Device
Medium Dependent Interface
Medium
10 Gb/s

The XGMII interface consists of four lanes of 8 bits. At the transmit side of the XAUI interface, the data and control characters are converted within the XGXS into an 8B/10B encoded data stream. Each data stream is then transmitted across a single differential pair running at 3.125 Gbps (3.75 Gbps for HiGig). At the XAUI receiver, the incoming data is decoded and mapped back to the 32-bit XGMII format. This provides a transparent extension of the physical reach of the XGMII and also reduces the interface pin count.

XAUI functions as a self-managed interface because code group synchronization, channel deskew, and clock domain decoupling is handled with no upper layer support requirements. This functionality is based on the PCS code groups that are used during the IPG time and idle periods. PCS code groups are mapped by the XGXS to XGMII characters specified in Table 1–42.

Table 1–42. XGMII Character to PCS Code-Group Mapping (Part 1 of 2)

XGMII TXC	XGMII TXD *(1)*	PCD Code Group	Description
0	00 through FF	Dxx,y	Normal data transmission
1	07	K28.0 or K28.3 or K28.5	Idle in ||I||
1	07	K28.5	Idle in ||T||
1	9C	K28.4	Sequence
1	FB	K27.7	Start

Table 1–42. XGMII Character to PCS Code-Group Mapping (Part 2 of 2)

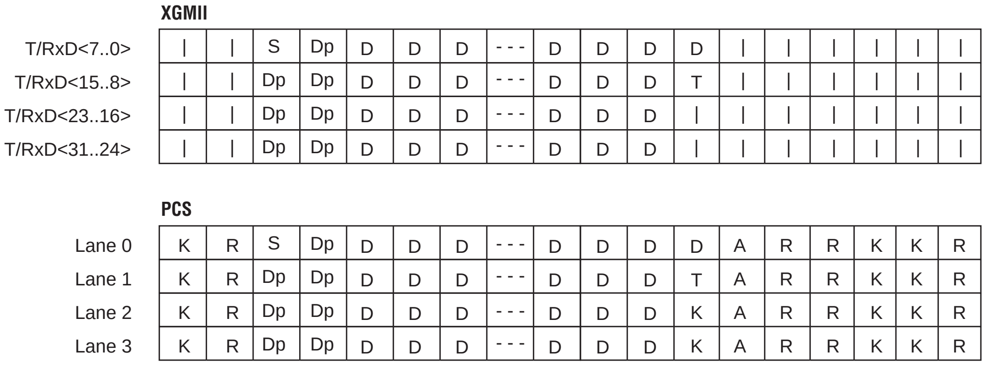
XGMII TXC	XGMII TXD (1)	PCD Code Group	Description
1	FD	K29.7	Terminate
1	FE	K30.7	Error
1	Any other value	K30.7	Invalid XGMII character

Note to Table 1–42:

(1) The values in the XGMII TXD column are in hexadecimal.

Figure 1–111 shows an example of mapping between XGMII characters and the PCS code groups that are used in XAUI. The idle characters are mapped to a pseudo-random sequence of /A/, /R/, and /K/ code groups.

Figure 1–111. Example of Mapping XGMII Characters to PCS Code Groups



XGMII

T/RxD<7..0>	|	|	S	Dp	D	D	D	- - -	D	D	D	D	|	|	|	|	|	|
T/RxD<15..8>	|	|	Dp	Dp	D	D	D	- - -	D	D	D	T	|	|	|	|	|	|
T/RxD<23..16>	|	|	Dp	Dp	D	D	D	- - -	D	D	D	|	|	|	|	|	|	|
T/RxD<31..24>	|	|	Dp	Dp	D	D	D	- - -	D	D	D	|	|	|	|	|	|	|

PCS

Lane 0	K	R	S	Dp	D	D	D	- - -	D	D	D	D	A	R	R	K	K	R
Lane 1	K	R	Dp	Dp	D	D	D	- - -	D	D	D	T	A	R	R	K	K	R
Lane 2	K	R	Dp	Dp	D	D	D	- - -	D	D	D	K	A	R	R	K	K	R
Lane 3	K	R	Dp	Dp	D	D	D	- - -	D	D	D	K	A	R	R	K	K	R

The PCS code groups are sent via PCS ordered sets. PCS ordered sets consist of combinations of special and data code groups defined as a column of code groups. These ordered sets are composed of four code groups beginning in lane 0. Table 1–43 lists the defined idle ordered sets (||I||) that are used for the self-managed properties of XAUI.

Table 1–43. Defined Idle Ordered Set

Code	Ordered Set	Number of Code Groups	Encoding
||I||	**Idle**		**Substitute for XGMII Idle**
||K||	Synchronization column	4	/K28.5/K28.5/K28.5/K28.5/
||R||	Skip column	4	/K28.0/K28.0/K28.0/K28.0/
||A||	Align column	4	/K28.3/K28.3/K28.3/K28.3/

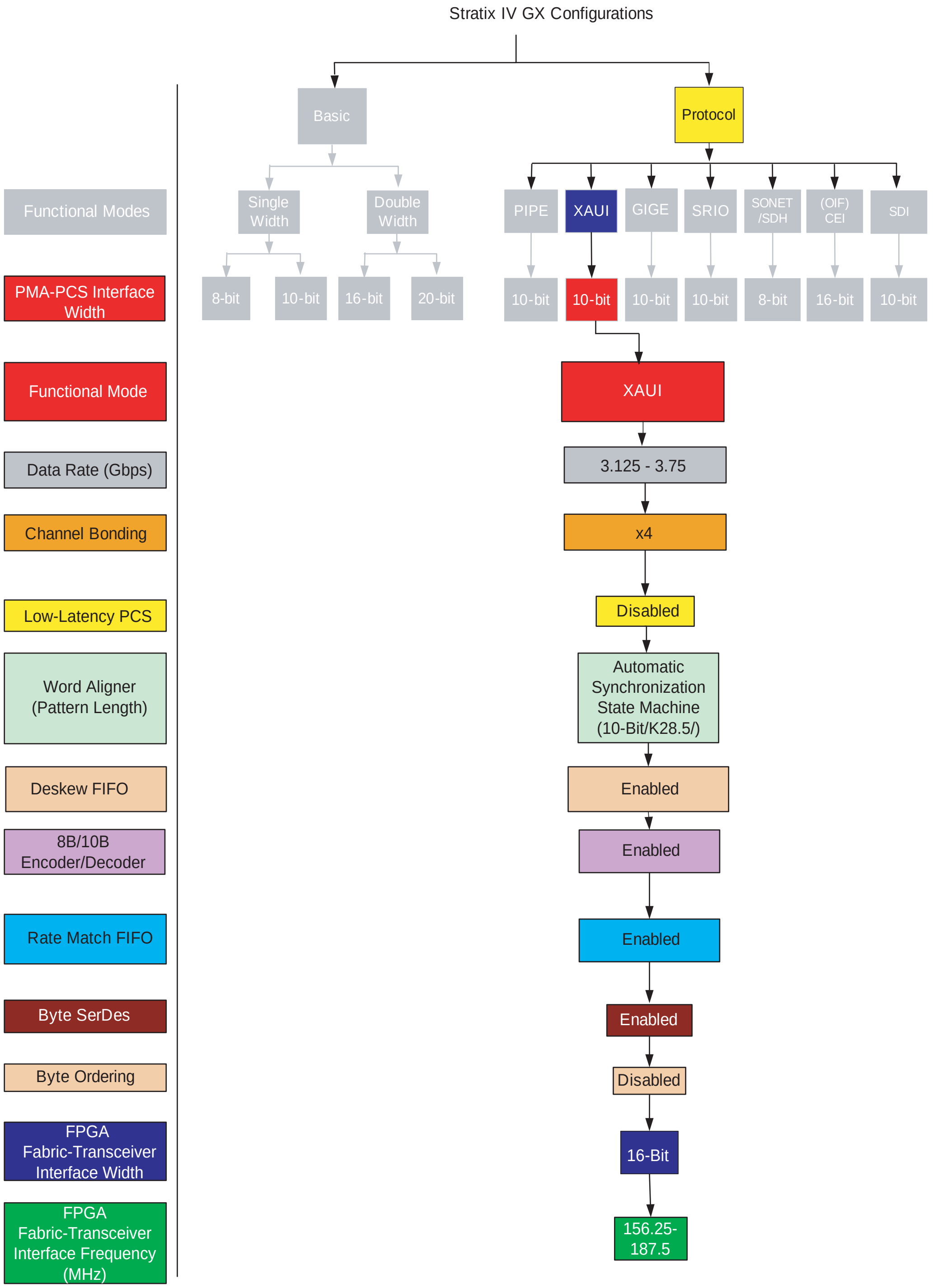
Stratix IV GX transceivers configured in XAUI mode provide the following protocol features:

- XGMII-to-PCS code conversion at the transmitter
- PCS-to-XGMII code conversion at the receiver
- 8B/10B encoding and decoding
- IEEE P802.3ae-compliant synchronization state machine

- ±100 PPM clock rate compensation
- Channel deskew of four lanes of the XAUI link

Figure 1–112 shows the XAUI mode configuration supported in Stratix IV GX devices.

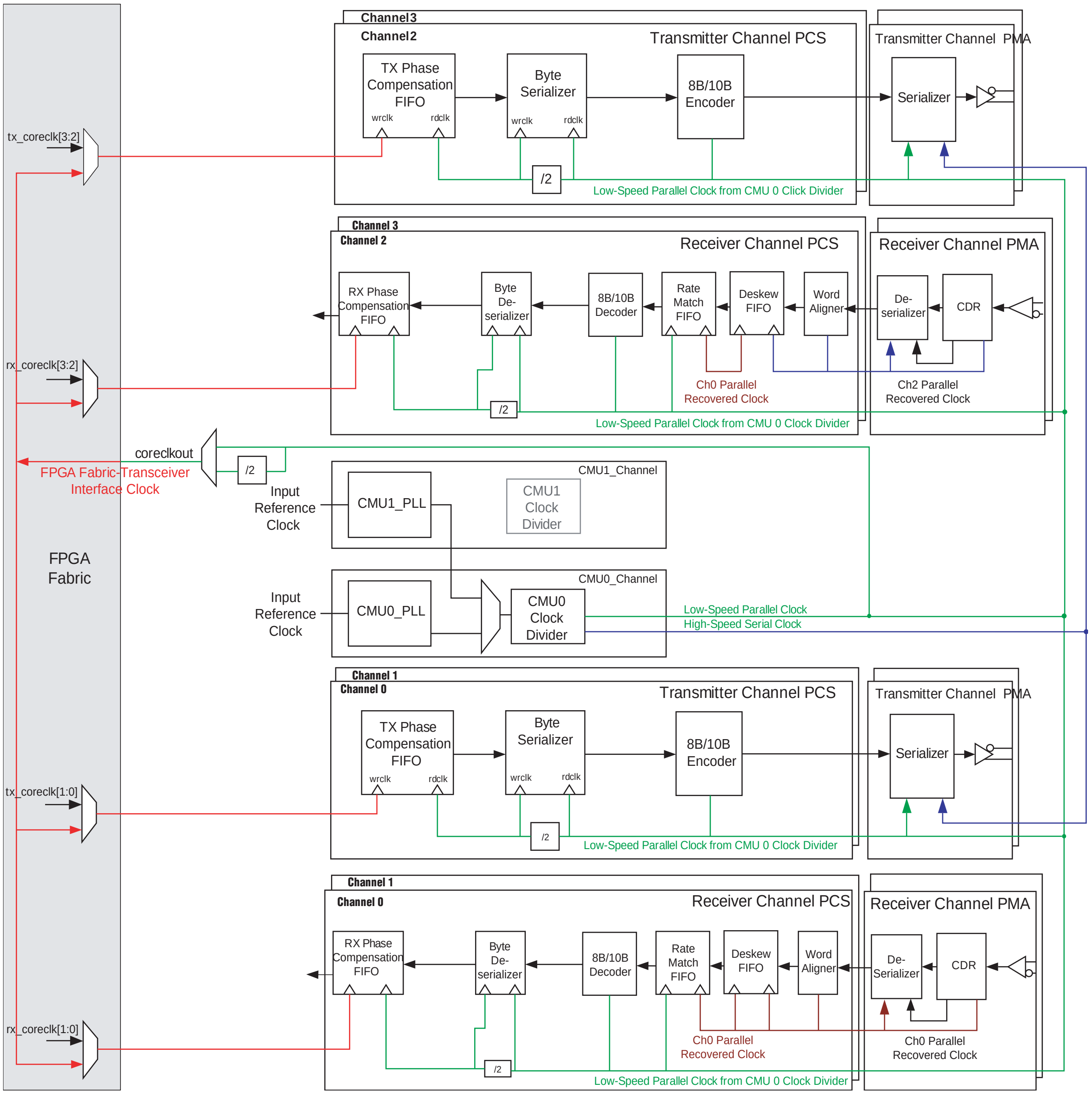
Figure 1–112. Stratix IV GX XAUI Mode Configuration



XAUI Mode Datapath

Figure 1–113 shows the ALTGX megafunction transceiver datapath when configured in XAUI mode.

Figure 1–113. Transceiver Datapath in XAUI Mode



XGMII-To-PCS Code Conversion at the Transmitter

In XAUI mode, the 8b/10b encoder in the Stratix IV GX transmitter datapath is controlled by a transmitter state machine that maps various 8-bit XGMII codes to 10-bit PCS code groups. This state machine complies with the IEEE P802.3ae PCS transmit source state diagram shown in Figure 1–114.

Figure 1–114. XGMII-To-PCS Code Conversion in XAUI Mode

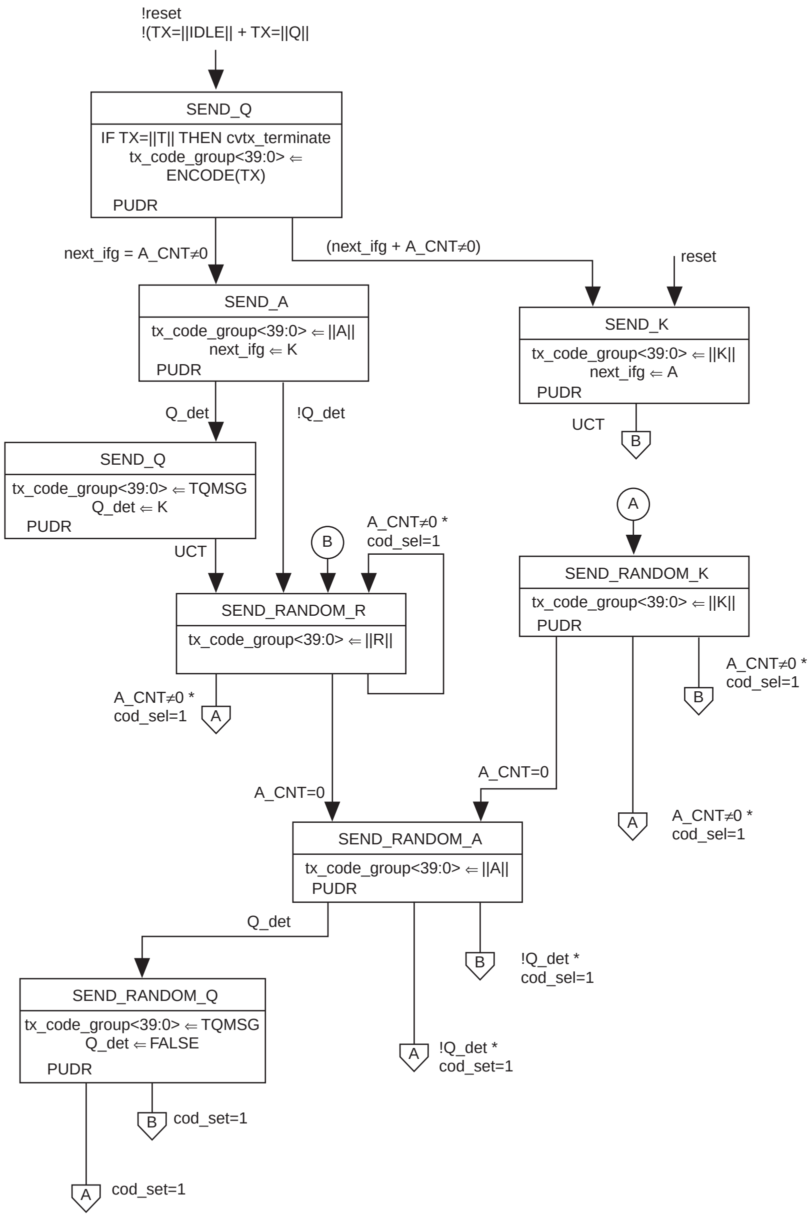


Table 1–44 lists the XGMII-to-PCS code group conversion in XAUI functional mode. The XGMII TXC control signal is equivalent to the `tx_ctrlenable` signal; the XGMII TXD control signal is equivalent to the `tx_datain[7:0]` signal.

Table 1–44. XGMII Character to PCS Code-Group Mapping (Part 1 of 2)

XGMII TXC	XGMII TXD (1)	PCD Code Group	Description
0	00 through FF	Dxx,y	Normal data transmission
1	07	K28.0 or K28.3 or K28.5	Idle in ||I||

Table 1–44. XGMII Character to PCS Code-Group Mapping (Part 2 of 2)

XGMII TXC	XGMII TXD *(1)*	PCD Code Group	Description
1	07	K28.5	Idle in ||T||
1	9C	K28.4	Sequence
1	FB	K27.7	Start
1	FD	K29.7	Terminate
1	FE	K30.7	Error
1	Any other value	K30.7	Invalid XGMII character

Note to Table 1–42:

(1) The values in the XGMII TXD column are in hexadecimal.

PCS-To-XGMII Code Conversion at the Receiver

In XAUI mode, the 8b/10b decoder in the Stratix IV GX receiver datapath is controlled by a XAUI receiver state machine that converts received PCS code groups into specific 8-bit XGMII codes.

Table 1–45 lists the PCS-to-XGMII code group conversion in XAUI functional mode. The XGMII RXC control signal is equivalent to the `rx_ctrldetect` signal; the XGMII RXD control signal is equivalent to the `rx_dataout[7:0]` signal.

Table 1–45. PCS Code Group to XGMII Character Mapping

XGMII RXC	XGMII RXD *(1)*	PCD Code Group	Description
0	00 through FF	Dxx,y	Normal data transmission
1	07	K28.0 or K28.3 or K28.5	Idle in ||I||
1	07	K28.5	Idle in ||T||
1	9C	K28.4	Sequence
1	FB	K27.7	Start
1	FD	K29.7	Terminate
1	FE	K30.7	Error
1	FE	Invalid code group	Received code group

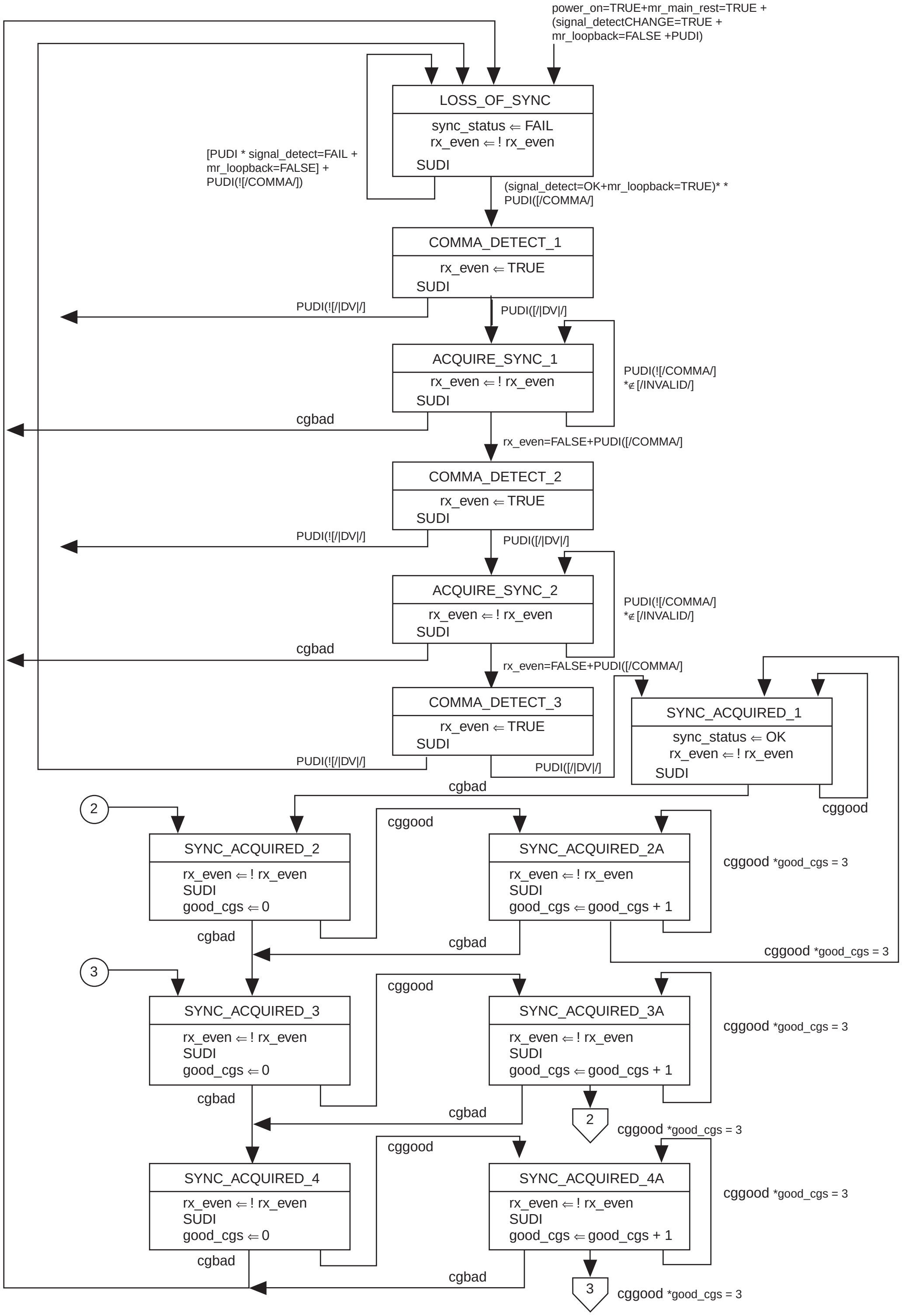
Note to Table 1–42:

(1) The values in the XGMII RXD column are in hexadecimal.

Word Aligner

The word aligner in XAUI functional mode is configured in automatic synchronization state machine mode. The Quartus II software automatically configures the synchronization state machine to indicate synchronization when the receiver receives four /K28.5/ comma code groups without intermediate invalid code groups. The synchronization state machine implemented in XAUI mode is compliant to the PCS synchronization state diagram specified in Clause 48 of the IEEE P802.3ae specification and is shown in Figure 1–115.

Figure 1–115. IEEE 802.3ae PCS Synchronization State Diagram



Receiver synchronization is indicated on the `rx_syncstatus` port of each channel. A high on the `rx_syncstatus` port indicates that the lane is synchronized; a low on the `rx_syncstatus` port indicates that it has fallen out of synchronization. The receiver loses synchronization when it detects four invalid code groups separated by less than four valid code groups or when it is reset.

Deskew FIFO

Code groups received across four lanes in a XAUI link can be misaligned with respect to one another because of skew in the physical medium or differences between the independent clock recoveries per lane. The XAUI protocol allows a maximum skew of 40 UI (12.8 ns) as seen at the receiver of the four lanes.

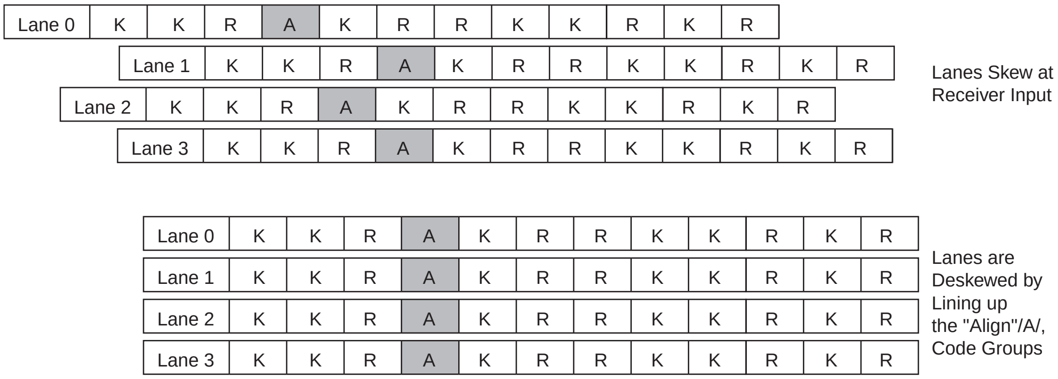
The XAUI protocol requires the physical layer device to implement a deskew circuitry to align all four channels. To enable the deskew circuitry at the receiver to align the four channels, the transmitter sends a /A/ (/K28.3/) code group simultaneously on all four channels during inter-packet gap. The skew introduced in the physical medium and the receiver channels can be /A/ code groups to be received misaligned with respect to each other.

The deskew operation is performed by the deskew FIFO in XAUI functional mode.

The deskew FIFO in each channel receives data from its word aligner. The deskew operation begins only after link synchronization is achieved on all four channels as indicated by a high on the `rx_syncstatus` signal from the word aligner in each channel. Until the first /A/ code group is received, the deskew FIFO read and write pointers in each channel are not incremented. After the first /A/ code group is received, the write pointer starts incrementing for each word received but the read pointer is frozen. If the /A/ code group is received on each of the four channels within 10 recovered clock cycles of each other, the read pointer of all four deskew FIFOs is released simultaneously, aligning all four channels.

Figure 1–116 shows lane skew at the receiver input and how the deskew FIFO uses the /A/ code group to align the channels.

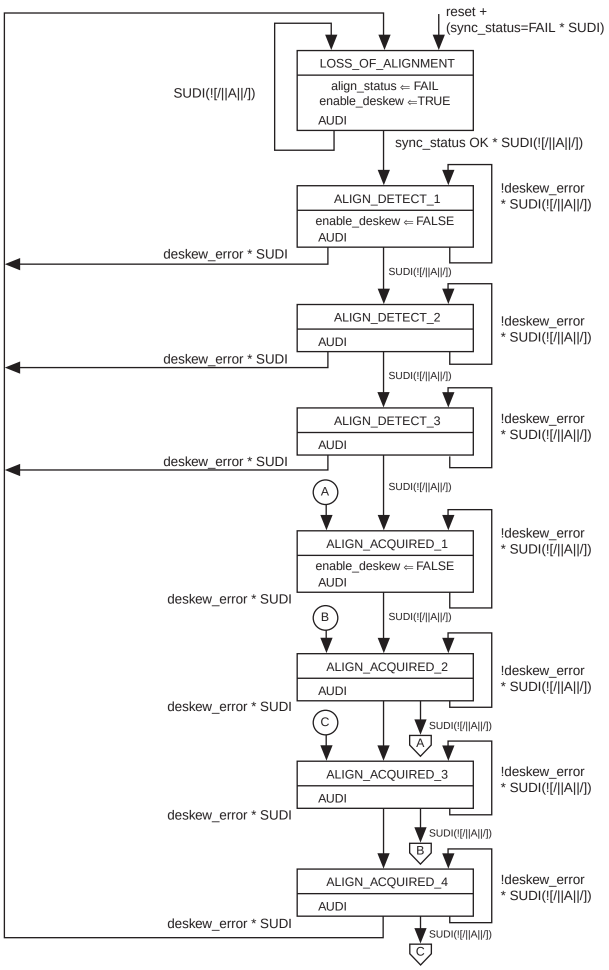
Figure 1–116. Receiver Input Lane Skew in XAUI Mode


Lane 0 K K R A K R R K K R K R
Lane 1 K K R A K R R K K R K R
Lane 2 K K R A K R R K K R K R
Lane 3 K K R A K R R K K R K R
Lanes Skew at Receiver Input
Lane 0 K K R A K R R K K R K R
Lane 1 K K R A K R R K K R K R
Lane 2 K K R A K R R K K R K R
Lane 3 K K R A K R R K K R K R
Lanes are Deskewed by Lining up the "Align"/A/, Code Groups

After alignment of the first ||A|| column, if three additional aligned ||A|| columns are observed at the output of the deskew FIFOs of the four channels, the `rx_channelaligned` signal is asserted high, indicating channel alignment is acquired. After acquiring channel alignment, if four misaligned ||A|| columns are seen at the output of the deskew FIFOs in all four channels with no aligned ||A|| columns in between, the `rx_channelaligned` signal is deasserted low, indicating loss of channel alignment.

The deskew FIFO operation in XAUI functional mode is compliant to the PCS deskew state machine diagram specified in clause 48 of the IEEE P802.3ae, as shown in Figure 1–117.

Figure 1–117. Deskew FIFO in XAUI Mode



Rate Match FIFO

In XAUI mode, the rate match FIFO is capable of compensating up to ±100 PPM (total 200 PPM total) difference between the upstream transmitter and the local receiver reference clock. The XAUI protocol requires the transmitter to send /R/ (/K28.0/) code groups simultaneously on all four lanes (denoted as ||R|| column) during inter-packet gaps, adhering to rules listed in the IEEE P802.3ae specification. The rate match FIFO operation in XAUI mode is compliant to the IEEE P802.3ae specification.

The rate match operation begins after:

- The synchronization state machine in the word aligner of all four channels indicates synchronization acquired by driving its `rx_syncstatus` signal high
- The deskew FIFO indicates alignment acquired by driving the `rx_channelaligned` signal high

The rate match FIFO looks for the ||R|| column (simultaneous /R/ code group on all four channels) and deletes or inserts ||R|| column to prevent the rate match FIFO from overflowing or under running. The rate match FIFO can insert or delete as many ||R|| columns as necessary to perform the rate match operation.

Two flags, `rx_rmfifodatadeleted` and `rx_rmfifodatainserted`, that indicate rate match FIFO deletion and insertion events, respectively, are forwarded to the FPGA fabric. If an ||R|| column is deleted, the `rx_rmfifodeleted` flag from each of the four channels goes high for one clock cycle per deleted ||R|| column. If an ||R|| column is inserted, the `rx_rmfifoinserted` flag from each of the four channels goes high for one clock cycle per inserted ||R|| column.

Figure 1–118 shows an example of rate match deletion in the case where three ||R|| columns are required to be deleted.

Figure 1–118. Rate Match Deletion in XAUI Mode

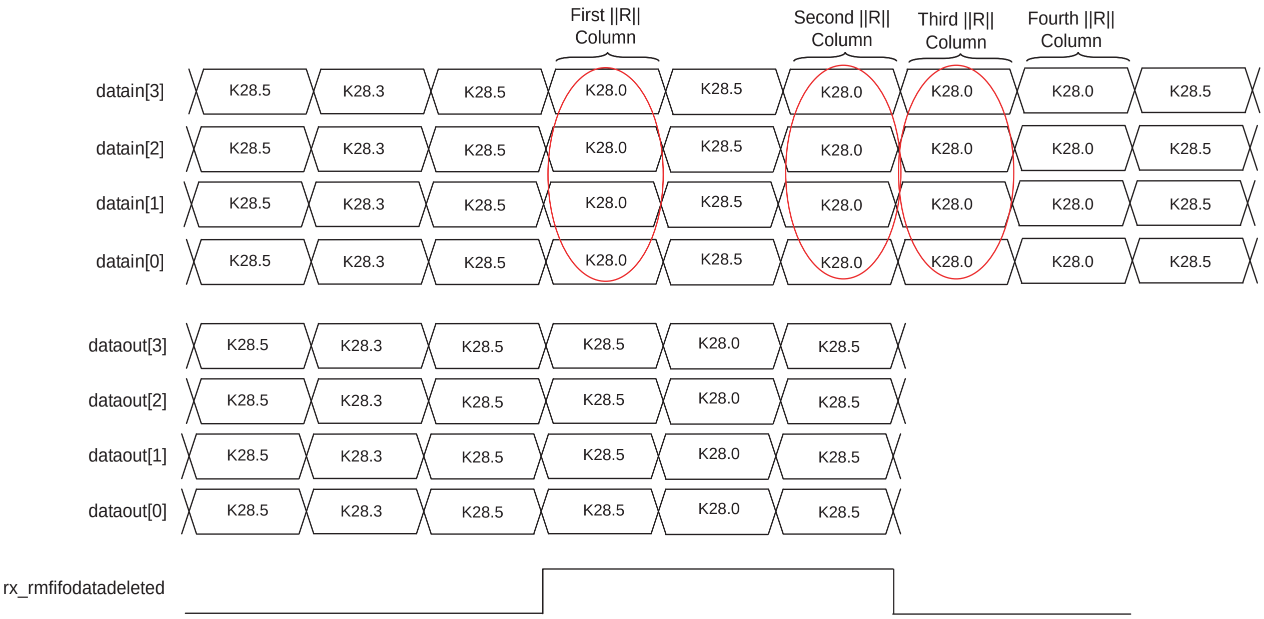
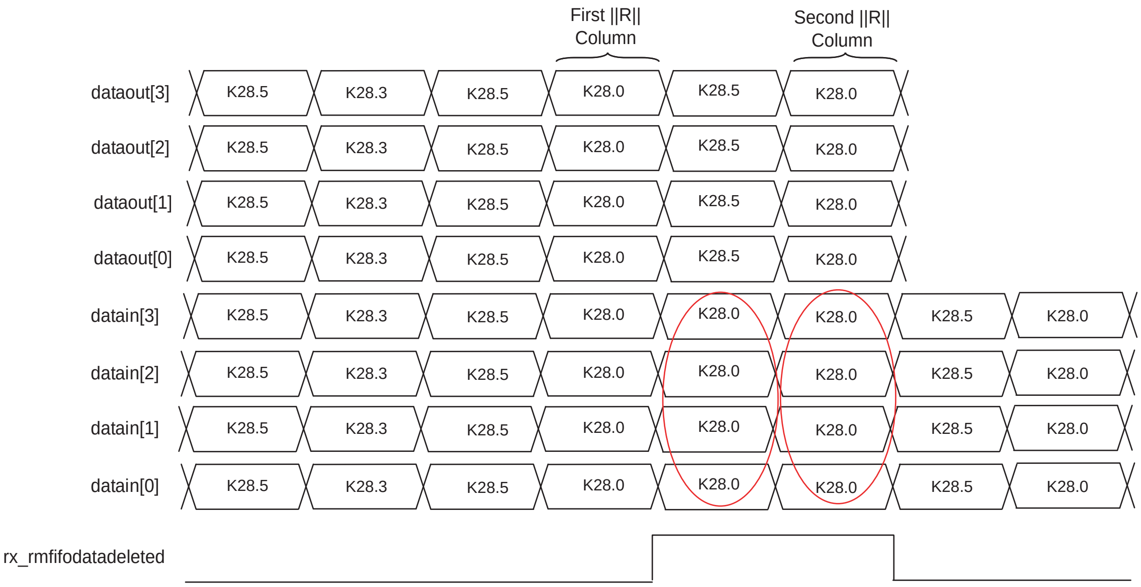


Figure 1–119 shows an example of rate match insertion in the case where two ||R|| columns are required to be inserted.

Figure 1–119. Rate Match Insertion in XAUI Mode


First ||R|| Column
Second ||R|| Column
dataout[3] K28.5 K28.3 K28.5 K28.0 K28.5 K28.0
dataout[2] K28.5 K28.3 K28.5 K28.0 K28.5 K28.0
dataout[1] K28.5 K28.3 K28.5 K28.0 K28.5 K28.0
dataout[0] K28.5 K28.3 K28.5 K28.0 K28.5 K28.0
datain[3] K28.5 K28.3 K28.5 K28.0 K28.0 K28.0 K28.5 K28.0
datain[2] K28.5 K28.3 K28.5 K28.0 K28.0 K28.0 K28.5 K28.0
datain[1] K28.5 K28.3 K28.5 K28.0 K28.0 K28.0 K28.5 K28.0
datain[0] K28.5 K28.3 K28.5 K28.0 K28.0 K28.0 K28.5 K28.0
rx_rmfifodatadeleted

GIGE Mode

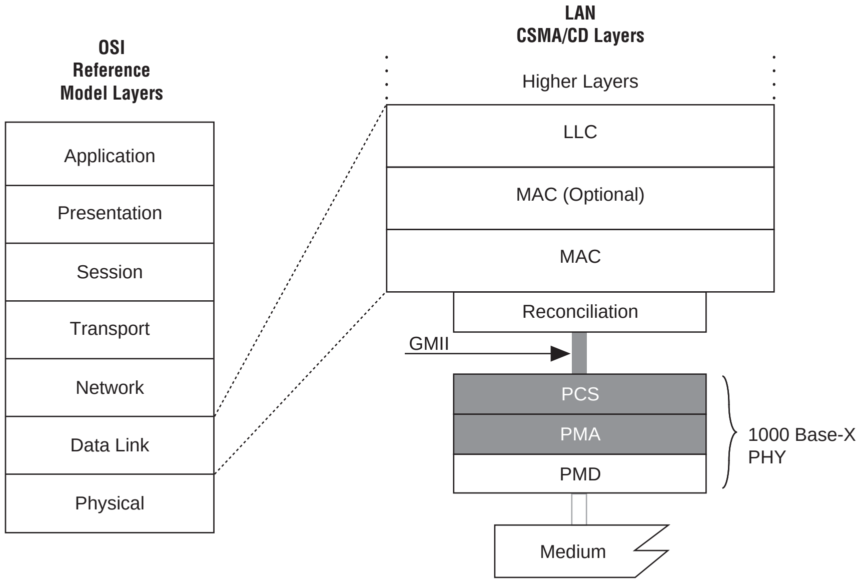
IEEE 802.3 defines the 1000 Base-X PHY as an intermediate, or transition, layer that interfaces various physical media with the media access control (MAC) in a gigabit ethernet system. It shields the MAC layer from the specific nature of the underlying medium. The 1000 Base-X PHY is divided into three sublayers:

- the physical coding sublayer
- the physical media attachment
- the physical medium dependent (PMD)

The PCS sublayer interfaces with the MAC through the gigabit medium independent interface (GMII). The 1000 Base-X PHY defines a physical interface data rate of 1 Gbps.

Figure 1–120 shows the 1000 Base-X PHY position in a Gigabit Ethernet OSI reference model.

Figure 1–120. 1000 Base-X PHY in a Gigabit Ethernet OSI Reference Model



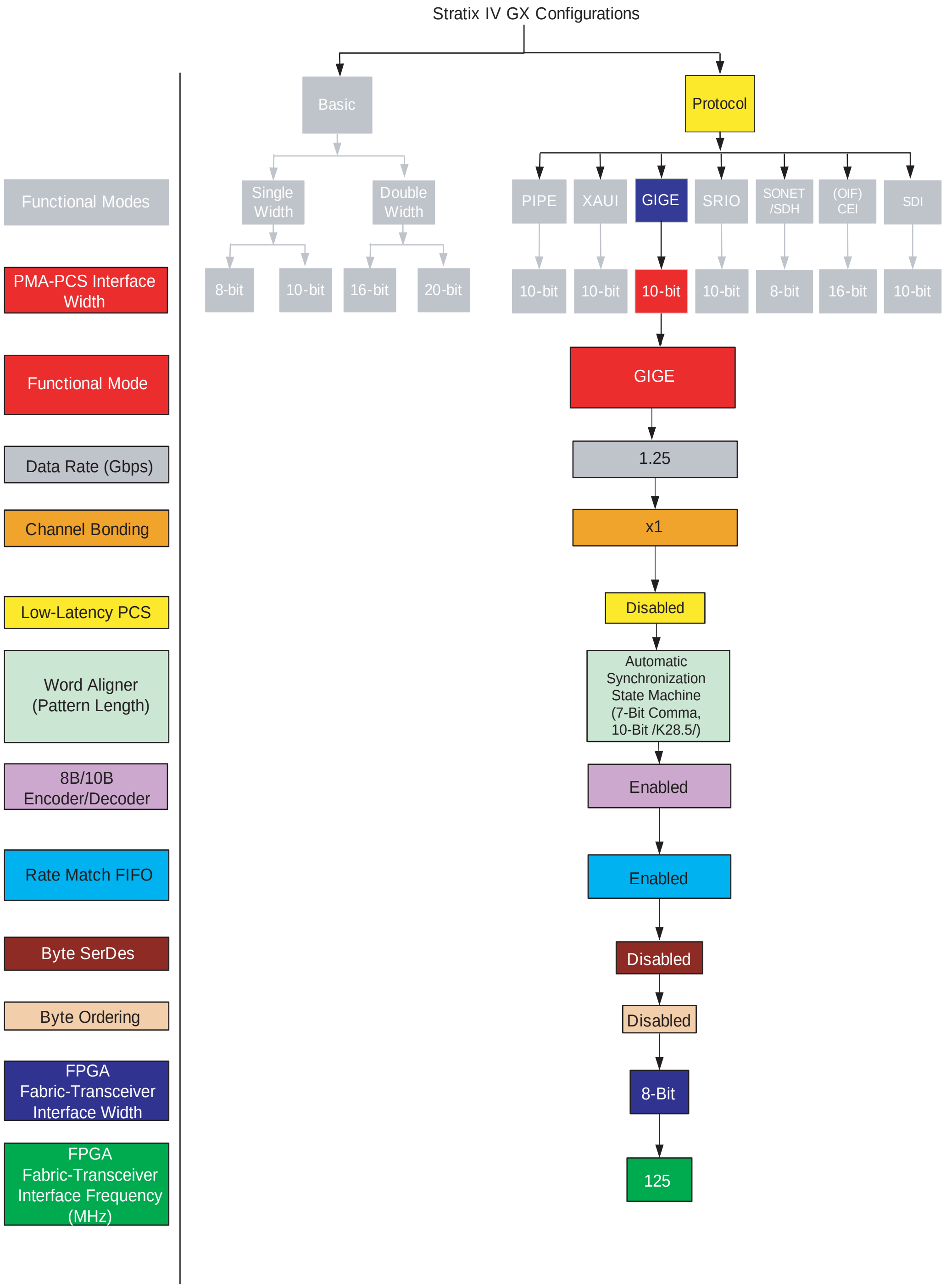
Stratix IV GX transceivers, when configured in GIGE functional mode, have built-in circuitry to support the following PCS and PMA functions defined in the IEEE 802.3 specification:

- 8B/10B encoding and decoding
- Synchronization
- Upstream transmitter and local receiver clock frequency compensation (rate matching)
- Clock recovery from the encoded data forwarded by the receiver PMD
- Serialization and deserialization

☞ Stratix IV GX transceivers do not have built-in support for other PCS functions; for example, auto-negotiation state machine, collision-detect, and carrier-sense. If required, you must implement these functions in a PLD logic array or external circuits.

Figure 1–121 shows the GIGE mode configuration supported in Stratix IV GX devices.

Figure 1–121. GIGE Mode



GIGE Mode Datapath

Figure 1–122 shows the transceiver datapath when configured in GIGE functional mode.

Figure 1–122. GIGE Mode Datapath

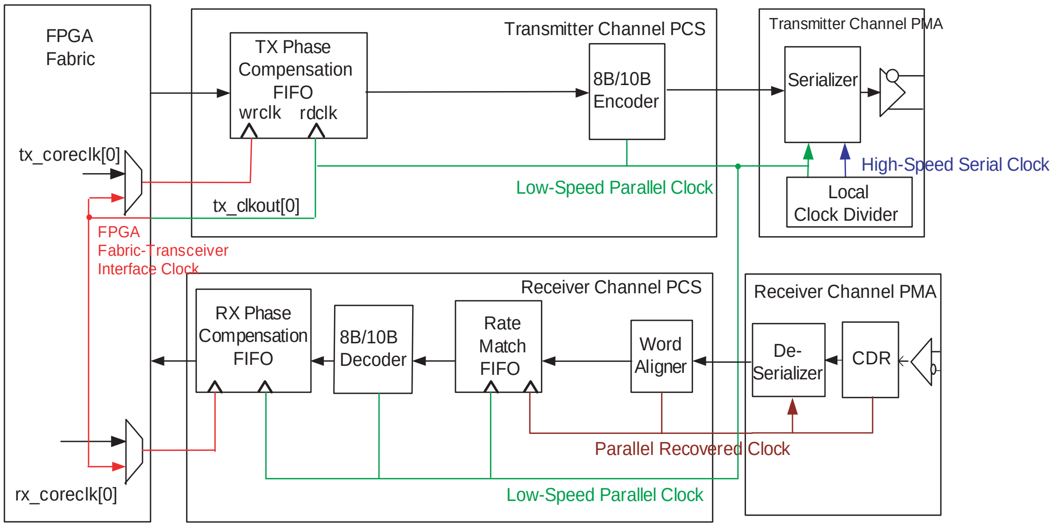


Table 1–46 shows the transceiver datapath clock frequencies in GIGE functional mode.

Table 1–46. Transceiver Datapath Clock Frequencies in GIGE Mode

Functional Mode	Data Rate	High-Speed Serial Clock Frequency	Parallel Recovered Clock and Low-Speed Parallel Clock Frequency	FPGA Fabric-Transceiver Interface Clock Frequency
GIGE	1.25 Gbps	625 MHz	125 MHz	125 MHz

8B/10B Encoder

In GIGE mode, the 8B/10B encoder clocks in 8-bit data and 1-bit control identifier from the transmitter phase compensation FIFO and generates 10-bit encoded data. The 10-bit encoded data is fed to the serializer. Refer to "8B/10B Encoder" on page 1–33 for more information about 8B/10B encoder functionality.

GIGE Protocol—Ordered Sets and Special Code Groups

Table 1–47 lists ordered sets and special code groups specified in the IEEE802.3 specification.

Table 1–47. GIGE Ordered Sets (Part 1 of 2) *(Note 1)*

Code	Ordered Set	Number of Code Groups	Encoding
/C/	Configuration	—	Alternating /C1/ and /C2/
/C1/	Configuration 1	4	/K28.5/D21.5/`Config_Reg` *(1)*
/C2/	Configuration 2	4	/K28.5/D2.2/`Config_Reg` *(1)*
/I/	IDLE	—	Correcting /I1/, Preserving /I2/
/I1/	IDLE 1	2	/K28.5/D5.6
/I2/	IDLE 2	2	/K28.5/D16.2

Table 1–47. GIGE Ordered Sets (Part 2 of 2) *(Note 1)*

Code	Ordered Set	Number of Code Groups	Encoding
	Encapsulation	—	—
/R/	`Carrier_Extend`	1	/K23.7/
/S/	`Start_of_Packet`	1	/K27.7/
/T/	`End_of_Packet`	1	/K29.7/
/V/	`Error_Propagation`	1	/K30.7/

Note to Table 1–47:

(1) Two data code groups representing the `Config_Reg` value.

Idle Ordered-Set Generation

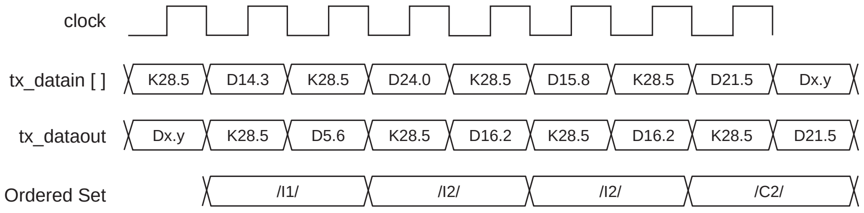
The IEEE 802.3 specification requires the GIGE PHY to transmit idle ordered sets (/I/) continuously and repetitively whenever the GMII is idle. This ensures that the receiver maintains bit and word synchronization whenever there is no active data to be transmitted.

In GIGE functional mode, any /Dx.y/ following a /K28.5/ comma is replaced by the transmitter with either a /D5.6/ (/I1/ ordered set) or a /D16.2/ (/I2/ ordered set), depending on the current running disparity. The exception is when the data following the /K28.5/ is /D21.5/ (/C1/ ordered set) or /D2.2/ (/C2/) ordered set. If the running disparity before the /K28.5/ is positive, an /I1/ ordered set is generated. If the running disparity is negative, a /I2/ ordered set is generated. The disparity at the end of a /I1/ is the opposite of that at the beginning of the /I1/. The disparity at the end of a /I2/ is the same as the beginning running disparity (right before the idle code). This ensures a negative running disparity at the end of an idle ordered set. A /Kx.y/ following a /K28.5/ is not replaced.

Note that /D14.3/, /D24.0/, and /D15.8/ are replaced by /D5.6/ or /D16.2/ (for /I1/, /I2/ ordered sets). /D21.5/ (part of the /C1/ order set) is not replaced.

Figure 1–123 shows the automatic idle ordered set generation.

Figure 1–123. Automatic Ordered Set Generation



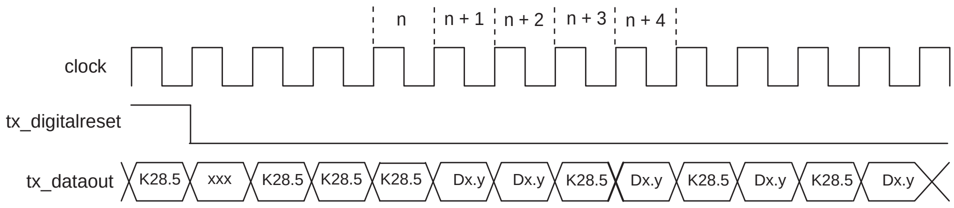
Reset Condition

After de-assertion of `tx_digitalreset`, the GIGE transmitter automatically transmits three /K28.5/ comma code groups before transmitting user data on the `tx_datain` port. This could affect the synchronization state machine behavior at the receiver.

Depending on when you start transmitting the synchronization sequence, there could be an even or odd number of /Dx.y/ code groups transmitted between the last of the three automatically sent /K28.5/ code groups and the first /K28.5/ code group of the synchronization sequence. If there is an even number of /Dx.y/ code groups received between these two /K28.5/ code groups, the first /K28.5/ code group of the synchronization sequence begins at an odd code group boundary (`rx_even` = FALSE). An IEEE802.3-compliant GIGE synchronization state machine treats this as an error condition and goes into Loss of Sync state.

Figure 1–124 shows an example of even numbers of /Dx.y/ between the last automatically sent /K28.5/ and the first user-sent /K28.5/. The first user-sent /K28.5/ code group received at an odd code group boundary in cycle n + 3 takes the receiver synchronization state machine in Loss of Sync state. The first synchronization ordered-set /K28.5/Dx.y/ in cycles n + 3 and n + 4 is discounted and three additional ordered sets are required for successful synchronization.

Figure 1–124. Reset Condition in GIGE Mode



Word Aligner

The word aligner in GIGE functional mode is configured in automatic synchronization state machine mode. The Quartus II software automatically configures the synchronization state machine to indicate synchronization when the receiver receives three consecutive synchronization ordered sets. A synchronization ordered set is a /K28.5/ code group followed by an odd number of valid /Dx.y/ code groups. The fastest way for the receiver to achieve synchronization is to receive three continuous {/K28.5/, /Dx.y/} ordered set.

Receiver synchronization is indicated on the `rx_syncstatus` port of each channel. A high on the `rx_syncstatus` port indicates that the lane is synchronized; a low on the `rx_syncstatus` port indicates that the lane has fallen out of synchronization. The receiver loses synchronization when it detects four invalid code groups separated by less-than-three valid code groups or when it is reset.

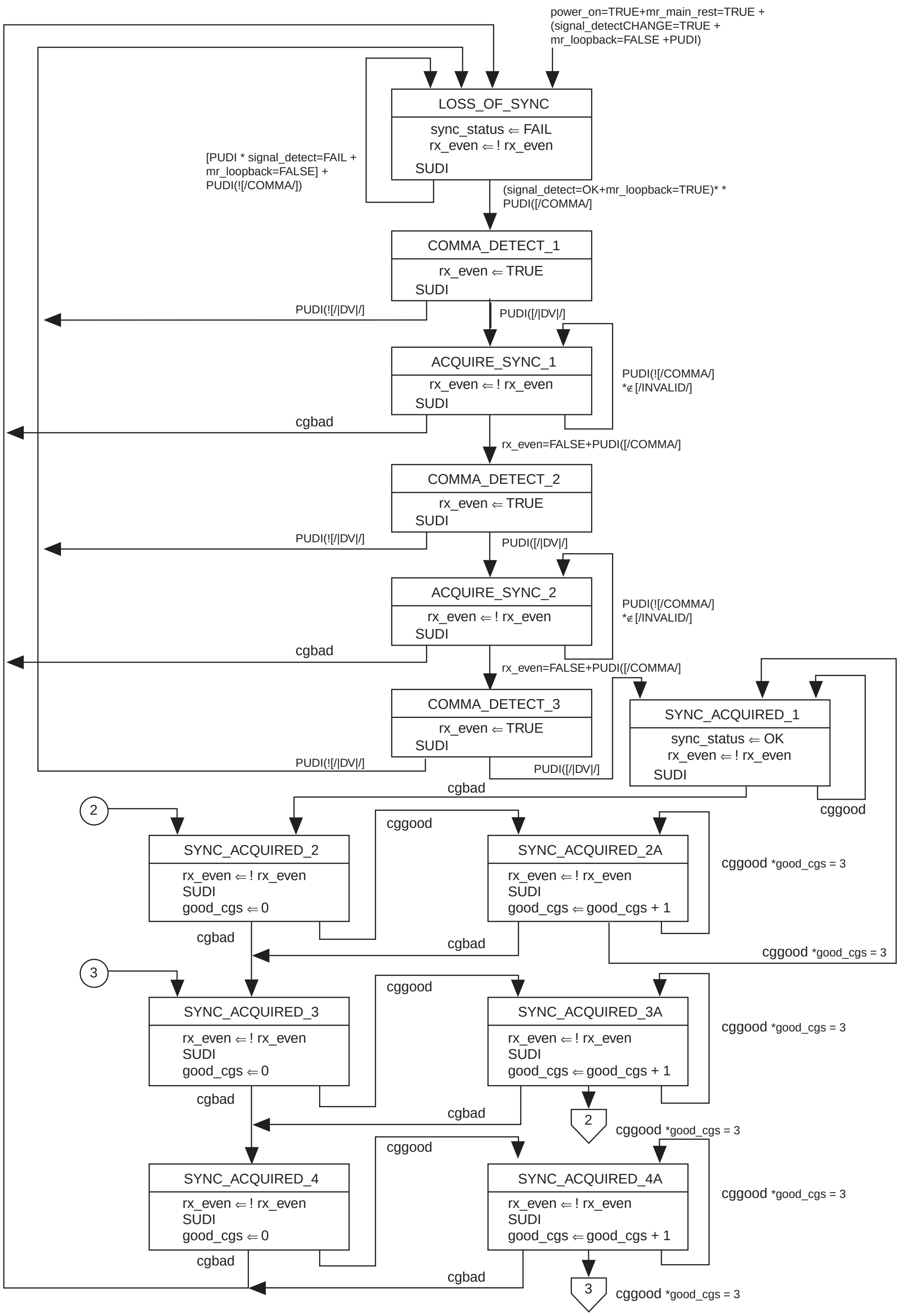
Table 1–48 lists the synchronization state machine parameters when configured in GIGE mode.

Table 1–48. Synchronization State Machine Parameters in GIGE Functional Mode

Synchronization State Machine Parameters	Setting
Number of valid {/K28.5/, /Dx,y/} ordered-sets received to achieve synchronization	3
Number of errors received to lose synchronization	4
Number of continuous good code groups received to reduce the error count by 1	4

Figure 1–125 shows the synchronization state machine implemented in GIGE mode.

Figure 1–125. Synchronization State Machine in GIGE Mode



Rate Match FIFO

In GIGE mode, the rate match FIFO is capable of compensating up to ±100 PPM (total 200 PPM total) difference between the upstream transmitter and the local receiver reference clock. The GIGE protocol requires the transmitter to send idle ordered sets /I1/ (/K28.5/D5.6/) and /I2/ (/K28.5/D16.2/) during inter-packet gaps adhering to the rules listed in the IEEE 802.3 specification.

The rate match operation begins after the synchronization state machine in the word aligner indicates synchronization is acquired by driving the `rx_syncstatus` signal high. The rate matcher deletes or inserts both symbols (/K28.5/ and /D16.2/) of the /I2/ ordered-sets even if it requires deleting only one symbol to prevent the rate match FIFO from overflowing or under running. It can insert or delete as many /I2/ ordered sets as necessary to perform the rate match operation.

Two flags, `rx_rmfifodatadeleted` and `rx_rmfifodatainserted`, indicate rate match FIFO deletion and insertion events, respectively, are forwarded to the FPGA fabric. Both the `rx_rmfifodatadeleted` and `rx_rmfifodatainserted` flags are asserted for two clock cycles for each deleted and inserted /I2/ ordered-set, respectively.

Figure 1–126 shows an example of rate match FIFO deletion in the case where three symbols are required to be deleted. Because the rate match FIFO can only delete /I2/ ordered-set, it deletes two /I2/ ordered-sets (four symbols deleted).

Figure 1–126. Rate Match Deletion in GIGE Mode

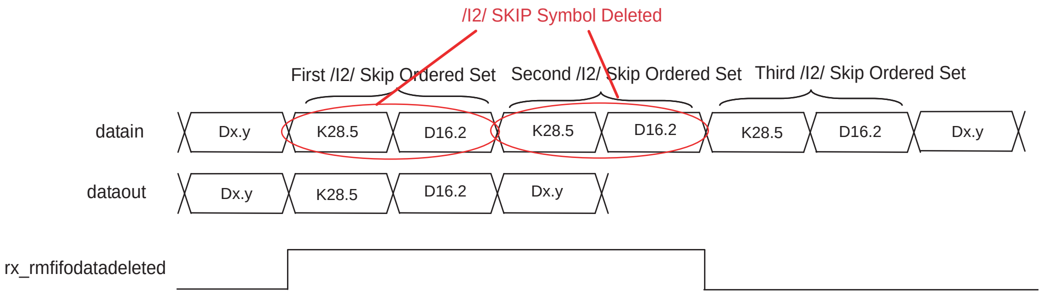
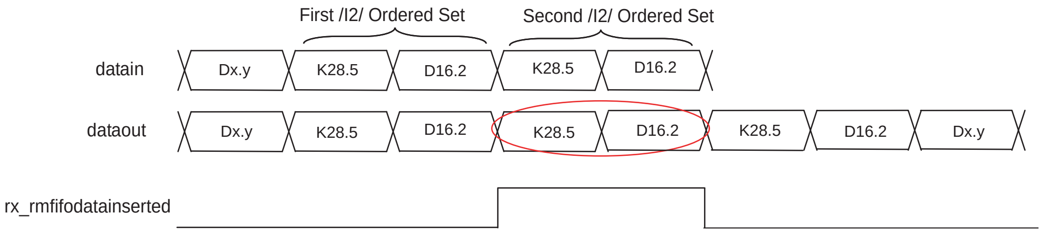


Figure 1–127 shows an example of rate match FIFO insertion in the case where one symbol is required to be inserted. Because the rate match FIFO can only delete /I2/ ordered-set, it inserts one /I2/ ordered-sets (two symbols inserted).

Figure 1–127. Rate Match Insertion in GIGE Mode



SONET/SDH Mode

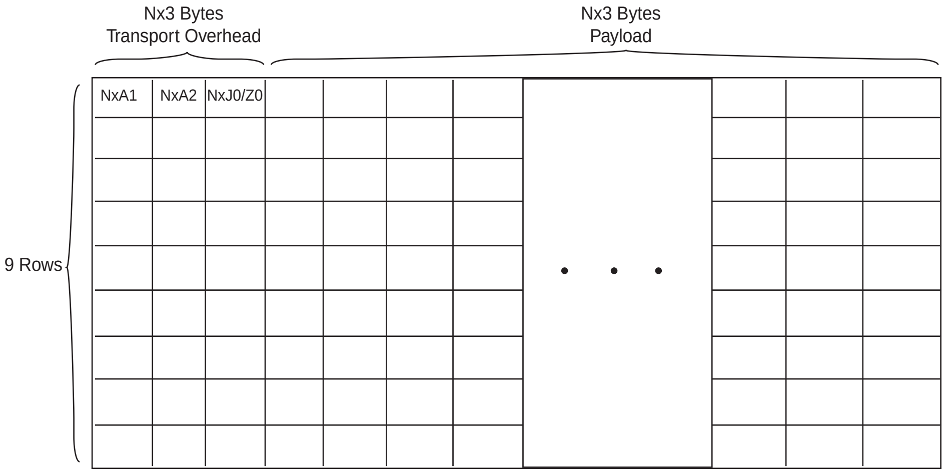
SONET/SDH is one of the most common serial-interconnect protocols used in backplanes deployed in communications and telecom applications. SONET/SDH defines various optical carrier (OC) subprotocols for carrying signals of different capacities through a synchronous optical hierarchy.

SONET/SDH Frame Structure

Base OC-1 frames are byte-interleaved to form SONET/SDH frames. For example, 12 OC-1 frames are byte-interleaved to form 1 OC-12 frame; 48 OC-1 frames are byte-interleaved to form 1 OC-48 frame and so on. SONET/SDH frame sizes are constant, with a frame transfer rate of 125 µs.

Figure 1–128 shows the SONET/SDH frame structure.

Figure 1–128. SONET/SDH Mode



Transport overhead bytes A1 and A2 are used for restoring frame boundary from the serial data stream. Frame sizes are fixed, so the A1 and A2 bytes appear within the serial data stream every 125 µs . In an OC-12 system, 12 A1 bytes are followed by 12 A2 bytes. Similarly, in an OC-48 system, 48 A1 bytes are followed by 48 A2 bytes.

In SONET/SDH systems, byte values of A1 and A2 are fixed as follows:

- A1 = 11110110 or 8'hF6
- A2 = 00101000 or 8'h28

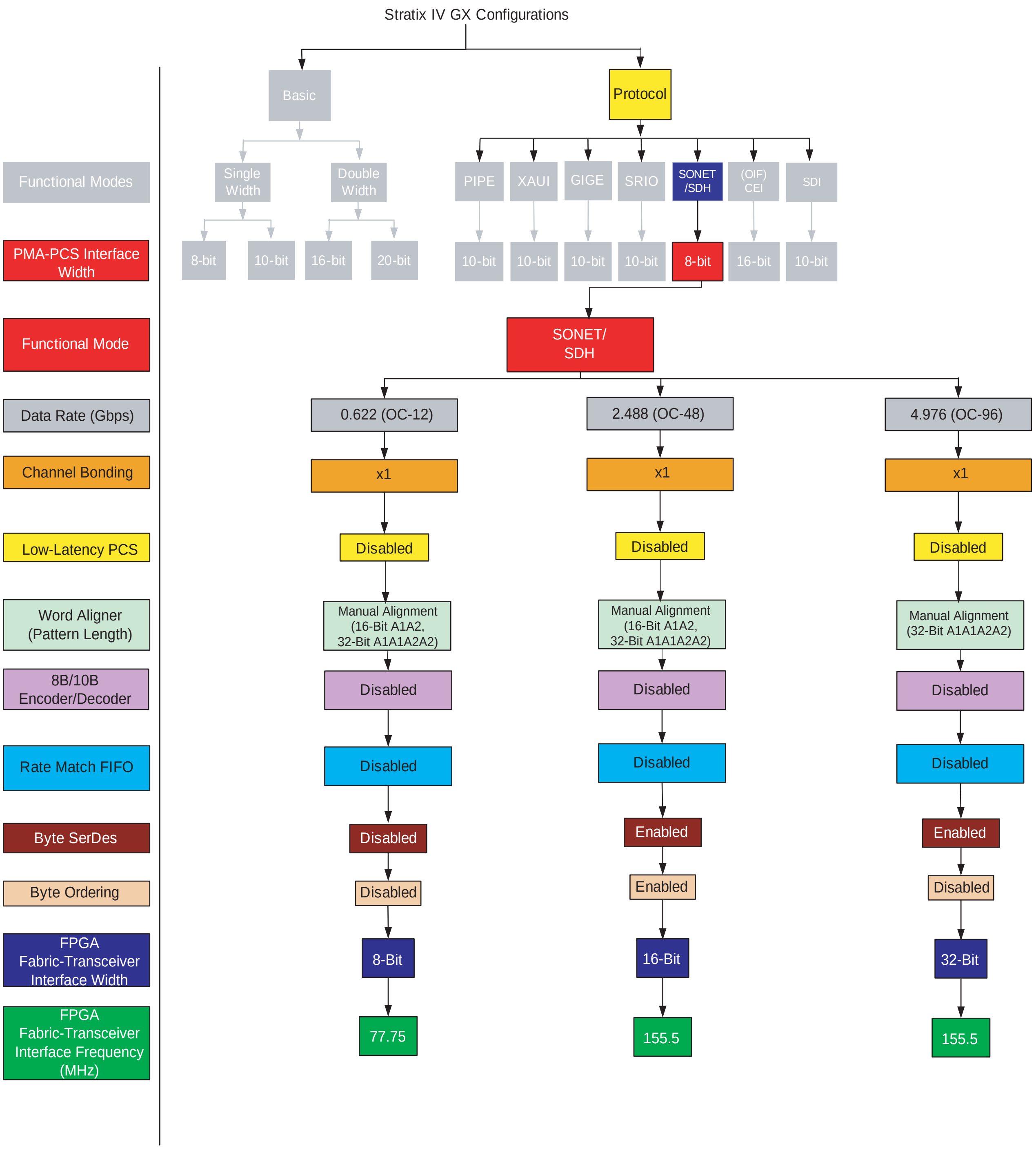
You can employ Stratix IV GX transceivers as physical layer devices in a SONET/SDH system. These transceivers provide support for SONET/SDH protocol-specific functions and electrical features; for example, alignment to A1A2 or A1A1A2A2 pattern.

Stratix IV GX transceivers are designed to support the following three SONET/SDH subprotocols:

- OC-12 at 622 Mbps with 8-bit channel width
- OC-48 at 2488.32 Mbps with 16-bit channel width
- OC-96 at 4976 Mbps with 32-bit channel width

Figure 1–129 shows SONET/SDH mode configurations supported in Stratix IV GX devices.

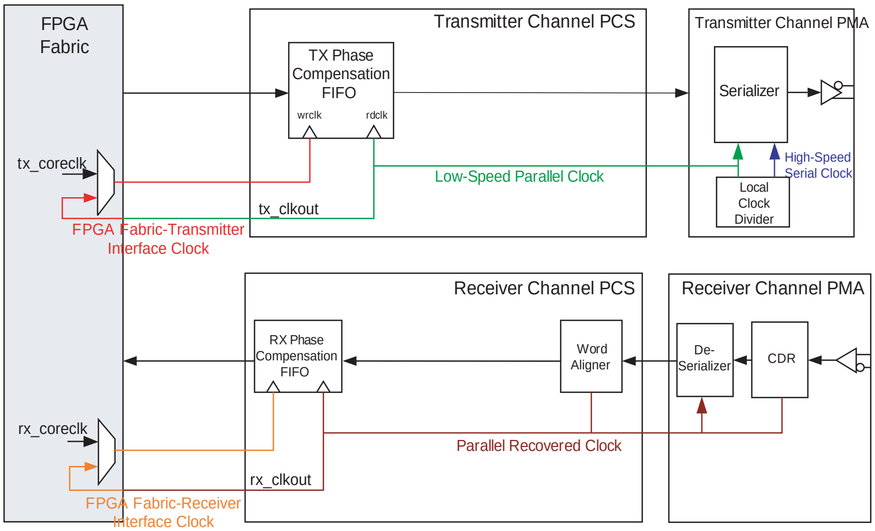
Figure 1–129. SONET/SDH Mode Configurations in Stratix IV Devices



SONET/SDH OC-12 Datapath

Figure 1–130 shows the transceiver datapath when configured in SONET/SDH OC-12 mode.

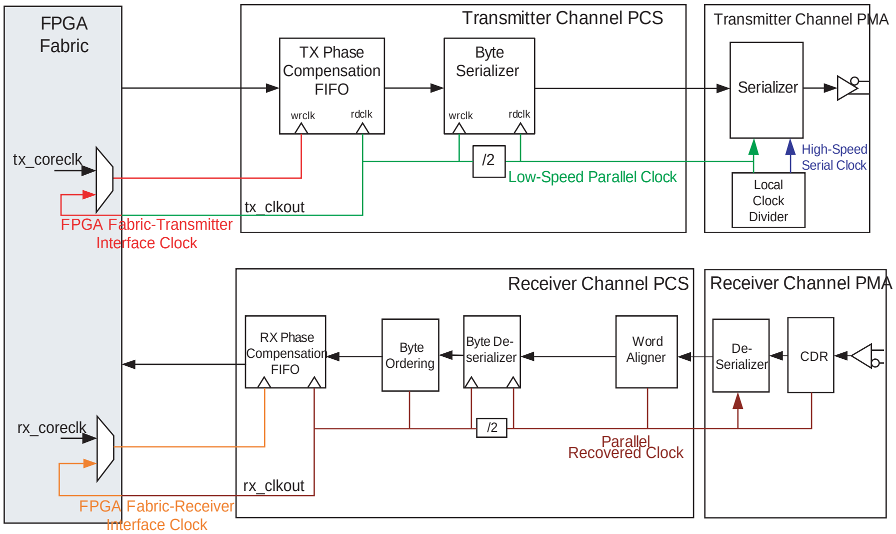
Figure 1–130. SONET/SDH OC-12 Datapath



SONET/SDH OC-48 Datapath

Figure 1–131 shows the transceiver datapath when configured in SONET/SDH OC-48 mode.

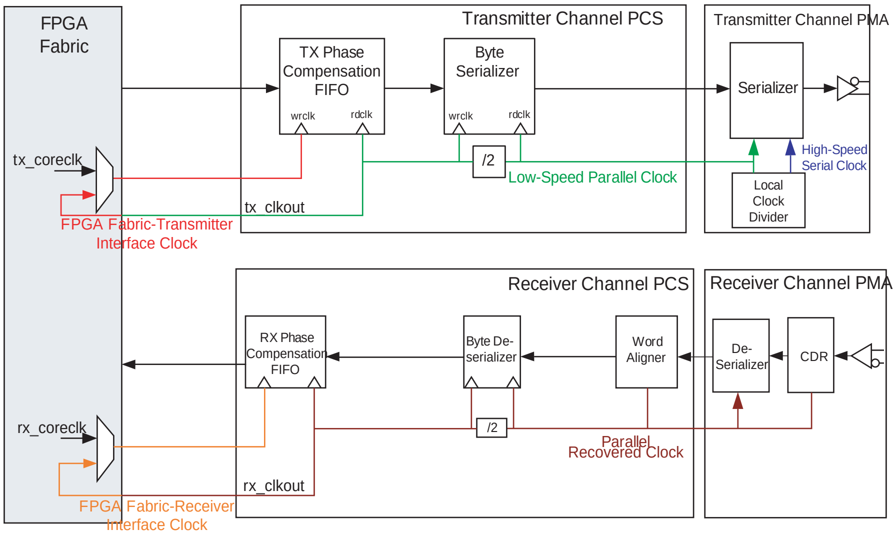
Figure 1–131. SONET/SDH OC-48 Datapath



SONET/SDH OC-96 Datapath

Figure 1–132 shows the transceiver datapath when configured in SONET/SDH OC-96 mode.

Figure 1–132. SONET/SDH OC-96 Datapath



SONET/SDH Transmission Bit Order

Unlike Ethernet, where the least significant bit of the parallel data byte is transferred first, SONET/SDH requires the most significant bit to be transferred first and the least significant bit to be transferred last. To facilitate the MSB-to-LSB transfer, you must enable the following options in the ALTGX MegaWizard Plug-In Manager:

- Flip transmitter input data bits
- Flip receiver output data bits

Depending on whether data bytes are transferred MSB-to-LSB or LSB-to-MSB, you must select the appropriate word aligner settings in the ALTGX MegaWizard Plug-In Manager. Table 1–49 lists the correct word aligner settings for each bit transmission order.

Word Alignment

The word aligner in SONET/SDH OC-12, OC-48, and OC-96 modes is configured in manual alignment mode as described in "Word Aligner in Single-Width Mode with 8-Bit PMA-PCS Interface Modes" on page 1–65.

In OC-12 and OC-48 configurations, you can configure the word aligner to either align to a 16-bit A1A2 pattern or a 32-bit A1A1A2A2 pattern. This is controlled by the `rx_a1a2size` input port to the transceiver. A low level on the `rx_a1a2size` port configures the word aligner to align to a 16-bit A1A2 pattern; a high level on the `rx_a1a2size` port configures the word aligner to align to a 32-bit A1A1A2A2 pattern.

In OC-96 configuration, the word aligner is only allowed to align to a A1A1A2A2 pattern, so input port `rx_ala2size` is unavailable. Barring this difference, the OC-96 word alignment operation is similar to that of the OC-12 and OC-48 configurations.

You can configure the word aligner to flip the alignment pattern bits programmed in the wizard and compare them with the incoming data for alignment. This feature offers flexibility to the SONET backplane system for either a MSBit-to-LSBit or LSBit-to-MSBit data transfer. Table 1–49 lists word alignment patterns that you must program in the ALTGX MegaWizard Plug-In Manager based on the bit-transmission order and the word aligner bit-flip option.

Table 1–49. Word Aligner Settings

Serial Bit Transmission Order	Word Alignment Bit Flip	Word Alignment Pattern
MSBit-to-LSBit	On	1111011000101000 (16'hF628)
MSBit-to-LSBit	Off	0001010001101111 (16'h146F)
LSBit-to-MSBit	Off	0010100011110110 (16'h28F6)

The behavior of the SONET/SDH word aligner control and status signals along with an operational timing diagram are explained in "Word Aligner in Single-Width Mode with 8-Bit PMA-PCS Interface Modes" on page 1–65.

OC-48 and OC-96 Byte Serializer and Deserializer

The OC-48 and OC-96 transceiver datapath includes the byte serializer and deserializer to allow the PLD interface to run at a lower speed. The OC-12 configuration does not use the byte serializer and deserializer blocks.

The byte serializer and deserializer blocks are explained in "Byte Serializer" on page 1–31 and "Byte Deserializer" on page 1–99, respectively.

The OC-48 byte serializer converts 16-bit data words from the FPGA fabric and translates the 16-bit data words into two 8-bit data bytes at twice the rate. The OC-48 byte deserializer takes in two consecutive 8-bit data bytes and translates them into a 16-bit data word to the FPGA fabric at half the rate.

The OC-96 byte serializer converts 32-bit data words from the FPGA fabric and translates them into two 16-bit data words at twice the rate. The OC-96 byte deserializer takes in two consecutive 16-bit data words and translates them into a 32-bit data word to the FPGA fabric at half the rate.

OC-48 Byte Ordering

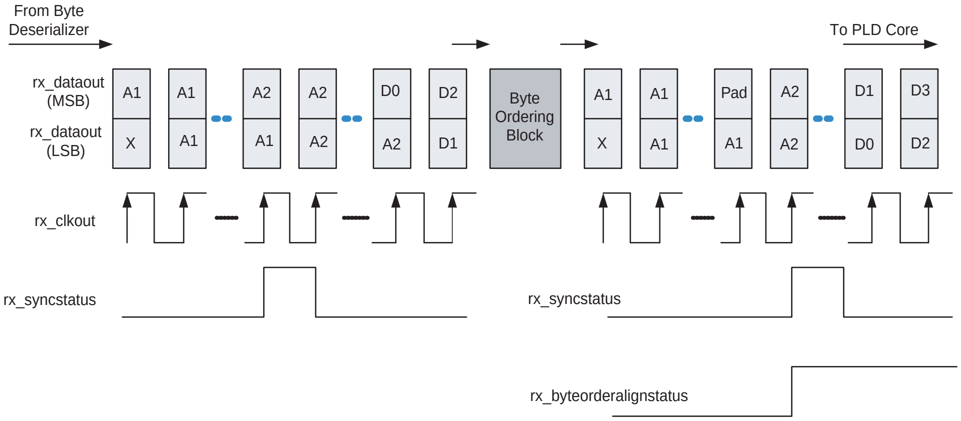
Because of byte deserialization, the most significant byte of a word might appear at the `rx_dataout` port along with the least significant byte of the next word.

In an OC-48 configuration, the byte ordering block is built into the datapath and can be leveraged to perform byte ordering. Byte ordering in an OC-48 configuration is automatic, as explained in "Word-Alignment-Based Byte Ordering" on page 1–104.

In automatic mode, the byte ordering block is triggered by the rising edge of the `rx_syncstatus` signal. As soon as the byte ordering block sees the rising edge of the `rx_syncstatus` signal, it compares the least significant byte coming out of the byte deserializer with the A2 byte of the A1A2 alignment pattern. If the least significant byte coming out of the byte deserializer does not match the A2 byte set in the ALTGX MegaWizard Plug-In Manager, the byte ordering block inserts a PAD character, as seen in Figure 1–133. Insertion of this PAD character enables the byte ordering block to restore the correct byte order.

☞ The PAD character is defaulted to the A1 byte of the A1A2 alignment pattern.

Figure 1–133. OC-48 Byte Ordering in Automatic Mode



SDI Mode

The Society of Motion Picture and Television Engineers (SMPTE) defines various SDI standards for transmission of uncompressed video.

The following three SMPTE standards are popular in video broadcasting applications:

- SMPTE 259M standard—more popularly known as the standard-definition (SD) SDI, is defined to carry video data at 270 Mbps.
- SMPTE 292M standard—more popularly known as the high-definition (HD) SDI, is defined to carry video data at either 1485 Mbps or 1483.5 Mbps.
- SMPTE 424M standard—more popularly known as the third-generation (3G) SDI, is defined to carry video data at either 2970 Mbps or 2967 Mbps.

You can configure Stratix IV GX transceivers in HD-SDI or 3G-SDI configuration using the ALTGX MegaWizard Plug-In Manager.

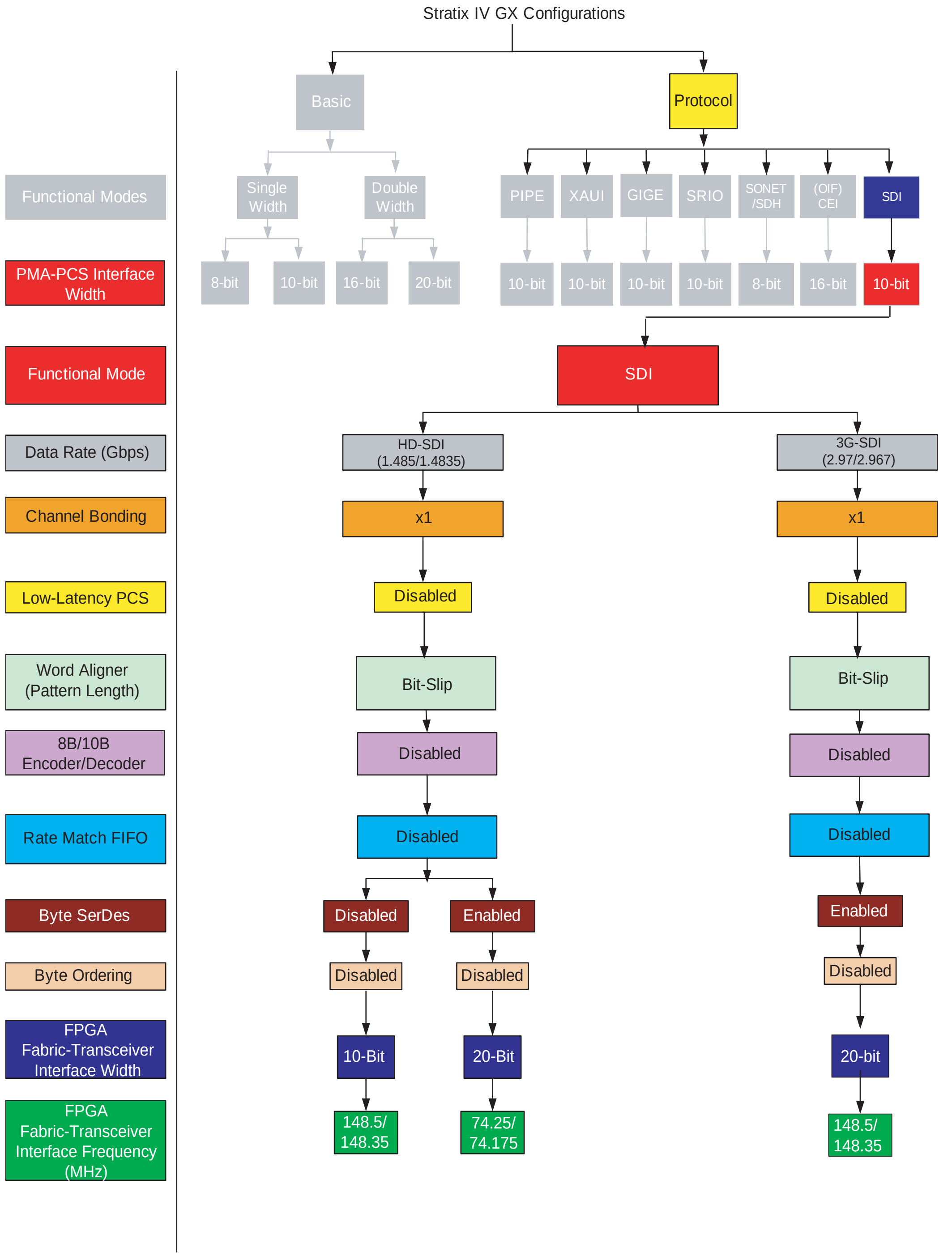
Table 1–50 shows ALTGX configurations supported by the Stratix IV GX transceivers in SDI mode.

Table 1–50. ALTGX Configurations in SDI Mode

Configuration	Data Rate (Mbps)	refclk Frequencies (MHz)	FPGA Fabric-Transceiver Interface Width
HD	1485	74.25, 148.5	10-bit, 20-bit
	1483.5	74.175, 148.35	10-bit, 20-bit
3G	2970	148.5, 297	Only 20-bit interface allowed in 3G
	2967	148.35, 296.7	Only 20-bit interface allowed in 3G

Figure 1–134 shows SDI mode configurations supported in Stratix IV GX devices.

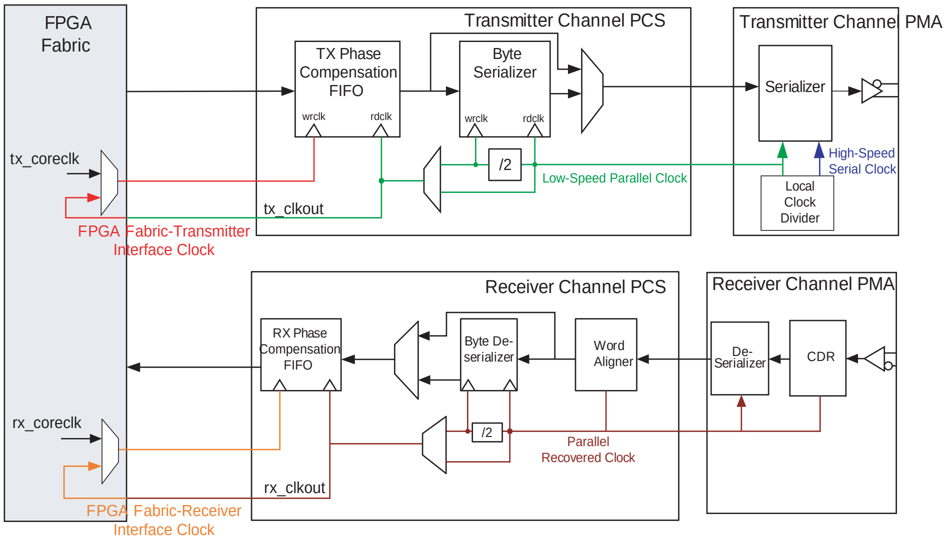
Figure 1–134. SDI Mode



SDI Mode Datapath

Figure 1–135 shows the transceiver datapath when configured in SDI mode.

Figure 1–135. SDI Mode Datapath



Transmitter Datapath

The transmitter datapath, in HD-SDI configuration with 10-bit wide FPGA fabric-transceiver interface, consists of the transmitter phase compensation FIFO and the 10:1 serializer. The transmitter datapath, in HD-SDI and 3G-SDI configurations with 20-bit wide FPGA fabric-transceiver interface, also includes the byte serializer.

☞ In SDI mode, the transmitter is purely a parallel-to-serial converter. SDI transmitter functions, such as scrambling and cyclic redundancy check (CRC) code generation, must be implemented in the FPGA logic array.

Receiver Datapath

In the 10-bit channel width SDI configuration, the receiver datapath is comprised of the clock recovery unit (CRU), 1:10 deserializer, word aligner in bit-slip mode, and receiver phase compensation FIFO. In the 20-bit channel width SDI configuration, the receiver datapath also includes the byte deserializer.

☞ SDI receiver functions, such as de-scrambling, framing, and CRC checker, must be implemented in the FPGA logic array.

Receiver Word Alignment and Framing

In SDI systems, the word aligner in the receiver datapath is not useful because word alignment and framing happens after de-scrambling. Altera recommends driving the ALTGXB megafunction `rx_bitslip` signal low to avoid the word aligner from inserting bits in the received data stream.

(OIF) CEI PHY Interface Mode

The (OIF) CEI PHY interface mode is intended to support two main protocols:

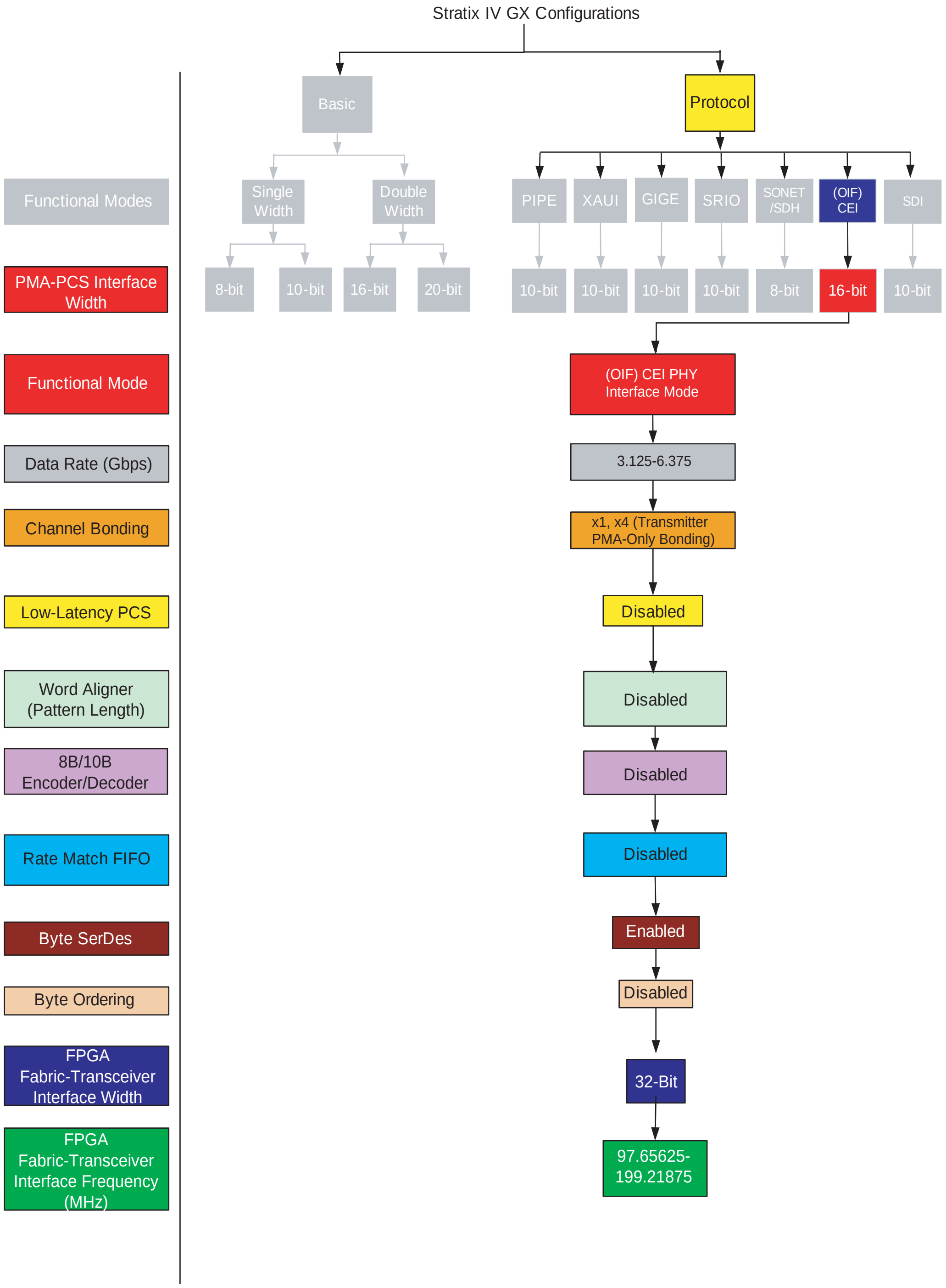
- Common Electrical I/O (CEI-6G) protocol defined by the Optical Internetworking Forum (OIF) at data rates between 4.976 Gbps and 6.375 Gbps

- Interlaken protocol at data rates between 3.125 Gbps and 6.375 Gbps

Stratix IV GX transceivers support a data rate between 3.125 Gbps and 6.375 Gbps in (OIF) CEI PHY interface mode.

Figure 1–136 shows (OIF) CEI PHY interface mode configurations supported in Stratix IV GX devices.

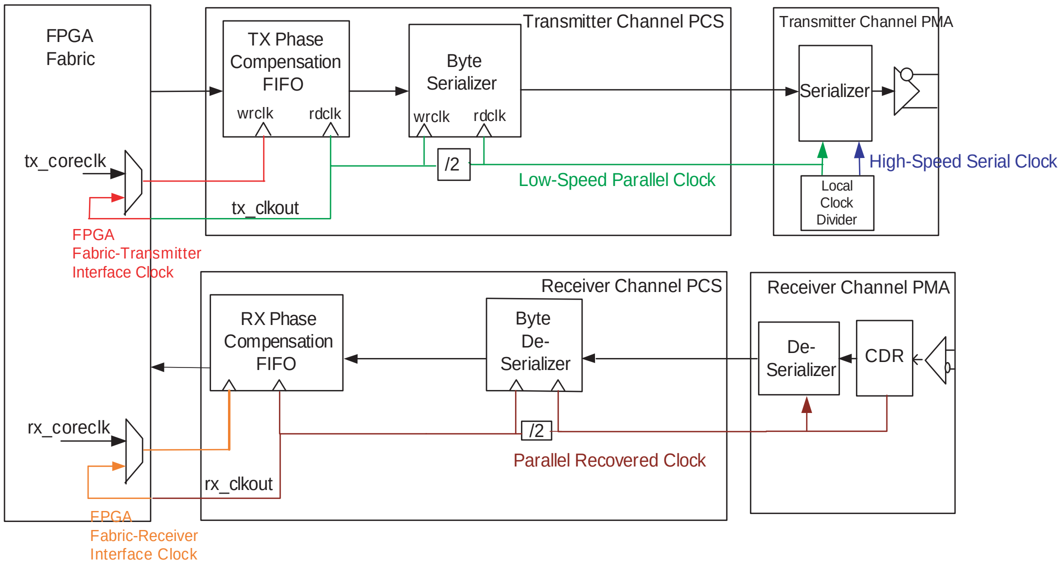
Figure 1–136. (OIF) CEI PHY Interface Mode



(OIF) CEI PHY Interface Mode Datapath

Figure 1–137 shows the ALTGX megafunction transceiver datapath when configured in (OIF) CEI PHY interface mode.

Figure 1–137. (OIF) CEI PHY Interface Mode Datapath



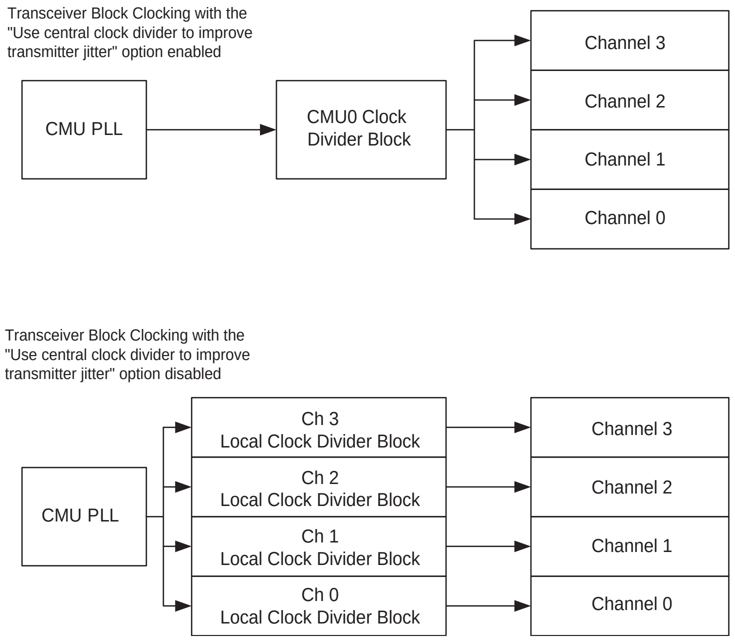
(OIF) CEI PHY Interface Mode Clocking

For improved transmitter jitter performance, the ALTGX MegaWizard Plug-In Manager provides the **Use central clock divider to improve transmitter jitter** option. If you select this option, the high-speed serial clock generated by the `CMU0` clock divider block clocks all four transceiver channels within the same transceiver block. Otherwise, the high-speed serial clock generated by the local clock divider in each channel clocks the respective channel.

☞ Unlike PIPE 4, XAUI or Basic ×4 mode, the transmitter PCS is not bonded in the (OIF) CEI PHY interface mode with the low-jitter option selected.

Figure 1–138 shows transceiver clocking in (OIF) CEI PHY interface mode with and without the improved transmitter jitter option enabled.

Figure 1–138. Transceiver Clocking in (OIF) CEI PHY Interface Mode


Transceiver Block Clocking with the "Use central clock divider to improve transmitter jitter" option enabled
CMU PLL
CMU0 Clock Divider Block
Channel 3
Channel 2
Channel 1
Channel 0
Transceiver Block Clocking with the "Use central clock divider to improve transmitter jitter" option disabled
CMU PLL
Ch 3 Local Clock Divider Block
Ch 2 Local Clock Divider Block
Ch 1 Local Clock Divider Block
Ch 0 Local Clock Divider Block
Channel 3
Channel 2
Channel 1
Channel 0

Transceiver Placement Limitations with Use Central Clock Divider to Improve Transmitter Jitter Option Enabled

If one or more channels in a transceiver block are configured to (OIF) CEI PHY interface mode with the improved jitter clocking option enabled, the remaining channels in that transceiver block must either be configured in (OIF) CEI PHY interface mode with this option enabled or must be unused. All used channels within a transceiver block configured in (OIF) CEI PHY interface mode with improved jitter clocking option enabled must also run at the same data rate.

Figure 1–139 and 1–169 show two examples each of legal and illegal transceiver placements with respect to the improved jitter clocking option in (OIF) CEI PHY interface mode.

Figure 1–139. Examples of Legal Transceiver Placement in (OIF) CEI PHY Interface Mode

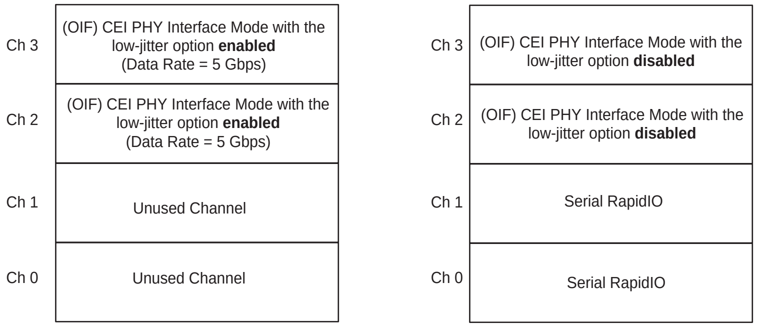
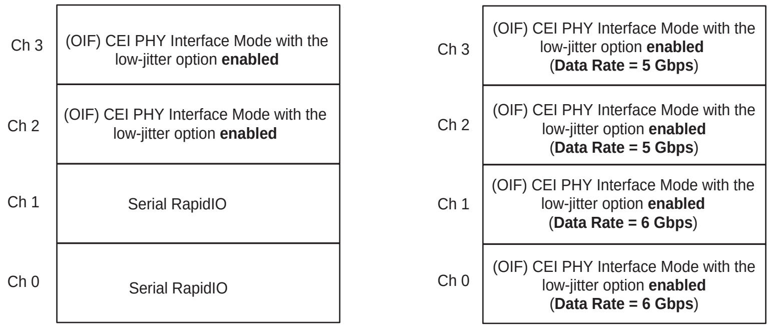

Ch 3
(OIF) CEI PHY Interface Mode with the low-jitter option **enabled** (Data Rate = 5 Gbps)
Ch 2
(OIF) CEI PHY Interface Mode with the low-jitter option **enabled** (Data Rate = 5 Gbps)
Ch 1
Unused Channel
Ch 0
Unused Channel
Ch 3
(OIF) CEI PHY Interface Mode with the low-jitter option **disabled**
Ch 2
(OIF) CEI PHY Interface Mode with the low-jitter option **disabled**
Ch 1
Serial RapidIO
Ch 0
Serial RapidIO

Figure 1–140. Examples of Illegal Transceiver Placement in (OIF) CEI PHY Interface Mode


Ch 3
(OIF) CEI PHY Interface Mode with the low-jitter option **enabled**
Ch 2
(OIF) CEI PHY Interface Mode with the low-jitter option **enabled**
Ch 1
Serial RapidIO
Ch 0
Serial RapidIO
Ch 3
(OIF) CEI PHY Interface Mode with the low-jitter option **enabled** (**Data Rate = 5 Gbps**)
Ch 2
(OIF) CEI PHY Interface Mode with the low-jitter option **enabled** (**Data Rate = 5 Gbps**)
Ch 1
(OIF) CEI PHY Interface Mode with the low-jitter option **enabled** (**Data Rate = 6 Gbps**)
Ch 0
(OIF) CEI PHY Interface Mode with the low-jitter option **enabled** (**Data Rate = 6 Gbps**)

Serial RapidIO Mode

The RapidIO Trade Association defines a high-performance, packet-switched interconnect standard to pass data and control information between microprocessors, digital signal, communications, and network processors, system memories, and peripheral devices.

Serial RapidIO physical layer specification defines three line rates:

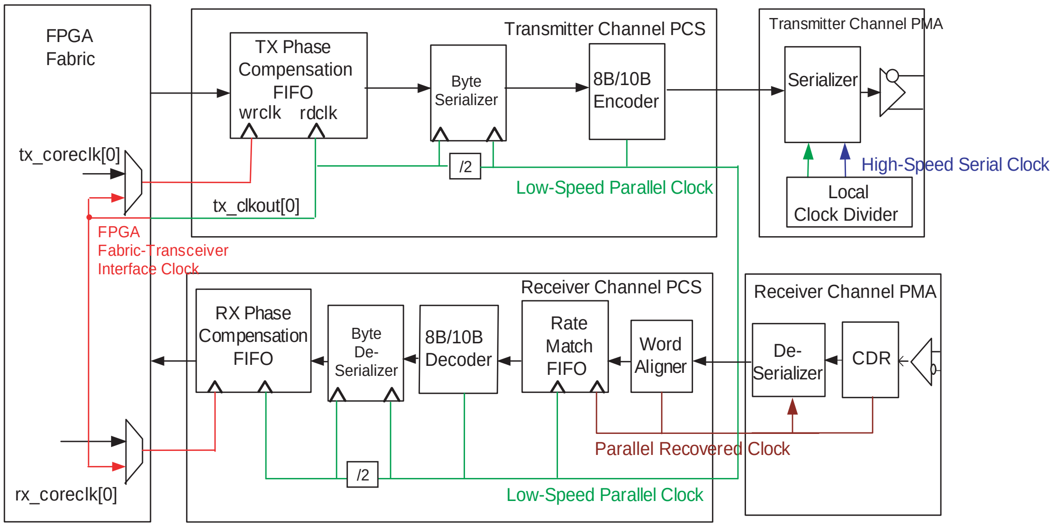
- 1.25 Gbps
- 2.5 Gbps
- 3.125 Gbps

It also defines two link widths—single-lane (1×) and bonded four-lane (4×) at each line rate.

Stratix IV GX transceivers support only single-lane (1×) configuration at all three line rates. Four 1× channels configured in Serial RapidIO mode can be instantiated to achieve a 4× Serial RapidIO link. The four transmitter channels in this 4× Serial RapidIO link are not bonded. The four receiver channels in this 4× Serial RapidIO link do not have lane alignment or deskew capability.

Figure 1–141 shows the ALTGX transceiver data path when configured in Serial RapidIO mode.

Figure 1–141. Serial RapidIO Mode Datapath



Stratix IV GX transceivers, when configured in Serial RapidIO functional mode, provide the following PCS and PMA functions:

- 8B/10B encoding/decoding
- Word alignment
- Lane Synchronization State Machine
- Clock recovery from the encoded data
- Serialization/deserialization

☞ Stratix IV GX transceivers do not have built-in support for other PCS functions; for example, pseudo-random idle sequence generation and lane alignment in 4× mode. Depending on your system requirements, you must implement these functions in the logic array or external circuits.

Synchronization State Machine

In Serial RapidIO mode, the ALTGX MegaWizard Plug-In Manager defaults the word alignment pattern to K28.5. The word aligner has a synchronization state machine that handles the receiver lane synchronization.

The ALTGX MegaWizard Plug-In Manager automatically defaults the synchronization state machine to indicate synchronization when the receiver receives 127 K28.5 (10'b0101111100 or 10'b1010000011) synchronization code groups without receiving an intermediate invalid code group. Once synchronized, the state machine indicates loss of synchronization when it detects three invalid code groups separated by less than 255 valid code groups or when it is reset.

Receiver synchronization is indicated on the `rx_syncstatus` port of each channel. A high on the `rx_syncstatus` port indicates that the lane is synchronized and a low indicates that it has fallen out of synchronization.

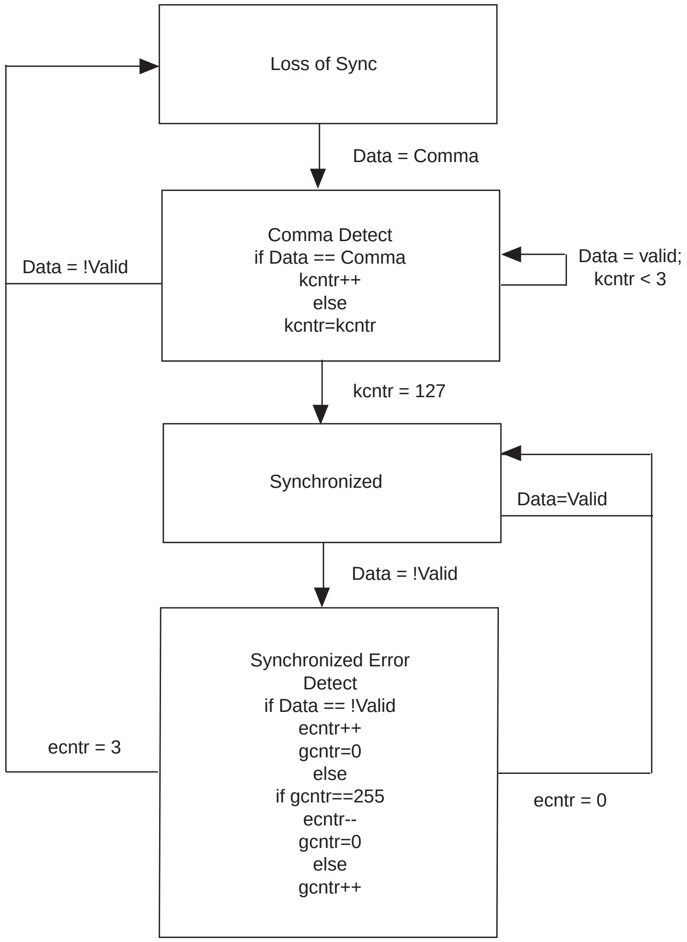
Table 1–51 lists the ALTGX megafunction synchronization state machine parameters when configured in Serial RapidIO mode.

Table 1–51. Synchronization State Machine Parameters in Serial RapidIO Mode

Parameters	Number
Number of valid K28.5 code groups received to achieve synchronization.	127
Number of errors received to lose synchronization.	3
Number of continuous good code groups received to reduce the error count by one.	255

Figure 1–142 gives a conceptual view of the synchronization state machine implemented in Serial RapidIO functional mode.

Figure 1–142. Synchronization State Machine in Serial RapidIO Mode



Loopback Modes

Stratix IV GX devices provide various loopback options that allow you to verify the working of different functional blocks in the transceiver channel. The available loopback options are:

- Serial loopback—available in all functional modes except PCI Express (PIPE) mode
- Reverse serial loopback—available in Basic mode only

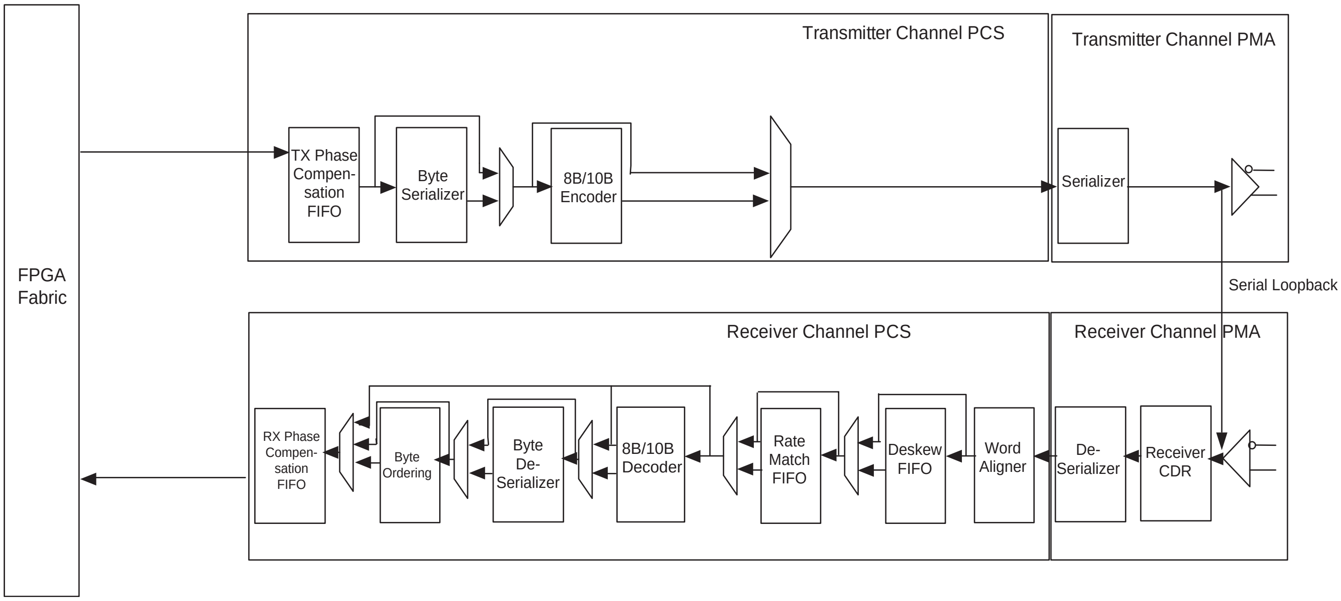
- Reverse serial pre-CDR loopback—available in Basic mode only
- PCI Express (PIPE) reverse parallel loopback—supported in PCI Express (PIPE) protocol only

Serial Loopback

The serial loopback option is available for all functional modes except PCI Express (PIPE) mode. Figure 1–143 shows the datapath for serial loopback. The data from the FPGA fabric passes through the transmitter channel and gets looped back to the receiver channel bypassing the receiver buffer. The received data is available to the FPGA logic for verification. Using this option, you can check the working for all enabled PCS and PMA functional blocks in the transmitter and receiver channel. When you enable the serial loopback option, the ALTGX MegaWizard Plug-In Manager provides the `rx_seriallpbken` port to dynamically enable serial loopback on a channel-by-channel basis. Set the `rx_seriallpbken` signal to logic high to enable serial loopback.

When serial loopback is enabled, the transmitter channel sends the data to both the `tx_dataout` output port and to the receiver channel. The differential output voltage on the `tx_dataout` ports is based on the selected VOD settings. The looped back data is received by the receiver CDR and is retimed through different clock domains. You must provide an alignment pattern for the word aligner to enable the receiver channel to retrieve the byte boundary.

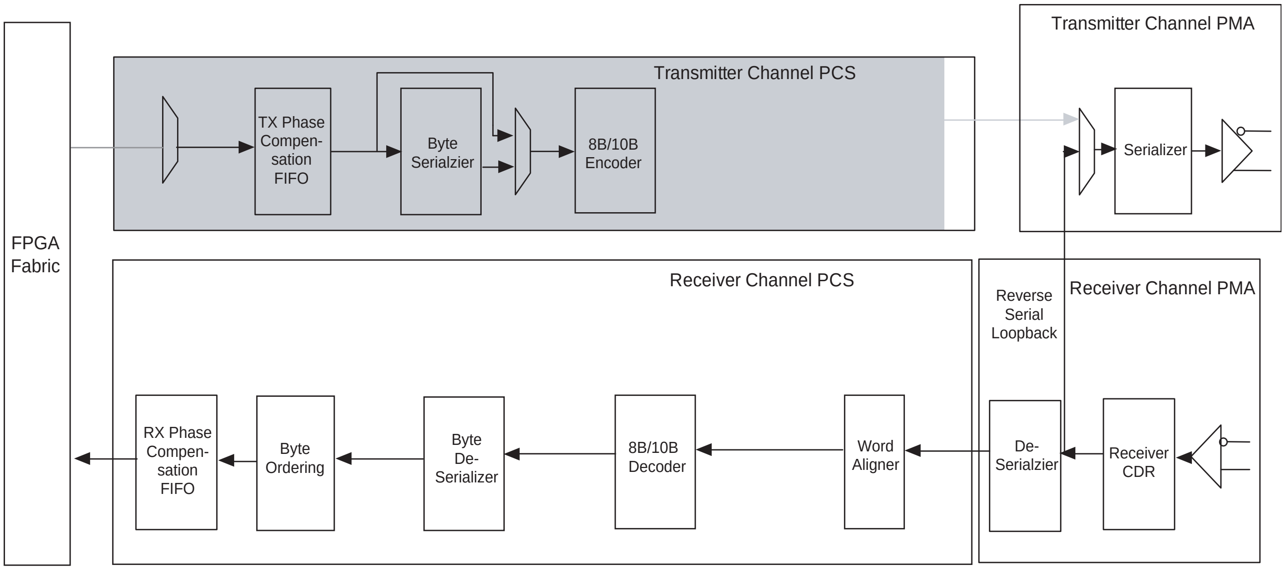
Figure 1–143. Serial Loopback Datapath



Reverse Serial Loopback

Reverse serial loopback is available as a subprotocol under Basic functional mode. In reverse serial loopback mode, the data is received through the `rx_datain` port, retimed through the receiver CDR, and sent out to the `tx_dataout` port. The received data is also available to the FPGA logic. Figure 1–144 shows the transceiver channel datapath for reverse serial loopback mode. Note that the active block of the transmitter channel is only the transmitter buffer. You can change the output differential voltage on the transmitter buffer through the ALTGX MegaWizard Plug-In Manager. You cannot alter the pre-emphasis settings for the transmitter buffer. Reverse serial loopback is often implemented when using a bit error rate tester (BERT) on the upstream transmitter.

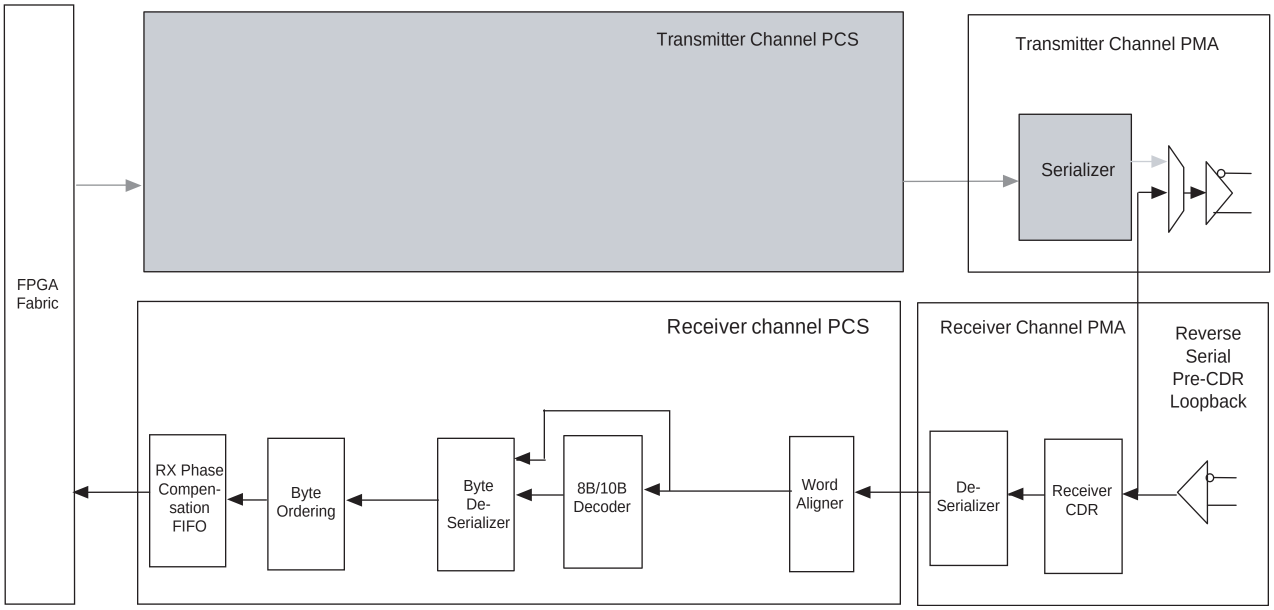
Figure 1–144. Reverse Serial Loopback Datapath (Grayed-Out Blocks are Not Active in this Mode)



Reverse Serial Pre-CDR Loopback

The reverse serial pre-CDR loopback is available as a subprotocol under Basic functional mode. In reverse serial pre-CDR loopback, the data received through the `rx_datain` port is looped back to the `tx_dataout` port **BEFORE** the receiver CDR. The received data is also available to the FPGA logic. Figure 1–145 shows the transceiver channel datapath for reverse serial pre-CDR loopback mode. Note that the active block of the transmitter channel is only the transmitter buffer. You can change the output differential voltage on the transmitter buffer through the ALTGX MegaWizard Plug-In Manager. You cannot change the pre-emphasis settings for the transmitter buffer.

Figure 1–145. Reverse Serial Pre-CDR Loopback Datapath



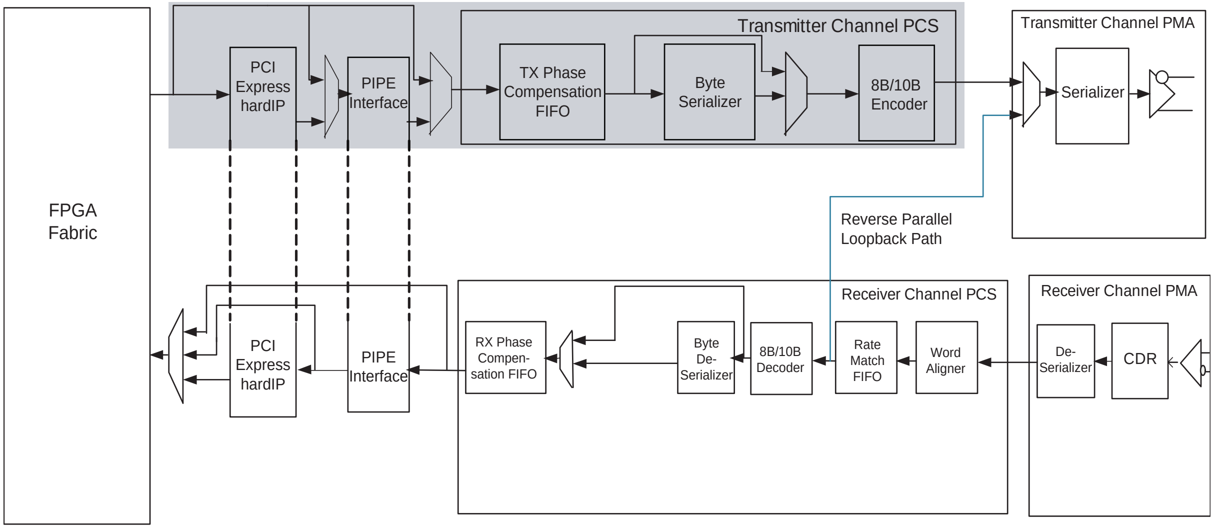
PCI Express (PIPE) Reverse Parallel Loopback

PCI Express (PIPE) reverse parallel loopback is only available in PCI Express (PIPE) functional mode for Gen1 and Gen2 data rates. As shown in Figure 1–146, the received serial data passes through the receiver CDR, deserializer, word aligner, and rate matching FIFO buffer. It is then looped back to the transmitter serializer and transmitted out through the `tx_dataout` port. The received data is also available to the FPGA fabric through the `rx_dataout` port. This loopback mode is complaint with the PCI Express (PIPE) specification 2.0. To enable this loopback mode, assert the `tx_detectrxloopback` port.

This is the only loopback option supported in PCI Express (PIPE) functional mode.

In Figure 1–146, the grayed areas show the inactive paths when the PCI Express (PIPE) reverse parallel loop back mode is enabled.

Figure 1–146. PCI Express (PIPE) Reverse Parallel Loopback Mode Datapath (Grayed-Out Blocks are Not Active in this Mode)



Calibration Blocks

Stratix IV GX devices contain calibration circuits that calibrate the OCT resistors and the analog portions of the transceiver blocks to ensure that the functionality is independent of process, voltage, or temperature variations.

Calibration Block Location

Figure 1–147 shows the location and number of calibration blocks available for different transceiver block device families. In Figure 1–147 through 1–177, the calibration block R0 and L0 refer to the calibration blocks on the right and left side, respectively.

Figure 1–147. Calibration Blocks

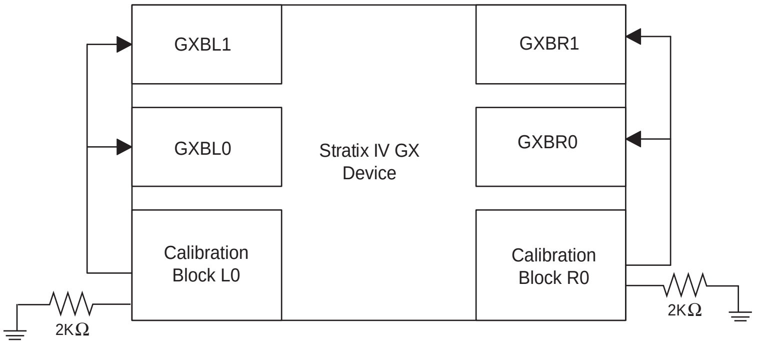


Figure 1–148 shows Stratix IV GX device families that have three transceiver blocks each on the left and right side.

Figure 1–148. Calibration Block Locations in Stratix IV GX Device Families with Three Tranceiver Blocks (on Each Side)

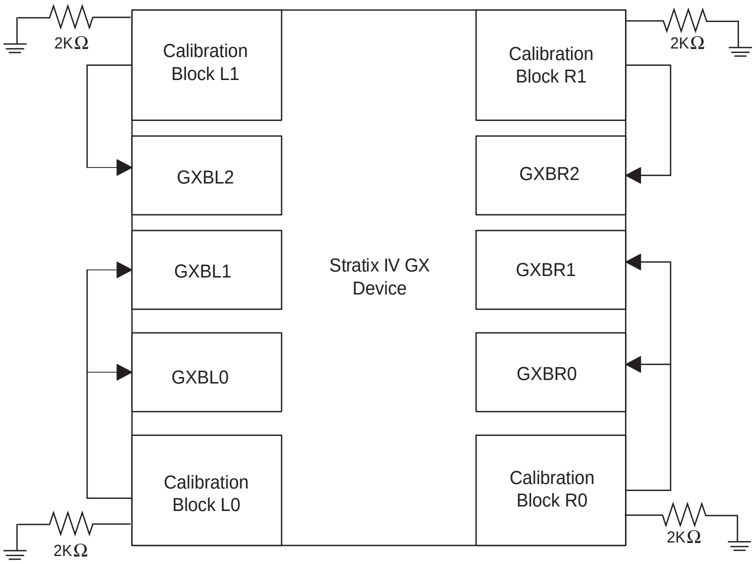


Figure 1–149 shows Stratix IV GX device families that have three transceiver blocks each on the left and right side.

Figure 1–149. Calibration Three-Transceiver Blocks

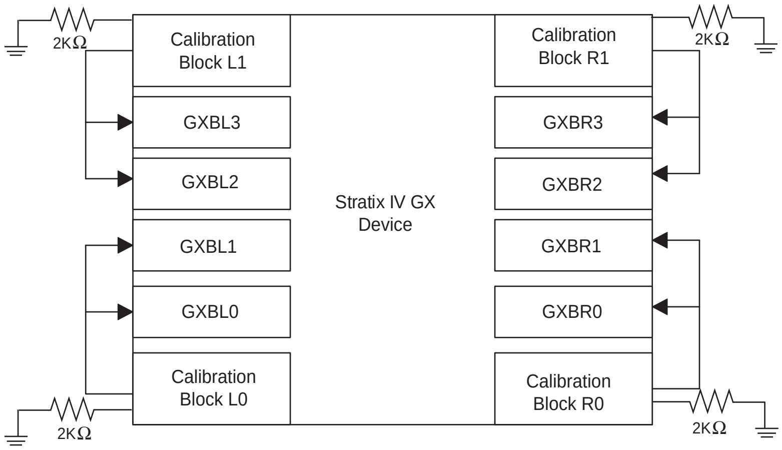
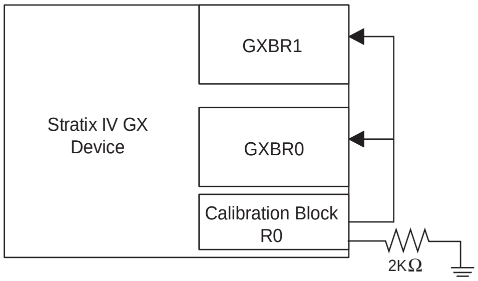


Figure 1–150 shows Stratix IV GX device families that have two transceiver blocks only on the right side of the device.

Figure 1–150. Calibration Two Transceiver Blocks, Right Side Only



The Quartus II software automatically selects the appropriate calibration block based on the assignment of the transceiver `tx_dataout` and `rx_datain` pins.

Calibration

The calibration block internally generates a constant internal reference voltage, independent of process, voltage or temperature variations. It uses the internal reference voltage and external reference resistor (you must connect the resistor to the RREF pin) to generate constant reference currents. These reference currents are used by the analog block calibration circuit to calibrate the transceiver blocks.

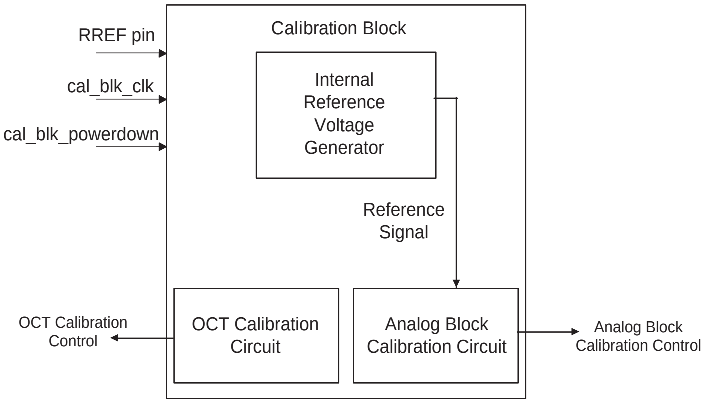
The OCT calibration circuit calibrates the OCT resistors present in the transceiver channels. You can enable the OCT resistors in the transceiver channels through the ALTGX MegaWizard Plug-In Manager.

You must connect a separate 2 kΩ (tolerance max ± 1%) external resistor on each RREF pin in the Stratix IV GX device to ground. To ensure proper operation of the calibration block, the RREF resistor connection in the board must be free from any external noise.

Input Signals to the Calibration Block

Figure 1–151 shows the required inputs to the calibration block. The ALTGX MegaWizard Plug-In Manager provides the `cal_blk_clk` and `cal_blk_powerdown` ports to control the calibration block.

Figure 1–151. Input Signals to the Calibration Blocks



- `cal_blk_clk`—you must use the `cal_blk_clk` port to provide input clock to the calibration clock. The frequency of `cal_blk_clk` must be within 10 MHz to 125 MHz (this range is preliminary. Final values will be available upon characterization). You can use dedicated clock routes such as the global or regional clock. If you do not have suitable input reference clock or dedicated clock routing resources available, use divide-down logic from the FPGA fabric to generate a slow clock and use local clocking routing. Drive the `cal_blk_clk` port of all ALTGX instances that are associated with the same calibration block from the same input pin or logic.
- `cal_blk_powerdown`—you can perform calibration multiple times by using the `cal_blk_powerdown` port available through the ALTGX MegaWizard Plug-In Manager. Assert this signal for approximately 500 ns (this is preliminary. Final values will be available upon characterization). Following de-assertion of `cal_blk_powerdown`, the calibration block restarts the calibration process. Drive the `cal_blk_powerdown` port of all ALTGX instances that are associated with the same calibration block from the same input pin or logic.

Referenced Documents

This chapter references the following documents:

- *Configuration, Design Security, Remote System Upgrades with Stratix IV Devices* chapter in volume 1 in the *Stratix IV Device Handbook*
- *Configuring Multiple Protocols and Data Rates* chapter in volume 2 of *Stratix IV Device Handbook*
- *PCI Express Compiler User Guide*
- *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook*
- *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*

Document Revision History

Table 1–52 shows the revision history for this document.

Table 1–52. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	Added Offset Cancellation in the Receiver Buffer and Receiver CDR to the Receiver Channel Datapath section	—
May 2008 v1.0	Initial release.	—

2. Stratix IV Transceiver Clocking

SIV52002-2.0

Introduction

This chapter provides detailed information about Stratix® IV GX transceiver clocking architecture. It is divided into following main sections:

- "CMU PLL and Receiver CDR Input Reference Clocking" on page 2–1
- "Transceiver Channel Datapath Clocking" on page 2–7
- "FPGA Fabric-Transceiver Interface Clocking" on page 2–28
- "FPGA Fabric PLLs-Transceiver PLLs Cascading" on page 2–57

CMU PLL and Receiver CDR Input Reference Clocking

Each transceiver block has:

- Two clock multiplier unit (CMU) phase-locked loops (PLLs) (`CMU0 PLL` and `CMU1 PLL`), one in each clock multiplier unit channel (`CMU Channel`)
- Four clock data recovery (CDR) units, one in each receiver channel

The CMU PLLs and the receiver CDRs require an input reference clock for their operation. The CMU PLL synthesizes the input reference clock to generate the high-speed serial clock used in the transmitter PMA. The receiver CDR uses the input reference clock as a training clock when it is in lock-to-reference (LTR) mode.

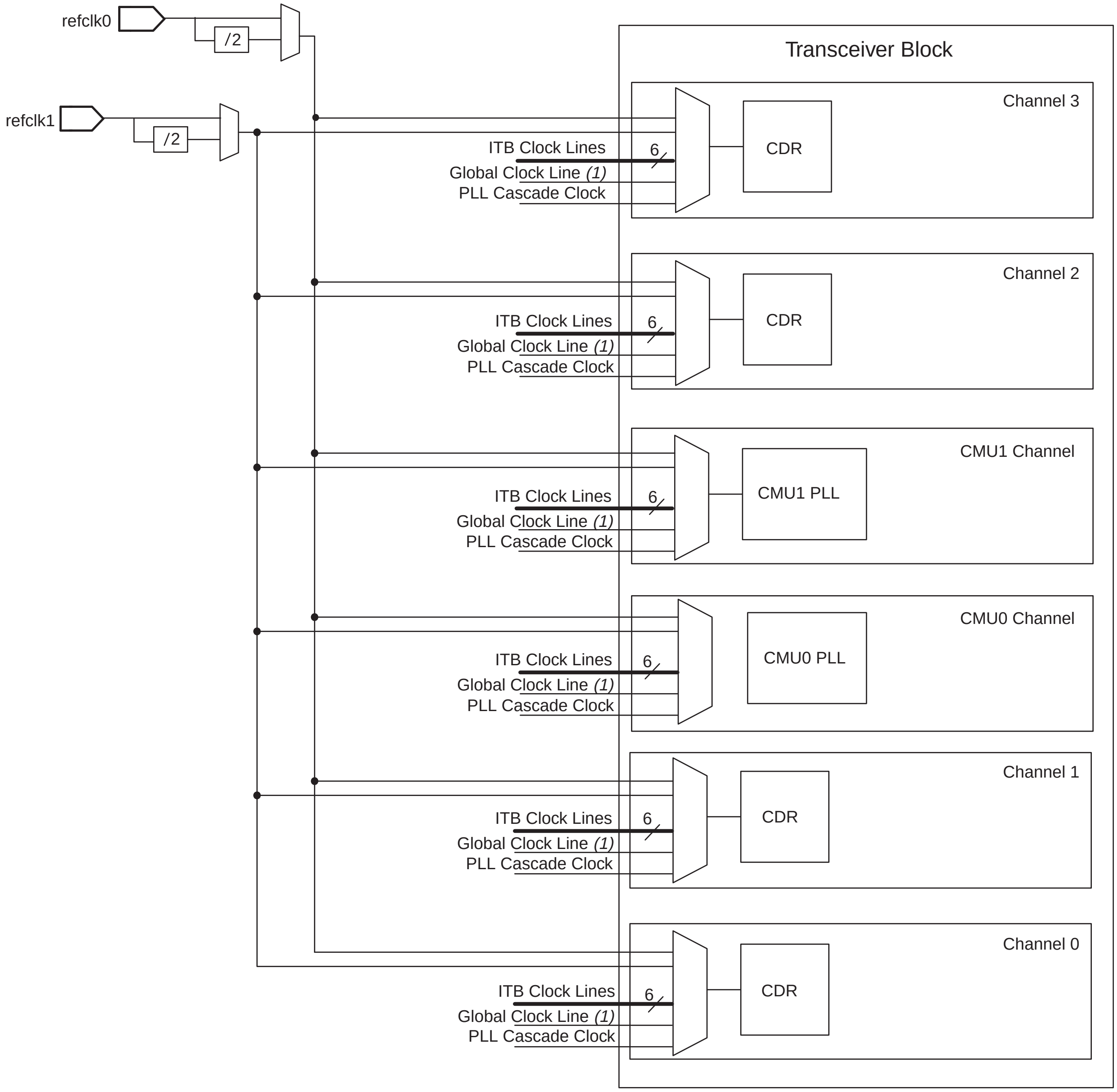
Input Reference Clock Source

The CMU PLLs and the receiver CDRs in each transceiver block can derive the input reference from one of the following sources:

- `refclk0` and `refclk1` pins of the same transceiver block
- `refclk0` and `refclk1` pins of other transceiver blocks on the same side of the device using the inter transceiver block (ITB) clock network
- Dedicated CLK input pins on the FPGA global clock network
- Clock output from the left and right PLLs in the FPGA fabric

Figure 2–1 shows the input reference clock sources for CMU PLLs and receiver CDRs within a transceiver block.

Figure 2–1. Input Reference Clock Sources in a Transceiver Block

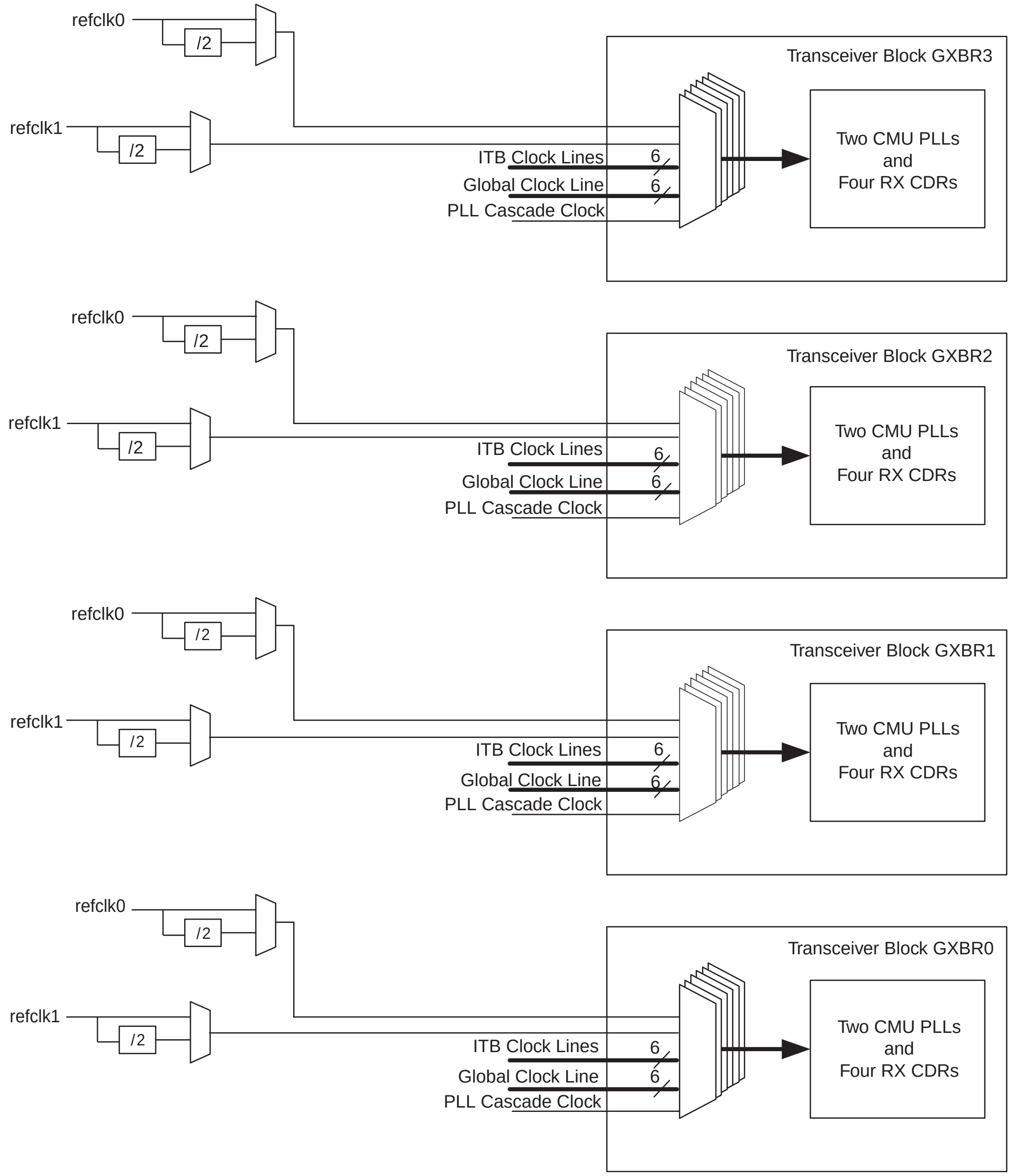


Note to Figure 2–1:

(1) One global clock line is available for each CMU PLL and receiver CDR in a transceiver block. This allows each CMU PLL and receiver CDR to derive its input reference clock from a separate FPGA CLK input pin.

Figure 2–2 shows the input reference clock sources for CMU PLLs and receiver CDRs in four transceiver blocks on the right side of the EP4SX530F45 device.

Figure 2–2. Input Reference Clock Sources Across Transceiver Blocks *(Note 1)*



Note to Figure 2–2:

(1) This figure shows input reference clock sources for four transceiver blocks located only on the right side of the EP4SGX530NF45 device.The EP4SGX530NF45 device has similar input reference clock resources available for the four transceiver blocks located on the left side of the device as well.

refclk0 and refclk1 Pins

Each transceiver block has two dedicated `refclk` pins that you can use to drive the CMU PLL and/or receiver CDR input reference clocks. Each of the two CMU PLLs and four receiver CDRs within a transceiver block can derive its input reference clock from either the `refclk0` or `refclk1` pin.

☞ The `refclk` pins provide the cleanest input reference clock path to the CMU PLLs. Altera recommends using the `refclk` pins to drive the CMU PLL input reference clock for improved transmitter output jitter performance.

Table 2–1 shows the electrical specifications for the input reference clock signal driven on the `refclk` pins.

Table 2–1. Electrical Specifications for the Input Reference Clock

Protocol	I/O Standard	Coupling	Termination
■ GIGE ■ XAUI ■ Serial RapidIO ■ SONET/SDH ■ SDI ■ (OIF) CEI PHY Interface ■ Basic	■ 1.2-V PCML ■ 1.5-V PCML ■ 3.3-V PCML ■ Differential LVPECL ■ LVDS	AC	On-chip
PCI Express (PIPE)	■ 1.2-V PCML ■ 1.5-V PCML ■ 3.3-V PCML ■ Differential LVPECL ■ LVDS	AC	On-chip
	■ HCSL *(1)*	DC	Off-chip *(2)*

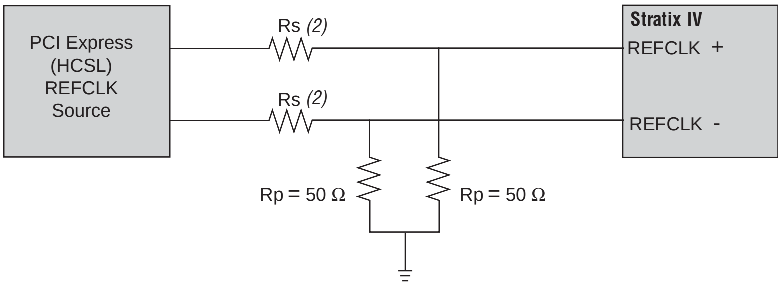
Notes to Table 2–1:

(1) In PIPE mode, you have the option of selecting the HCSL standard for the reference clock if compliance to PCI Express protocol is required. The Quartus® II software automatically selects DC coupling with external termination for the `refclk` pins signal if configured as HCSL.

(2) Refer to Figure 2–3 for an example termination scheme.

Figure 2–3 shows an example termination scheme for a reference clock signal when configured as HCSL.

Figure 2–3. Termination Scheme for a Reference Clock Signal When Configured as HCSL *(Note 1)*, *(2)*



Notes to Figure 2–3:

(1) No biasing is required if the reference clock signals are generated from a clock source that conforms to the PCI Express specification.

(2) Select resistor values as recommended by the PCI Express clock source vendor.

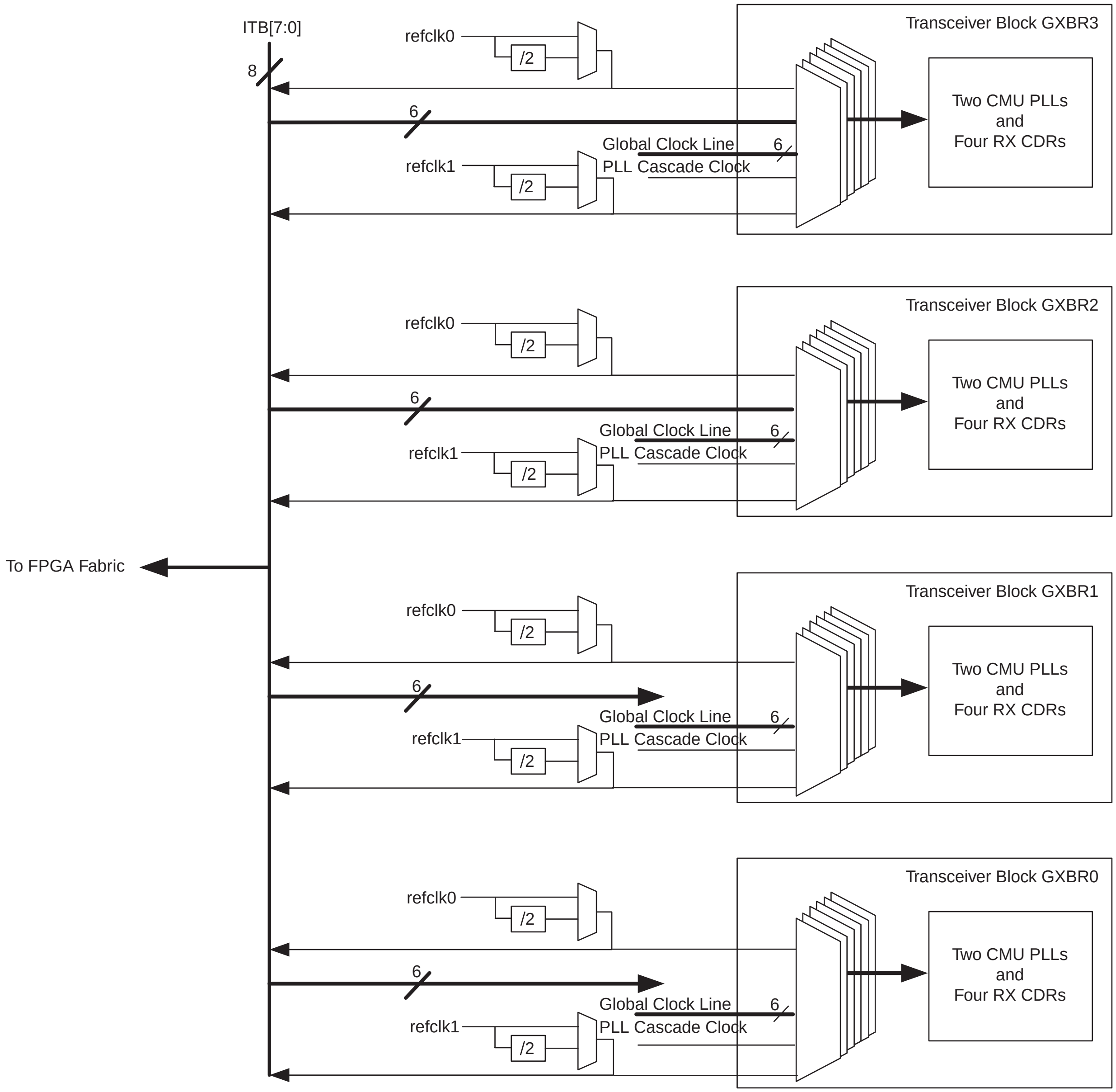
Inter-Transceiver Block (ITB) Clock Lines

The ITB clock lines provide an input reference clock path from the `refclk` pins of one transceiver block to the CMU PLLs and receiver CDRs of other transceiver blocks. In designs that have channels located in different transceiver blocks, the ITB clock lines eliminate the need to connect the on-board reference clock crystal oscillator to the `refclk` pin of each transceiver block. The ITB clock lines also drive the clock signal on the `refclk` pins to clock logic in the FPGA fabric.

Each `refclk` pin drives one ITB clock line for a total of up to eight ITB clock lines on each of the right and left sides of the device, as shown in Figure 2–4.

☞ The ITB clock lines provide input reference clock paths from the `refclk` pins of one transceiver block to the CMU PLLs and receiver CDRs of other transceiver blocks located on the same side of the device.

Figure 2–4. Inter-Transceiver Block (ITB) Clock Lines *(Note 1)*



Note for Figure 2–4:

(1) This figure shows the ITB clock lines on the right side of the EP4SGX530NF45 device. The number of ITB clock lines available in any Stratix IV GX device is equal to the number of `refclk` pins available in that device.

Dedicated CLK Input Pins on the FPGA Global Clock Network

Stratix IV GX devices provide 16 differential `CLK[15:0]` input pins located in non-transceiver I/O banks that you can use to provide the input reference clock to the transceiver blocks. The Quartus II software automatically chooses the global clock (GCLK) network to route the input reference clock signal from the CLK pins to the transceiver blocks.

☞ For more information, refer to the Dedicated Clock Input Pins section in the *Clock Networks and PLLs in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook*.

One global clock resource is available for each CMU PLL and receiver CDR within a transceiver block. This allows each CMU PLL and receiver CDR to derive its input reference clock from a separate FPGA CLK input pin.

Clock Output from Left and Right PLLs in the FPGA Fabric

You can use the synthesized clock output from one of the left or right PLLs in the FPGA fabric to provide the input reference clock to the CMU PLLs and receiver CDRs. Stratix IV GX devices provide a dedicated clock path from the left PLLs (`PLL_L1`, `PLL_L2`, `PLL_L3`, and `PLL_L4`) in the FPGA fabric to the PLL cascade network in the transceiver blocks located on the left side of the device. Stratix IV GX devices also provide a dedicated clock path from the right PLLs (`PLL_R1`, `PLL_R2`, `PLL_R3`, and `PLL_R4`) in the FPGA fabric to the PLL cascade network in the transceiver blocks located on the right side of the device. The additional clock multiplication factors available in the left and right PLLs allow more options for on-board crystal oscillator frequencies. For more details, refer to "FPGA Fabric PLLs-Transceiver PLLs Cascading" on page 2–57.

Transceiver Channel Datapath Clocking

This section details the transmitter channel and receiver channel datapath clocking in various configurations. Datapath clocking varies with physical coding sublayer (PCS) configurations in different functional modes as well as channel bonding options.

Transmitter Channel Datapath Clocking

This section details the transmitter channel PMA and PCS datapath clocking in non-bonded and bonded channel configurations. Transmitter datapath clocking in bonded channel configurations is set up to provide low channel-to-channel skew when compared to non-bonded channel configurations.

The following factors contribute to transmitter channel-to-channel skew:

- High-speed serial clock and low-speed parallel clock skew between channels
- Unequal latency in the transmitter phase compensation FIFO

In non-bonded channel configurations, the high-speed serial clock and low-speed parallel clock in each channel are generated independently by its local clock divider. This results in higher channel-to-channel clock skew. The transmitter phase compensation FIFO in each non-bonded channel has its own pointers and control logic that can result in unequal latency in the transmitter phase compensation FIFO of each channel. The higher transceiver clock skew and unequal latency in the transmitter phase compensation FIFO in each channel can result in higher channel-to-channel skew in non-bonded channel configurations.

In bonded channel configurations, the high-speed serial clock and low-speed parallel clock for all bonded channels are generated by the same `CMU0` clock divider block, resulting in lower channel-to-channel clock skew. The transmitter phase compensation FIFO in all bonded channels share common pointers and control logic generated in `CMU0` channel, resulting in equal latency in the transmitter phase compensation FIFO of all bonded channels. The lower transceiver clock skew and equal latency in the transmitter phase compensation FIFOs in all channels provides lower channel-to-channel skew in bonded channel configurations.

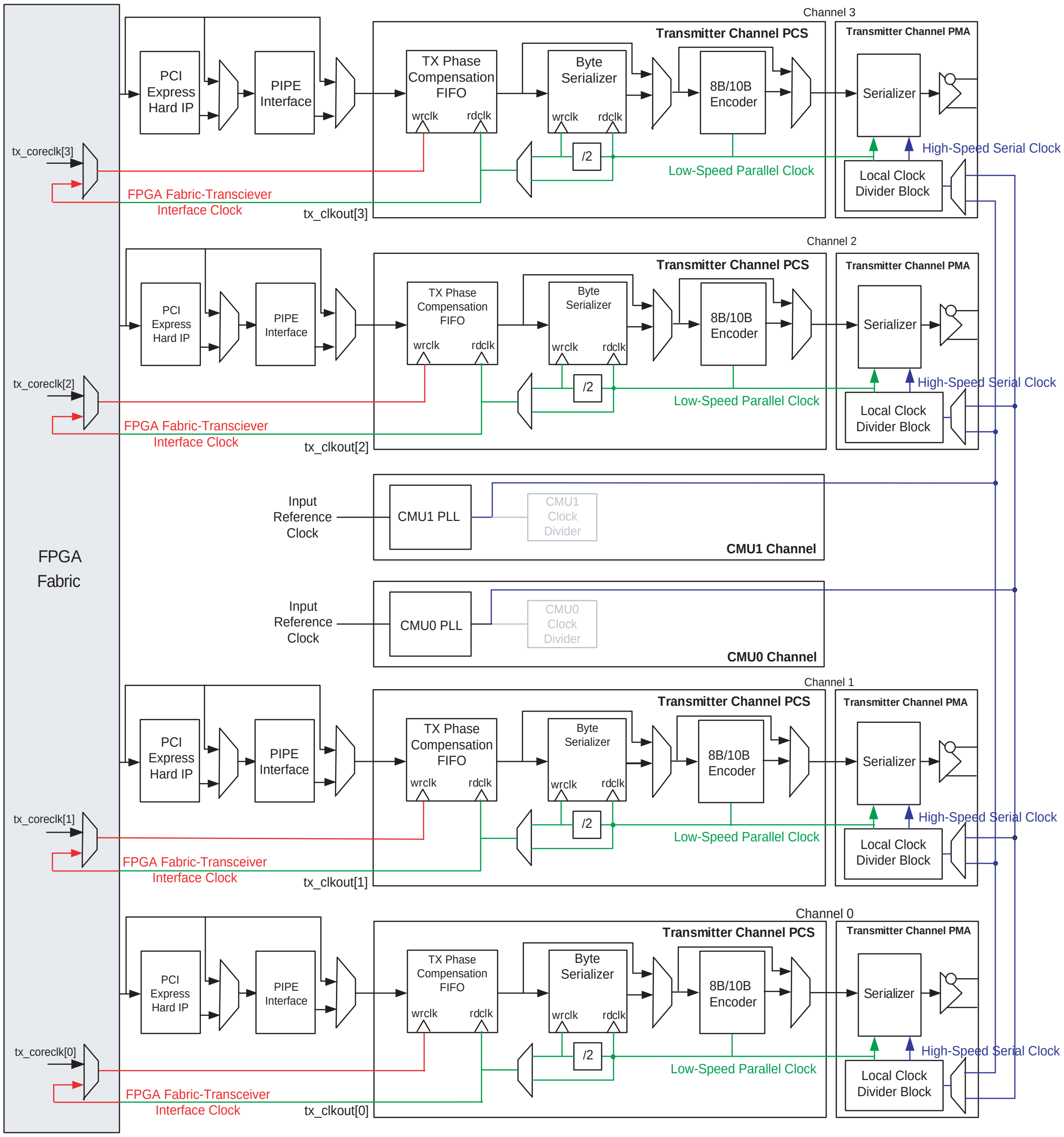
Non-Bonded Channel Configurations

The following functional modes have non-bonded transmitter channel configuration:

- PCI Express (PIPE) ×1
- Gigabit Ethernet (GIGE)
- Serial RapidIO
- SONET/SDH
- SDI
- (OIF) CEI PHY Interface (without low-jitter option selected)
- Basic (except Basic ×4 mode)

Figure 2–5 shows the transmitter channel datapath clocking in a non-bonded configuration.

Figure 2–5. Transmitter Datapath Clocking in a Non-Bonded Configuration



In non-bonded channel configurations, each channel can derive its clock independently from either `CMU0 PLL` or `CMU1 PLL` within the same transceiver block. The CMU PLL synthesizes the input reference clock to generate a clock that runs at a frequency of half the configured data rate. This half-rate clock from the CMU PLL is fed to the local clock divider block in each channel. Depending on the configured functional mode, the local clock divider block in each channel generates the low-speed parallel clock and the high-speed serial clock. The serializer in the transmitter channel PMA uses both the low-speed parallel clock and high-speed serial clock for its parallel-in-serial-out operation. The low-speed parallel clock clocks the 8B/10B encoder (if enabled) and the write port of the byte serializer (if enabled) in the transmitter channel PCS.

If the configured functional mode does not use the byte serializer, the low-speed parallel clock clocks the read port of the transmitter phase compensation FIFO. The low-speed parallel clock is also driven directly on the `tx_clkout` port as the FPGA fabric-Transceiver interface clock. You can use the `tx_clkout` port to clock transmitter data and control logic in the FPGA fabric.

If the configured functional mode uses a byte serializer to reduce the FPGA fabric-Transceiver interface speed, the low-speed parallel clock is divided by two. This divide-by-two version of the low-speed parallel clock clocks the read port of the transmitter phase compensation FIFO. It is also driven on the `tx_clkout` port as the FPGA fabric-transceiver interface clock. You can use the `tx_clkout` to clock transmitter data and control logic in the FPGA fabric.

Table 2–2 shows the transmitter channel datapath clock frequencies in non-bonded functional modes that have a fixed data rate.

Table 2–2. Transmitter Channel Datapath Clock Frequencies in Non-Bonded Functional Modes

Functional Mode	Data Rate	High-Speed Serial Clock Frequency	Low-Speed Parallel Clock Frequency	FPGA Fabric-Transceiver Interface Clock Frequency	
				Without Byte Serializer	With Byte Serializer
PCI Express (PIPE) ×1 (Gen 1)	2.5 Gbps	1.25 GHz	250 MHz	250 MHz	125 MHz
PCI Express (PIPE) ×1 (Gen 2)	5 Gbps	2.5 GHz	500 MHz	N/A	250 MHz
GIGE	1.25 Gbps	625 MHz	125 MHz	125 MHz	N/A
Serial RapidIO	1.25 Gbps	625 MHz	125 MHz	N/A	62.5 MHz
	2.5 Gbps	1.25 GHz	250 MHz	N/A	125 MHz
	3.125 Gbps	1.5625 GHz	312.5 MHz	N/A	156.25 MHz
SONET/SDH OC12	622 Mbps	311 MHz	77.75 MHz	77.75 MHz	N/A
SONET/SDH OC48	2.488 Gbps	1.244 GHz	311 MHz	N/A	155.5 MHz
HD-SDI	1.485 Gbps	742.5 MHz	148.5 MHz	148.5 MHz	74.25 MHz
	1.4835 Gbps	741.75 MHz	148.35 MHz	148.35 MHz	74.175 MHz
3G-SDI	2.97 Gbps	1.485 GHz	297 MHz	N/A	148.5 MHz
	2.967 Gbps	1.4835 GHz	296.7 MHz	N/A	148.35 MHz

Bonded Channel Configurations

Stratix IV GX devices support the following two types of transmitter channel bonding:

- PCS and PMA Bonding
- PMA-Only Bonding

PCS and PMA Bonding

In PCS and PMA bonded channel configurations, the PCS and PMA blocks of all bonded channels are clocked by the same low-speed parallel clock and high-speed serial clock from the CMU0 clock divider. The phase compensation FIFOs of all bonded channels also share common read and write pointers and enable signals generated in the CMU0 channel.

Stratix IV GX devices support ×4 PCS and PMA channel bonding that allows bonding of four channels within the same transceiver block. Stratix IV GX devices also support ×8 channel bonding that allows bonding of eight PCS and PMA channels across two transceiver blocks on the same side of the device.

x4 PCS and PMA Bonded Channel Configuration

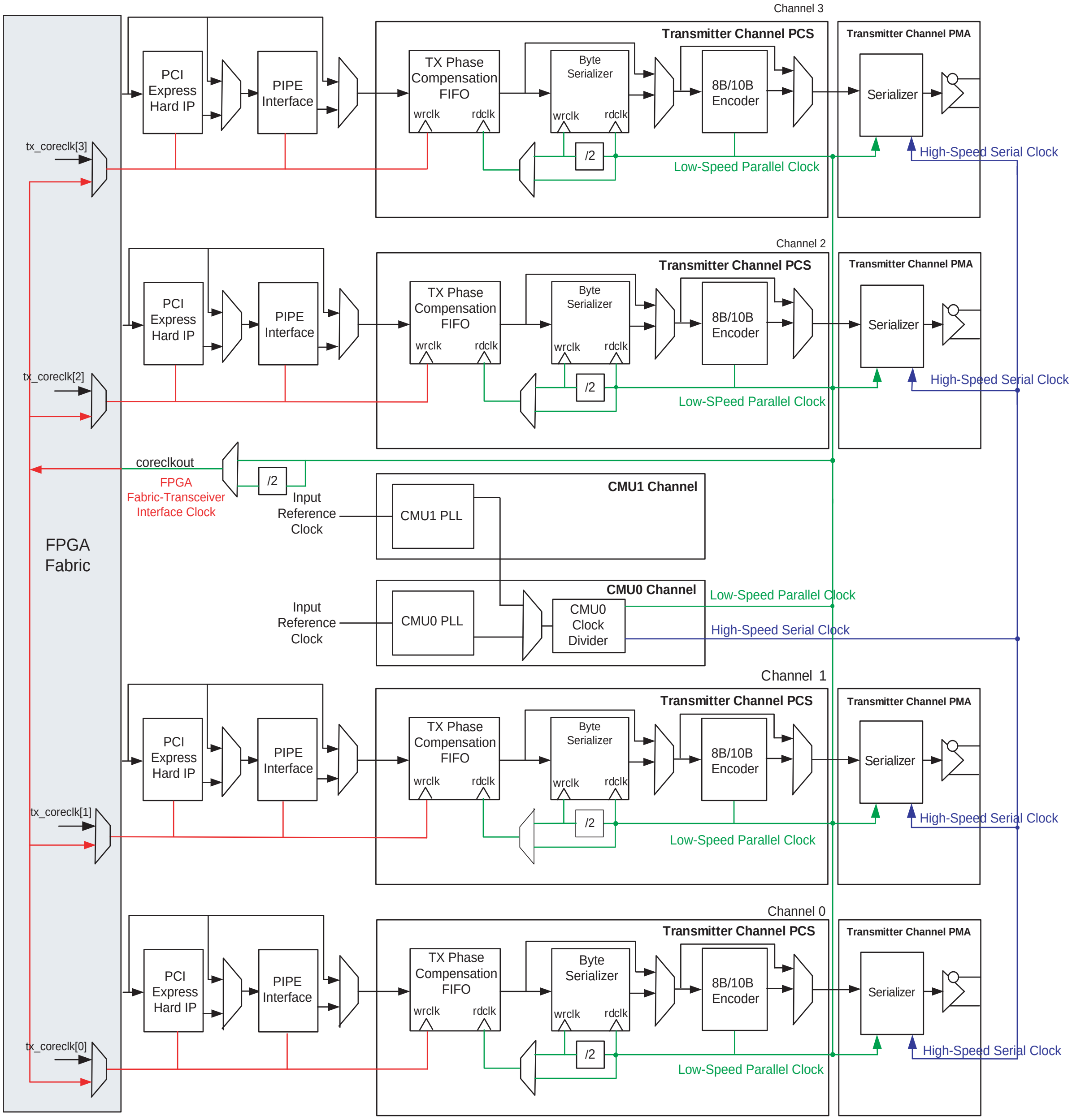
The following functional modes support ×4 PCS and PMA bonded transmitter channel configuration:

- PCI Express (PIPE) ×4
- XAUI
- Basic ×4

Figure 2–6 shows the transmitter channel datapath clocking in ×4 channel bonding configurations.

☞ You must assign tx_dataout[0] of the ×4 bonded link (XAUI or PCI Express [PIPE] ×4) to physical channel 0 of the transceiver block, tx_dataout[1] to physcial channel 1 of the transceiver block, tx_dataout[2] to physical channel 2 of the transceiver block, tx_dataout[3] to physical channel 3 of the transceiver block. Otherwise, the Quartus II compilation errors out.

Figure 2–6. Transmitter Datapath Clocking in x4 Bonded Configurations



In ×4 PCS and PMA bonded channel configurations, `CMU0 PLL` or `CMU1 PLL` synthesizes the input reference clock to generate a clock that runs at a frequency of half the configured data rate. The half-rate clock from either of the CMU PLLs is fed to the `CMU0` clock divider in `CMU0 Channel`. Depending on the configured functional mode, the `CMU0` clock divider block generates the high-speed serial clock and the low-speed parallel clock. The serializer in the transmitter channel PMA of the four bonded channels uses the same low-speed parallel clock and high-speed serial clock from `CMU0 Channel` for their parallel-in-serial-out operation. The low-speed parallel clock clocks the 8B/10B encoder and the write port of the byte serializer (if enabled) in the transmitter channel PCS.

If the configured functional mode does not use the byte serializer, the low-speed parallel clock from the `CMU0` clock divider block clocks the read port of the transmitter phase compensation FIFO in all four bonded channels. This low-speed parallel clock is also driven directly on the `coreclkout` port as the FPGA fabric-transceiver interface clock. You can use the `coreclkout` signal to clock the transmitter data and control logic in the FPGA fabric for all four bonded channels.

If the configured functional mode uses the byte serializer, the low-speed parallel clock from the `CMU0` clock divider is divided by two. This divide-by-two version of the low-speed parallel clock clocks the read port of the transmitter phase compensation FIFO in all four bonded channels. It is also driven on the `coreclkout` port as the FPGA fabric-transceiver interface clock. You can use the `coreclkout` signal to clock the transmitter data and control logic in the FPGA fabric for all four bonded channels.

In ×4 PCS and PMA bonded channel configurations, the transmitter phase compensation FIFOs in all four bonded channels share common read and write pointers and enable signals generated in the `CMU0` channel of the transceiver block. This ensures equal transmitter phase compensation FIFO latency across all four bonded channels, resulting in low transmitter channel-to-channel skew.

Table 2–3 shows the transmitter datapath clock frequencies in ×4 bonded functional modes that have a fixed data rate.

Table 2–3. Transmitter Datapath Clock Frequencies in x4 Bonded Functional Modes

Functional Mode	Data Rate	High-Speed Serial Clock Frequency	Low-Speed Parallel Clock Frequency	FPGA Fabric-Transceiver Interface Clock Frequency	
				Without Byte Serializer	With Byte Serializer
PCI Express (PIPE) ×4 (Gen 1)	2.5 Gbps	1.25 GHz	250 MHz	250 MHz	125 MHz
PCI Express (PIPE) ×4 (Gen 2)	5 Gbps	2.5 GHz	500 MHz	N/A	250 MHz
XAUI	3.125 Gbps	1.5625 GHz	312.5 MHz	N/A	156.25 MHz

x8 PCS and PMA Bonded Channel Configuration

The PCI Express (PIPE) ×8 and Basic ×8 functional modes support ×8 PCS and PMA bonded channel configuration. The eight bonded channels are located in two transceiver blocks, referred to as the master transceiver block and the slave transceiver block, with four channels each. The `CMU0` clock divider in `CMU0 Channel` of the master transceiver block provides the serial PMA clock and the parallel PCS clock to all eight bonded channels. The serializer in the transmitter channel PMA of the eight bonded channels uses the same low-speed parallel clock and high-speed serial clock from `CMU0 Channel` of the master transceiver block for their parallel-in-serial-out operation. The low-speed parallel clock from `CMU0 Channel` of the master transceiver block clocks the 8B/10B encoder and the write port of the byte serializer (if enabled) in the transmitter channel PCS of all eight channels.

In ×8 configurations that do not use the byte serializer, the low-speed parallel clock from the `CMU0` clock divider block in the master transceiver block clocks the read port of the transmitter phase compensation FIFO in all eight bonded channels. This low-speed parallel clock is also driven directly on the `coreclkout` port as the FPGA fabric-transceiver interface clock. You can use the `coreclkout` signal to clock the transmitter data and control logic in the FPGA fabric for all eight bonded channels.

In ×8 configurations that use the byte serializer, the low-speed parallel clock from the `CMU0` clock divider block in the master transceiver block is divided by two. This divide-by-two version of the low-speed parallel clock clocks the read port of the transmitter phase compensation FIFO in all eight bonded channels. It is also driven on the `coreclkout` port as the FPGA fabric-transceiver interface clock. You can use the `coreclkout` signal to clock the transmitter data and control logic in the FPGA fabric for all eight bonded channels.

In PCI Express (PIPE) ×8 and Basic ×8 bonded channel configurations, the transmitter phase compensation FIFOs in all eight bonded channels share common read and write pointers and enable signals generated in the `CMU0` channel of the master transceiver block. This ensures equal transmitter phase compensation FIFO latency across all eight bonded channels, resulting in low transmitter channel-to-channel skew.

Figure 2–7 shows the transmitter datapath clocking in PCI Express (PIPE) ×8 channel bonding configurations.

Figure 2–7. Transmitter Datapath Clocking in x8 Bonded Configuration

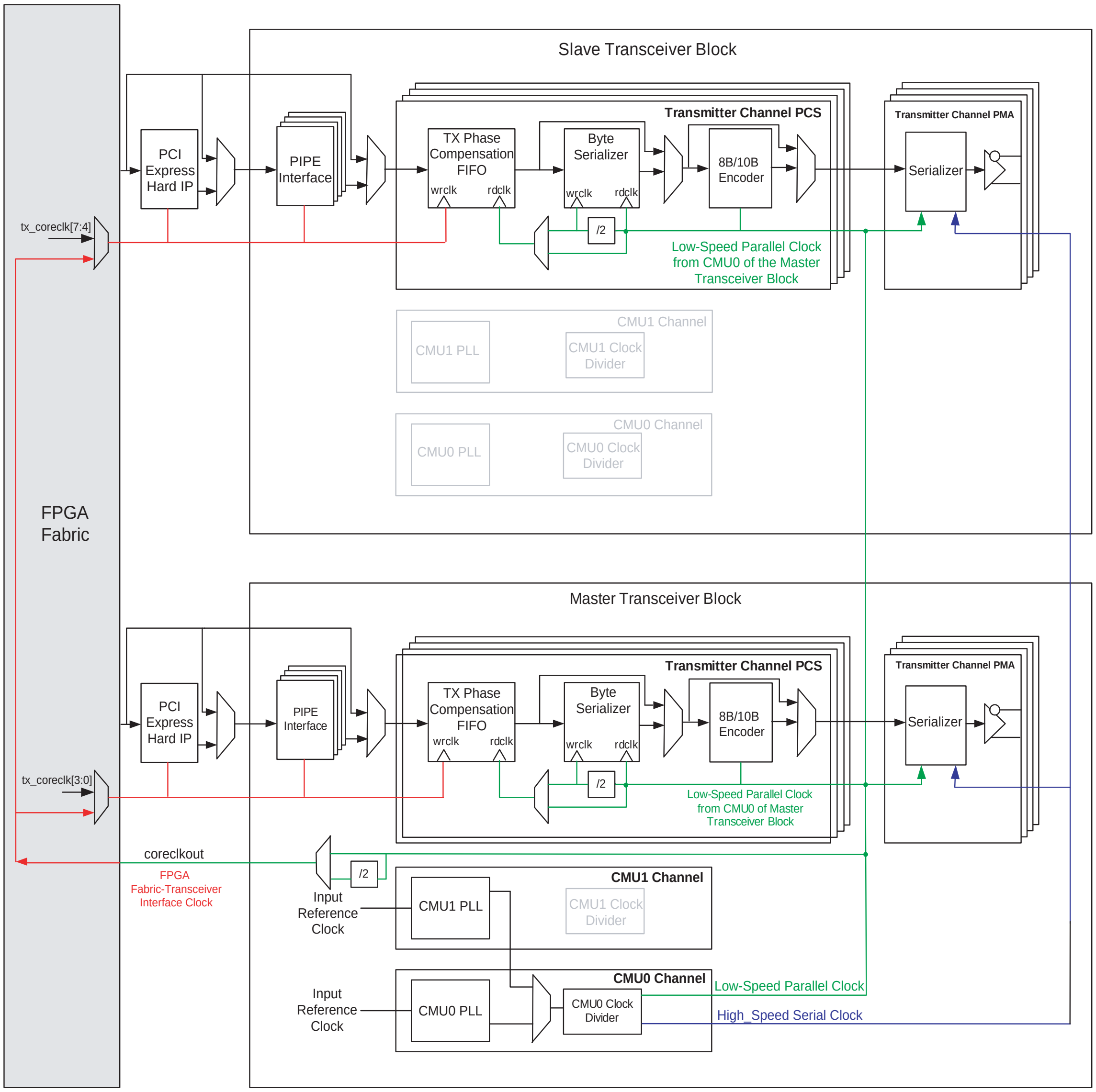


Figure 2–8 through 2–17 show allowed master and slave transceiver block locations and PCI Express (PIPE) logical lane to physical transceiver channel mapping in all Stratix IV GX devices.

The Quartus II compilation errors out if you do not map the PCI Express (PIPE) logical lanes to the physical transceiver channels, as shown in Figure 2–8 through 2–17.

Figure 2–8. One PCI Express (PIPE) x8 Link in Two Transceiver Block Devices

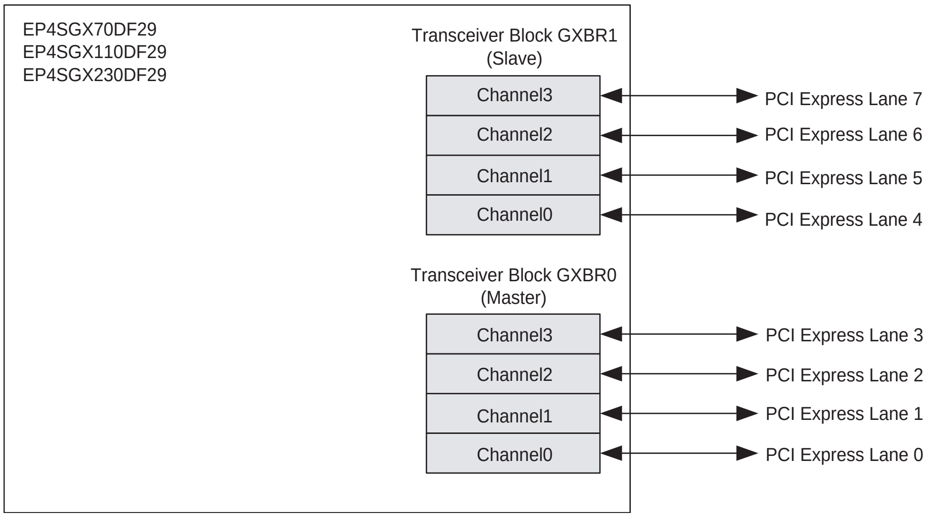


Figure 2–9. Two PCI Express (PIPE) x8 Link in Four Transceiver Block Devices

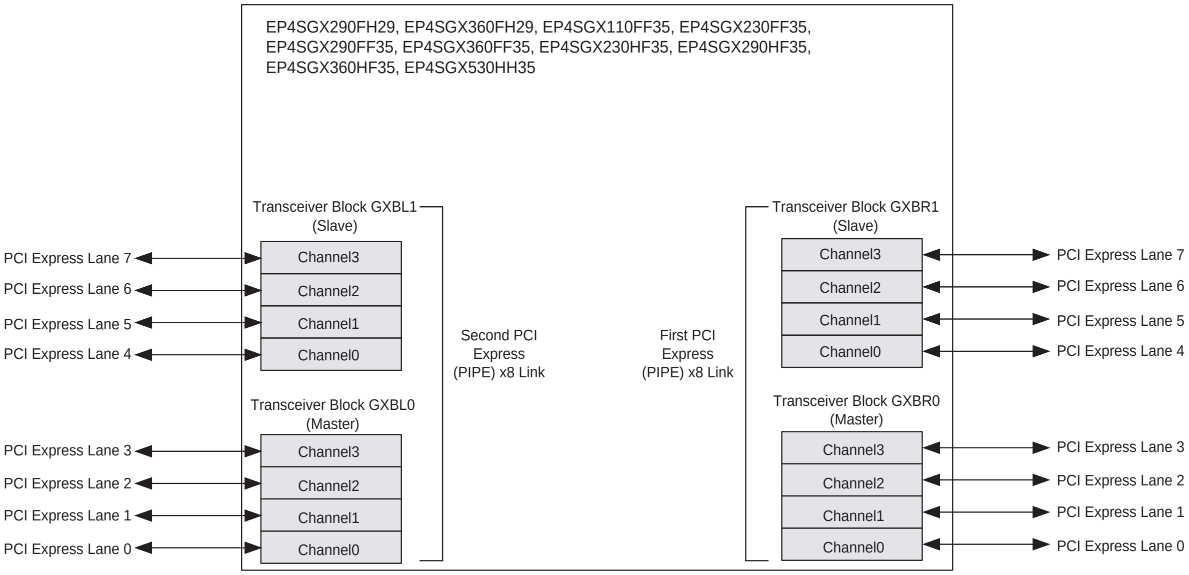
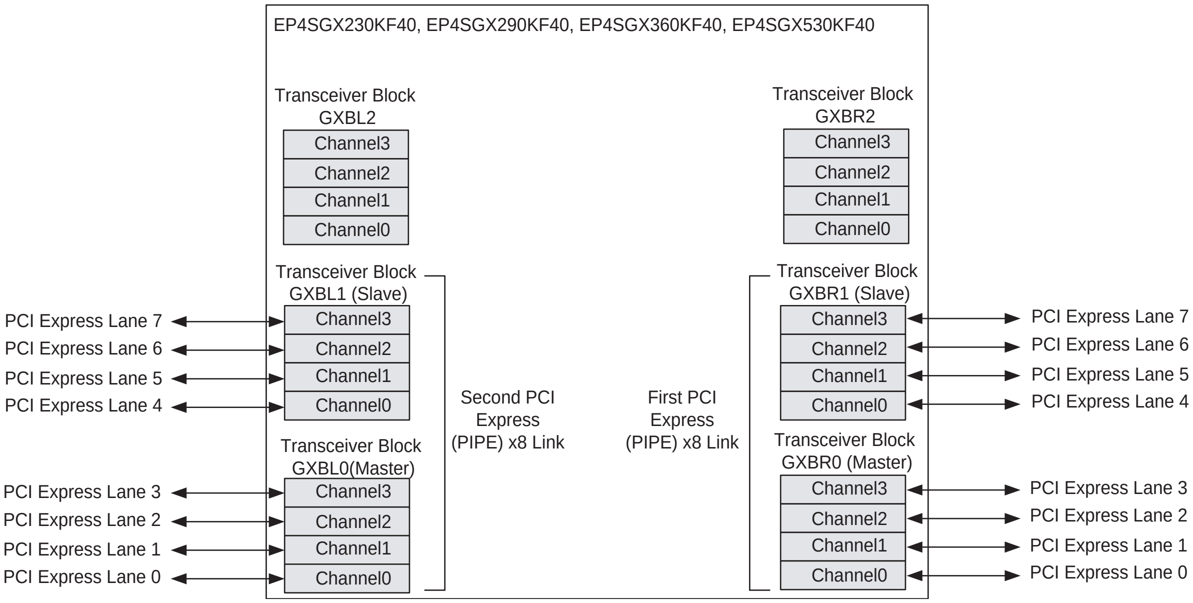


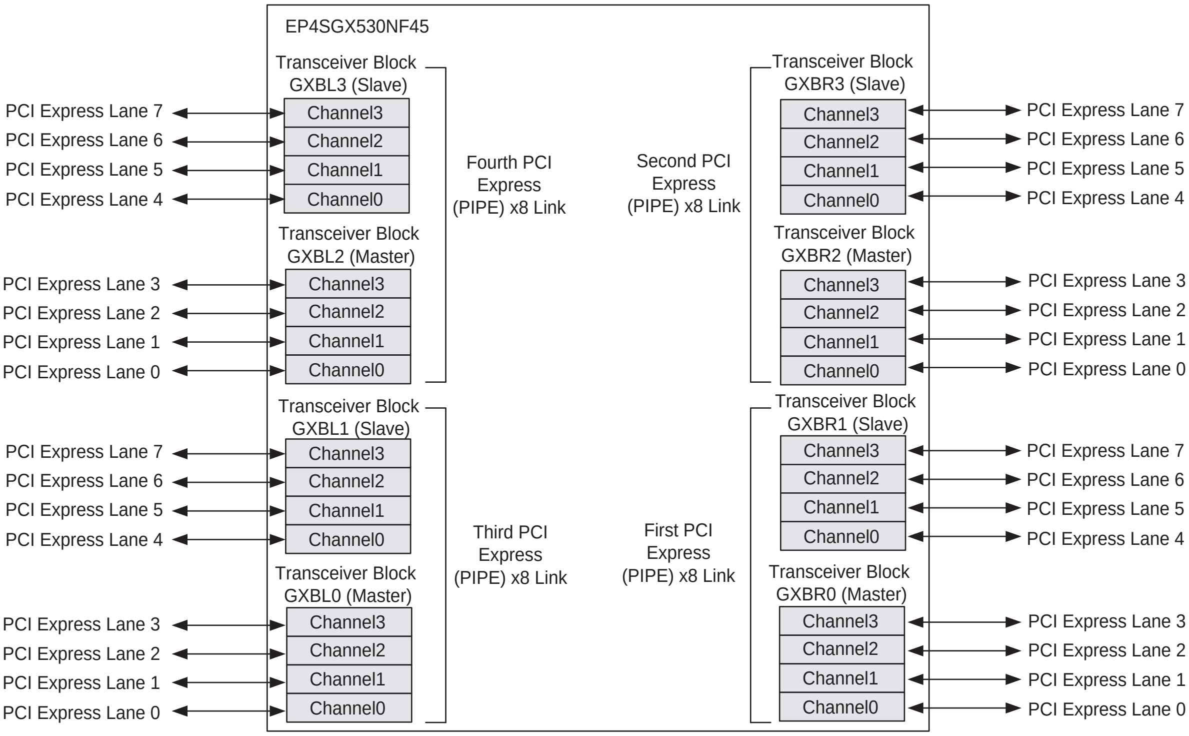
Figure 2–10. Two PCI Express (PIPE) x8 Link in Six Transceiver Block Devices *(Note 1)*



Note to Figure 2–10:

(1) Stratix IV GX devices with six transceiver blocks allow a maximum of two PCI Express (PIPE) ×8 links occupying four transceiver blocks. You can configure the other two transceiver blocks to implement other functional modes.

Figure 2–11. Four PCI Express (PIPE) x8 Link in Eight Transceiver Block Devices



PMA-Only Bonding

In PMA-Only bonded channel configurations, the PMA blocks of all bonded channels are clocked by the same high-speed serial clock from the `CMU0` clock divider. The PCS blocks of all bonded channels are not bonded; that is, the low-speed parallel clock that clocks the PCS blocks is generated independently in each transmitter channel. The phase compensation FIFOs of all PMA-Only bonded channels do not share common read and write pointers and enable signals.

The following functional mode supports PMA-Only bonded transmitter channel configuration:

- (OIF) CEI PHY Interface with the Use central clock divider to improve transmitter jitter option selected

☞ PMA-Only bonding is not available in any other functional mode except (OIF) CEI PHY Interface mode.

Receiver Channel Datapath Clocking

This section details the receiver PMA and PCS datapath clocking in supported configurations. The receiver datapath clocking varies between non-bonded and bonded channel configurations. It also varies with the use of PCS blocks, like deskew FIFO and rate matcher.

Non-Bonded Channel Configurations

In non-bonded channel configurations, receiver PCS blocks of each channel are clocked independently. Each non-bonded channel also has separate `rx_analogreset` and `rx_digitalreset` signals that allow independent reset of the receiver PCS logic in each channel.

☞ For more information about transceiver reset and power down signals, refer to the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook*.

In addition, in non-bonded channel configurations, receiver PCS clocking varies, depending on whether the configured functional mode uses the rate matcher block or not.

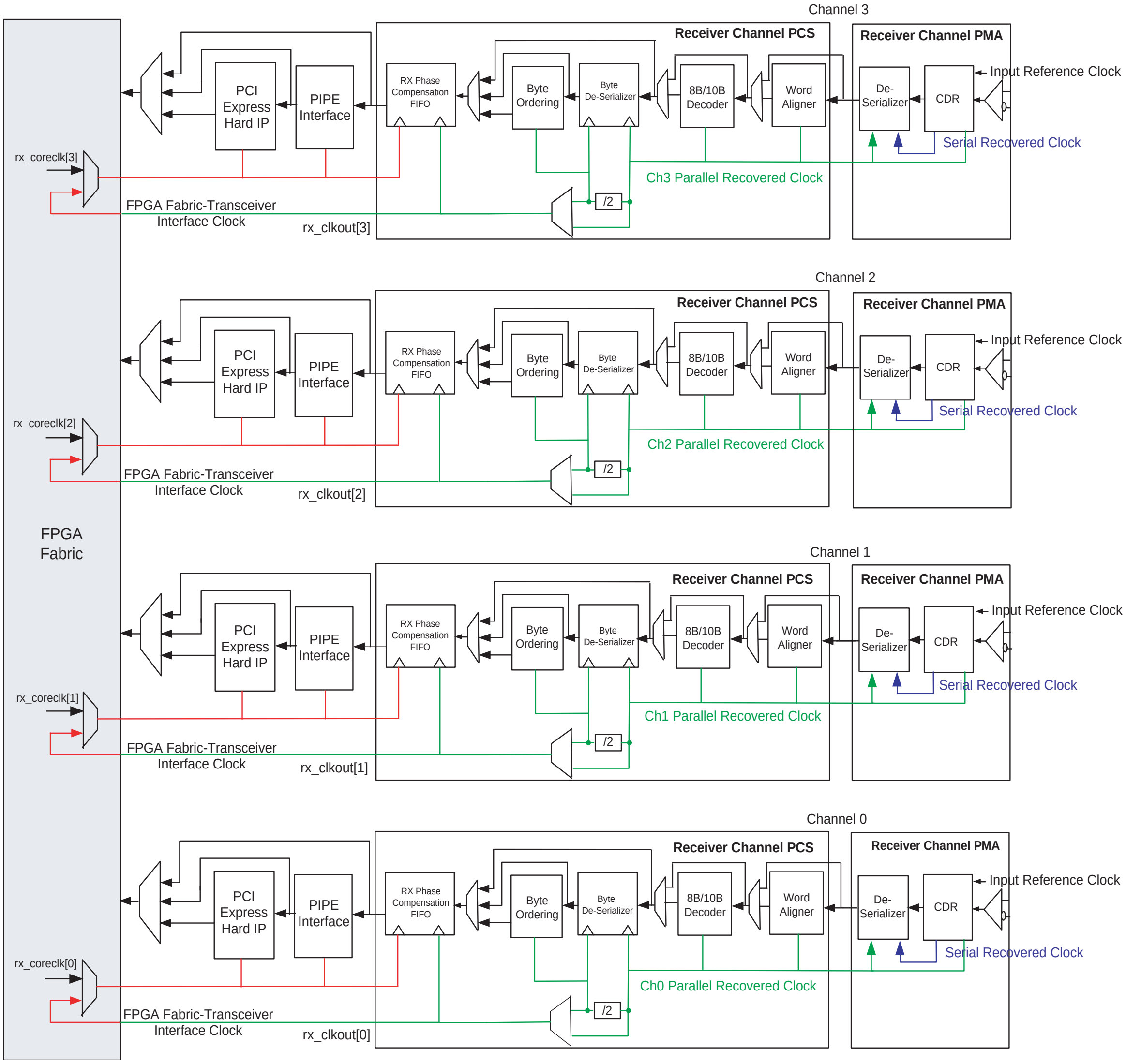
Non-Bonded Receiver Clocking without Rate Matcher

The following functional modes have non-bonded receiver channel configuration without rate-matcher:

- SONET/SDH
- SDI
- (OIF) CEI PHY Interface
- Basic without rate matcher

Figure 2–12 shows receiver datapath clocking in non-bonded channel configurations without rate matcher.

Figure 2–12. Receiver Datapath Clocking in Non-Bonded Configurations without Rate Matcher



In non-bonded configurations without rate matcher, the CDR in each receiver channel recovers the serial clock from the received data. Also, the serial recovered clock frequency is half the configured data rate due to the half-rate CDR architecture. The serial recovered clock is divided within the receiver PMA to generate the parallel recovered clock. The deserializer uses the serial recovered clock in the receiver PMA. The parallel recovered clock and deserialized data is forwarded to the receiver PCS. The parallel recovered clock in each channel clocks the word aligner and 8B/10B decoder (if enabled).

If the configured functional mode does not use the byte deserializer, the parallel recovered clock also clocks the write side of the receiver phase compensation FIFO. It is also driven on the `rx_clkout` port as the FPGA fabric-Transceiver interface clock. You can use the `rx_clkout` signal to latch the receiver data and status signals in the FPGA fabric.

If the configured functional mode uses the byte deserializer, the parallel recovered clock is divided by two. This divide-by-two version of the parallel recovered clock clocks the read side of the byte deserializer, the byte ordering block (if enabled), and the write side of the receiver phase compensation FIFO. It is also driven on the `rx_clkout` port as the FPGA fabric-transceiver interface clock. You can use the `rx_clkout` signal to latch the receiver data and status signals in the FPGA fabric.

Table 2–4 shows the receiver datapath clock frequencies in non-bonded functional modes without rate matcher.

Table 2–4. Receiver Datapath Clock Frequencies in Non-Bonded Functional Modes without Rate Matcher

				FPGA Fabric-Transceiver Interface Clock Frequency	
Functional Mode	**Data Rate**	**Serial Recovered Clock Frequency**	**Parallel Recovered Clock Frequency**	**Without Byte Deserializer**	**With Byte Deserializer**
SONET/SDH OC12	622 Mbps	311 MHz	77.75 MHz	77.75 MHz	N/A
SONET/SDH OC48	2.488 Gbps	1.244 GHz	311 MHz	N/A	155.5 MHz
HD-SDI	1.485 Gbps	742.5 MHz	148.5 MHz	148.5 MHz	74.25 MHz
	1.4835 Gbps	741.75 MHz	148.35 MHz	148.35 MHz	74.175 MHz
3G-SDI	2.97 Gbps	1.485 GHz	297 MHz	N/A	148.5 MHz
	2.967 Gbps	1.4835 GHz	296.7 MHz	N/A	148.35 MHz

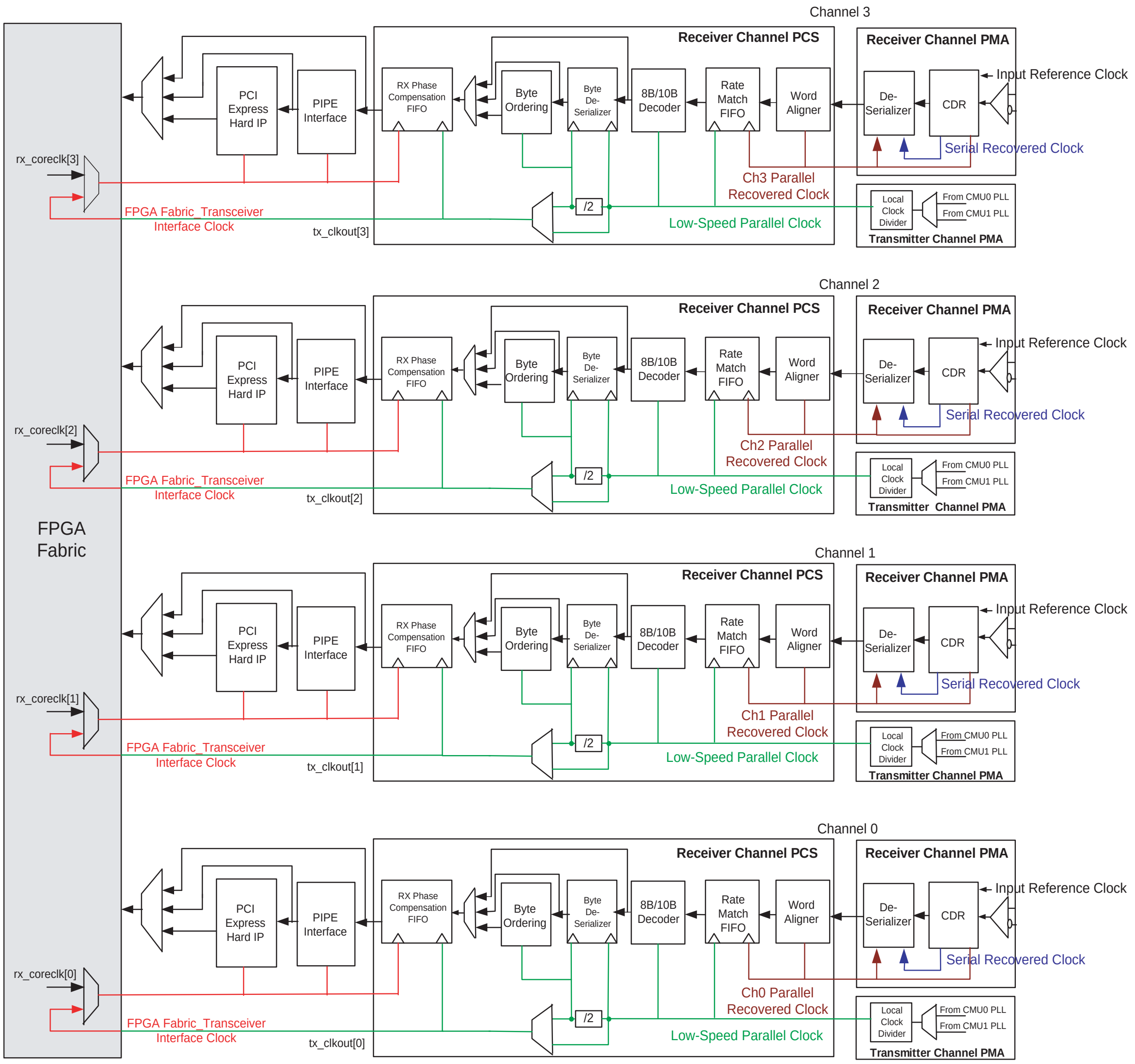
Non-Bonded Receiver Clocking with Rate Matcher

The following functional modes have non-bonded receiver channel configuration with rate-matcher:

- PCI Express (PIPE) ×1
- GIGE
- Serial RapidIO
- Basic with rate matcher

Figure 2–13 shows the receiver datapath clocking in non-bonded channel configurations with rate matcher.

Figure 2–13. Receiver Datapath Clocking in Non-Bonded Configurations with Rate Matcher



In non-bonded configurations with rate matcher, the CDR in each receiver channel recovers the serial clock from the received data. Also, the serial recovered clock frequency is half the configured data rate due to the half-rate CDR architecture. The serial recovered clock is divided within the receiver PMA to generate the parallel recovered clock. The deserializer uses the serial recovered clock in the receiver PMA. The parallel recovered clock and deserialized data is forwarded to the receiver PCS.

The parallel recovered clock from the receiver PMA in each channel clocks the word aligner and the write port of the rate match FIFO. The low-speed parallel clock from the transmitter local clock divider block in each channel clocks the read port of the rate match FIFO, the 8B/10B decoder, and the write port of the byte deserializer (if enabled). The parallel transmitter PCS clock or its divide-by-two version (if byte-deserializer is enabled) clocks the write port of the receiver phase compensation FIFO. It is also driven on the `tx_clkout` port as the FPGA fabric-transceiver interface clock. You can use the `tx_clkout` signal to latch the receiver data and status signals in the FPGA fabric.

Table 2–5 shows the receiver datapath clock frequencies in non-bonded functional modes with rate matcher.

Table 2–5. Receiver Datapath Clock Frequencies in Non-Bonded Functional Modes with Rate Matcher

Functional Mode	Data Rate	Serial Recovered Clock Frequency	Parallel Recovered Clock and Parallel Transmitter PCS Clock Frequency	FPGA Fabric-Transceiver Interface Clock Frequency	
				Without Byte Deserializer	With Byte Deserializer
PCI Express (PIPE) ×1 (Gen 1)	2.5 Gbps	1.25 GHz	250 MHz	250 MHz	125 MHz
PCI Express (PIPE) ×1 (Gen 2)	5 Gbps	2.5 GHz	500 MHz	N/A	250 MHz
GIGE	1.25 Gbps	625 MHz	125 MHz	125 MHz	N/A
Serial RapidIO	1.25 Gbps	625 MHz	125 MHz	N/A	62.5 MHz
	2.5 Gbps	1.25 GHz	250 MHz	N/A	125 MHz
	3.125 Gbps	1.5625 GHz	312.5 MHz	N/A	156.25 MHz

Bonded Channel Configurations

The Stratix IV GX device supports ×4 channel bonding that allows bonding of four channels within the same transceiver block. It also supports ×8 channel bonding that allows bonding of eight channels across two transceiver blocks on the same side of the device.

x4 Bonded Channel Configuration

The following functional modes support ×4 receiver channel bonded configuration:

- PCI Express (PIPE) ×4
- XAUI

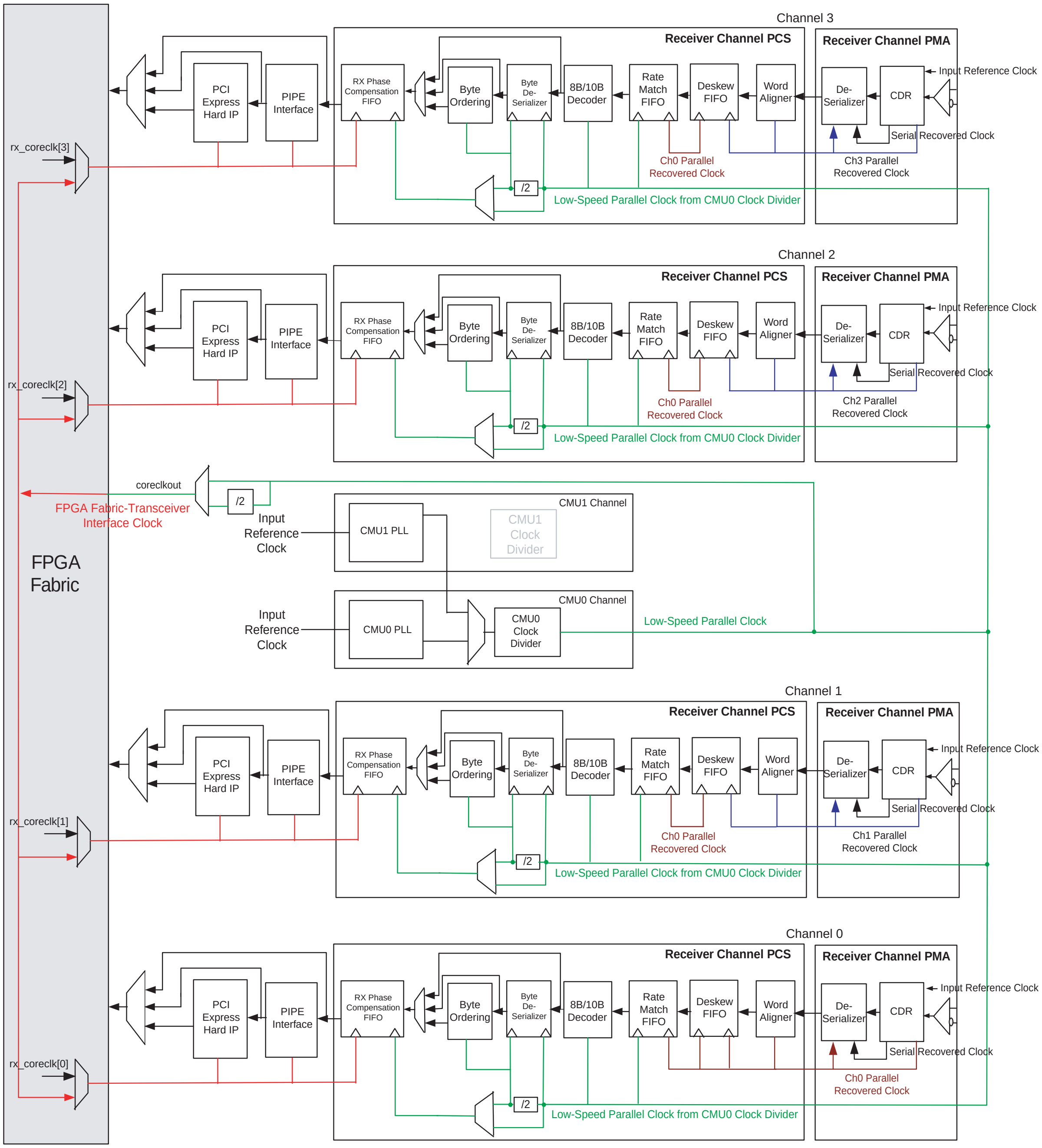
In ×4 bonded channel configurations, the receiver datapath clocking varies, depending on whether the configured functional mode uses the deskew FIFO or not.

x4 Bonded Channel Configuration with Deskew FIFO

XAUI functional mode has ×4 bonded channel configuration with deskew FIFO.

Figure 2–14 shows the receiver datapath clocking in ×4 channel bonding configurations with deskew FIFO.

Figure 2–14. Receiver Datapath Clocking in x4 Bonded Channel Configuration with Deskew FIFO



In ×4 bonded channel configurations with deskew FIFO, the CDR in each receiver channel recovers the serial clock from the received data. Also, the serial recovered clock frequency is half the configured data rate due to the half-rate CDR architecture. The serial recovered clock is divided within each channel's receiver PMA to generate the parallel recovered clock. The deserializer uses the serial recovered clock in the receiver PMA. The parallel recovered clock and deserialized data is forwarded to the receiver PCS in each channel.

The parallel recovered clock from the receiver PMA in each channel clocks the word aligner in that channel. The parallel recovered clock from `Channel 0` clocks the deskew FIFO and the write port of the rate match FIFO in all four bonded channels. The low-speed parallel clock from the `CMU0` clock divider block in `CMU0_Channel` clocks the read port of the rate match FIFO, the 8B/10B decoder, and the write port of the byte deserializer (if enabled) in all four bonded channels. The low-speed parallel clock or its divide-by-two version (if byte-deserializer is enabled) clocks the write port of the receiver phase compensation FIFO. It is also driven on the `coreclkout` port as the FPGA fabric-transceiver interface clock. You can use the `coreclkout` signal to latch the receiver data and status signals in the FPGA fabric for all four bonded channels.

In ×4 bonded channel configurations, the receiver phase compensation FIFOs in all four bonded channels share common read and write pointers and enable signals generated in the `CMU0` channel of the transceiver block.

Table 2–6 shows the receiver datapath clock frequencies in ×4 bonded functional modes with deskew FIFO.

Table 2–6. Receiver Datapath Clock Frequencies in x4 Bonded Functional Modes with Deskew FIFO

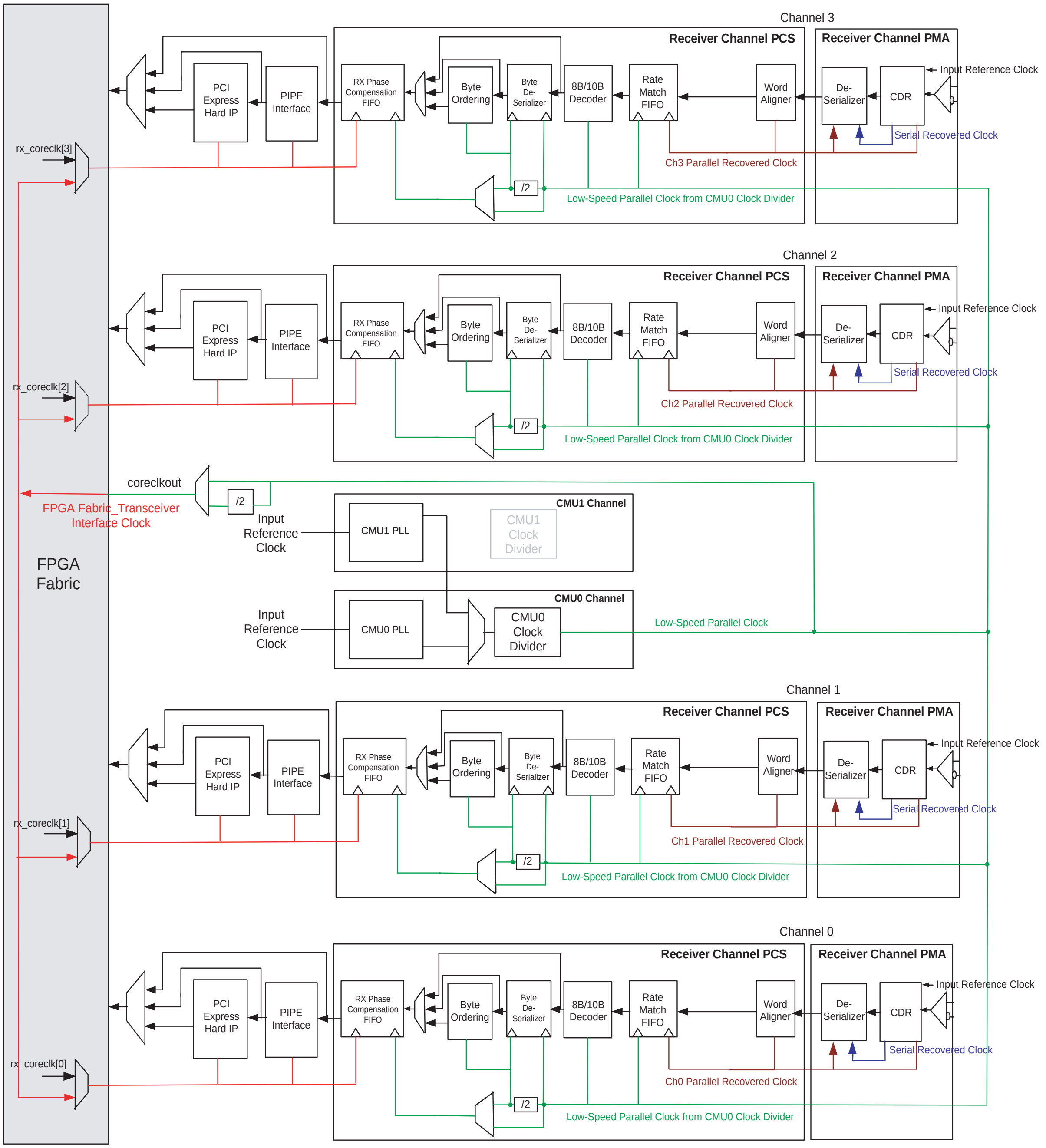
Functional Mode	Data Rate	Serial Recovered Clock Frequency	Parallel Recovered Clock and Parallel Transmitter PCS Clock Frequency	FPGA-Fabric Transceiver Interface Clock Frequency	
				Without Byte Deserializer	With Byte Deserializer
PCI Express (PIPE) ×1 (Gen 1)	2.5 Gbps	1.25 GHz	250 MHz	250 MHz	125 MHz
PCI Express (PIPE) ×1 (Gen 2)	5 Gbps	2.5 GHz	500 MHz	N/A	250 MHz
XAUI	3.125 Gbps	1.5625 MHz	312.5 MHz	N/A	156.25 MHz

x4 Bonded Channel Configuration without Deskew FIFO

PCI Express (PIPE) ×4 functional modes have ×4 bonded channel configuration without deskew FIFO.

Figure 2–15 shows the receiver datapath clocking in ×4 channel bonding configurations without deskew FIFO.

Figure 2–15. Receiver Datapath Clocking in x4 Bonded Channel Configuration without Deskew FIFO



In ×4 bonded channel configurations without deskew FIFO, the CDR in each receiver channel recovers the serial clock from the received data. The serial recovered clock frequency is half the configured data rate due to the half-rate CDR architecture. The serial recovered clock is divided within each channel’s receiver PMA to generate the parallel recovered clock. The deserializer uses the serial recovered clock in the receiver PMA. The parallel recovered clock and deserialized data is forwarded to the receiver PCS in each channel.

The parallel recovered clock from the receiver PMA in each channel clocks the word aligner and the write side of the rate matcher FIFO in that channel. The low-speed parallel clock from the CMU0 clock divider block in CMU0 Channel clocks the read port of the rate match FIFO, the 8B/10B decoder, and the write port of the byte deserializer (if enabled). The low-speed parallel clock or its divide-by-two version (if byte-deserializer is enabled) clocks the receiver phase compensation FIFO. It is also driven on the coreclkout port as the FPGA fabric-transceiver interface clock. You can use the coreclkout signal to latch the receiver data and status signals in the FPGA fabric for all four bonded channels.

In ×4 bonded channel configurations, the receiver phase compensation FIFOs in all four bonded channels share common read and write pointers and enable signals generated in the CMU0 channel of the transceiver block.

Table 2–7 shows the receiver datapath clock frequencies in ×4 bonded functional modes without deskew FIFO.

Table 2–7. Receiver Datapath Clock Frequencies in x4 Bonded Functional Modes without Deskew FIFO

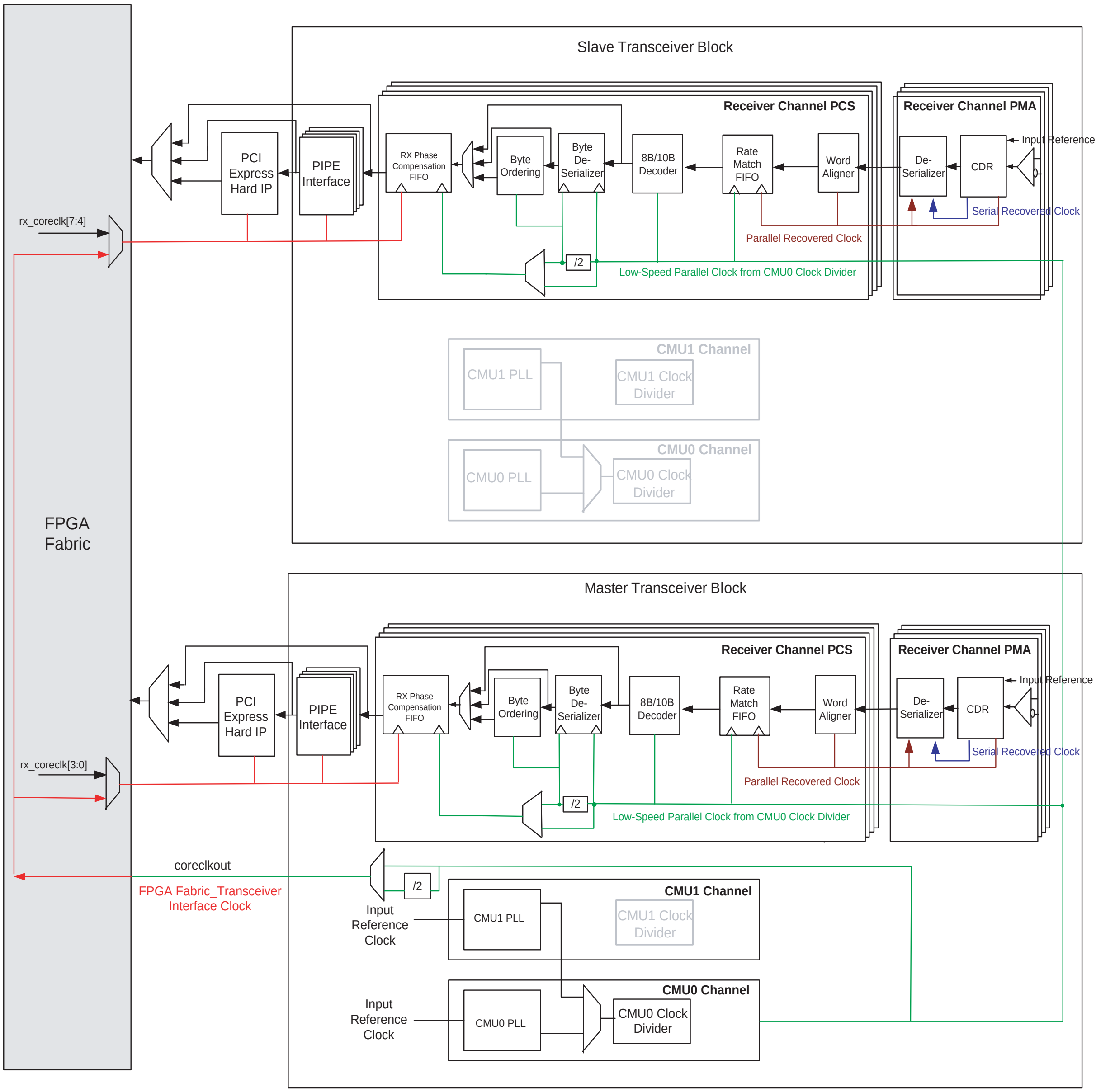
Functional Mode	Data Rate	Serial Recovered Clock Frequency	Parallel Recovered Clock and Parallel Transmitter PCS Clock Frequency	FPGA Fabric-Transceiver Interface Clock Frequency	
				Without Byte Deserializer	With Byte Deserializer
PCI Express (PIPE) ×4 (Gen 1)	2.5 Gbps	1.25 GHz	250 MHz	250 MHz	125 MHz
PCI Express (PIPE) ×4 (Gen 2)	5 Gbps	2.5 GHz	500 MHz	N/A	250 MHz

x8 Bonded Channel Configuration

PCI Express (PIPE) ×8 functional mode supports the ×8 receiver channel bonding configuration. The eight bonded channels are located in two transceiver blocks, referred to as the master transceiver block and the slave transceiver block, with four channels each.

Figure 2–16 shows the receiver datapath clocking in PCI Express (PIPE) ×8 bonded channel configuration.

Figure 2–16. Receiver Datapath Clocking in x8 Bonded Channel Configuration



The CDR in each of the eight receiver channels recovers the serial clock from the received data on that channel. The serial recovered clock frequency is half the configured data rate. The serial recovered clock is divided within each channel's receiver PMA to generate the parallel recovered clock. The deserializer uses the serial recovered clock in the receiver PMA. The parallel recovered clock and deserialized data from the receiver PMA in each channel is forwarded to the receiver PCS in that channel.

The parallel recovered clock from the receiver PMA in each channel clocks the word aligner and the write side of the rate matcher FIFO in that channel. The low-speed parallel clock from the CMU0 clock divider of the master transceiver block clocks the read port of the rate match FIFO, the 8B/10B decoder, and the write port of the byte deserializer (if enabled) in all eight channels. The low-speed parallel clock or its divide-by-two version (if byte-deserializer is enabled) clocks the write port of the receiver phase compensation FIFO in all eight channels. It is also driven on the coreclkout port as the FPGA fabric-transceiver interface clock. You can use the coreclkout signal to latch the receiver data and status signals in the FPGA fabric for all eight bonded channels.

The receiver phase compensation FIFO pointers and control circuitry from Channel 0 in the master transceiver block is shared by the receiver phase compensation FIFOs across all eight channels in PCI Express (PIPE) ×8 mode.

Table 2–8 shows the receiver datapath clock frequencies in PCI Express (PIPE) ×8 functional mode.

Table 2–8. Receiver Datapath Clock Frequencies PCI Express (PIPE) x8 Functional Mode

Functional Mode	Data Rate	Serial Recovered Clock Frequency	ParallelRecovered Clock and Parallel Transmitter PCS Clock Frequency	FPGA Fabric-Transceiver Interface Clock Frequency	
				Without Byte Deserializer	With Byte Deserializer
PCI Express (PIPE) ×8 (Gen 1)	2.5 Gbps	1.25 GHz	250 MHz	250 MHz	125 MHz
PCI Express (PIPE) ×8 (Gen 2)	5 Gbps	2.5 GHz	500 MHz	N/A	250 MHz

FPGA Fabric-Transceiver Interface Clocking

The FPGA fabric-Transceiver interface clocks consist of clock signals from the FPGA fabric to the transceiver blocks and clock signals from the transceiver blocks to the FPGA fabric.

The FPGA fabric-Transceiver interface clocks can be subdivided into the following three categories:

- "Input Reference Clock Source"
- "Phase Compensation FIFO Clocks"
- "Other Transceiver Clocks"

Input Reference Clocks

The CMU PLLs and the receiver CDRs in each transceiver block can derive the input reference from one of the following sources:

- refclk0 and refclk1 pins of the same transceiver block
- refclk0 and refclk1 pins of other transceiver blocks on the same side of the device using the inter transceiver block (ITB) clock network
- CLK input pins on the FPGA global clock (GCLK) network
- Clock output from the left and right PLLs in the FPGA fabric

☞ For more information, refer to "Input Reference Clock Source" on page 2–1.

If the input reference clock to the CMU PLL or receiver CDR is provided though the `FPGA CLK` input pins or the clock output from the left or right PLLs in the FPGA fabric, the input reference clock becomes a part of the FPGA fabric-Transceiver interface clocks. If the input reference clock is provided through the `FPGA CLK` input pins, the Quartus II software automatically routes the input reference clock on the FPGA fabric global clock (GCLK) network. If the output clock from a left or right PLL provides the input reference clock, the Quartus II software routes the input reference clock on a dedicated clock path from the left or right PLL to the CMU PLL or receiver CDR.

Phase Compensation FIFO Clocks

The transmitter and receiver phase compensation FIFOs in each channel ensure reliable transfer of data, control, and status signals between the FPGA fabric and the transceiver channels. The transceiver channel forwards the `tx_clkout` signal (in non-bonded modes) or `coreclkout` signal (in bonded channel modes) to the FPGA fabric to clock the data and control signals into the transmitter phase compensation FIFO. The transceiver channel also forwards the recovered clock `rx_clkout` (in configurations without rate matcher) or `tx_clkout/coreclkout` (in configurations with rate matcher) to the FPGA fabric to clock the data and status signals from the receiver phase compensation FIFO into the FPGA fabric.

The phase compensation FIFO clocks form a part of the FPGA fabric-Transceiver interface clocks and are routed on either a global clock resource (GCLK), regional clock resource (RCLK), or periphery clock (PCLK) resource in the FPGA fabric.

Other Transceiver Clocks

The following transceiver clocks form a part of the FPGA fabric-Transceiver interface clocks:

- `cal_blk_clk`—calibration block clock
- `fixed_clk`—125 MHz fixed-rate clock used in PCI Express receiver detect circuitry

The Quartus II software automatically routes `fixed_clk` on the FPGA fabric global clock (GCLK) or regional clock (RCLK) network.

Table 2–9 summarizes the FPGA fabric-Transceiver interface clocks.

Table 2–9. FPGA Fabric-Transceiver Interface Clocks *(Note 1)* (Part 1 of 2)

Clock Name	Clock Description	Interface Direction	FPGA Fabric Clock Resource Utilization *(1)*
`pll_inclk`	CMU PLL input reference clock when driven from an FPGA CLK input pin	FPGA fabric to transceiver	GCLK
`rx_cruclk`	Receiver CDR input reference clock when driven from an FPGA CLK input pin	FPGA fabric to transceiver	GCLK, RCLK
`tx_clkout`	Phase compensation FIFO clock	Transceiver to FPGA fabric	GCLK, RCLK, PCLK
`coreclkout`	Phase compensation FIFO clock	Transceiver to FPGA fabric	GCLK, RCLK, PCLK
`rx_clkout`	Phase compensation FIFO clock	Transceiver to FPGA fabric	GCLK, RCLK, PCLK

Table 2–9. FPGA Fabric-Transceiver Interface Clocks *(Note 1)* (Part 2 of 2)

Clock Name	Clock Description	Interface Direction	FPGA Fabric Clock Resource Utilization *(1)*
`fixed_clk`	PCI Express receiver detect clock	FPGA fabric to transceiver	GCLK, RCLK

Note to Table 2–9:

(1) For more information about GCLK, RCLK, and PCLK resources available in each device, refer to the *Clock Networks and PLLs in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook*.

The "FPGA Fabric-Transmitter Interface Clocking" on page 2–30 and "FPGA Fabric-Receiver Interface Clocking" on page 2–43 sections describe the criteria and methodology to share transmitter and receiver phase compensation FIFO clocks in order to reduce the GCLK, RCLK, and PCLK resource utilization in your design.

FPGA Fabric-Transmitter Interface Clocking

The transmitter phase compensation FIFO compensates for the phase difference between the FPGA fabric clock (phase compensation FIFO write clock) and the parallel transmitter PCS clock (phase compensation FIFO read clock). The transmitter phase compensation FIFO write clock forms the FPGA fabric-transmitter interface clock. The phase compensation FIFO write clock and read clocks must have exactly the same frequency (0 PPM frequency difference).

Stratix IV GX transceivers provide the following two options for selecting the transmitter phase compensation FIFO write clock:

- Quartus II software-selected transmitter phase compensation FIFO write clock
- User-selected transmitter phase compensation FIFO write clock

Quartus II-Selected Transmitter Phase Compensation FIFO Write Clock

If you do not select the `tx_coreclk` port in the ALTGX MegaWizard® Plug-In Manager, the Quartus II software automatically selects the transmitter phase compensation FIFO write clock for each channel in that ALTGX instance. The Quartus II software selects the FIFO write clock depending on the channel configuration.

Non-Bonded Channel Configuration

In non-bonded channel configuration, the transmitter channels may or may not be identical. Identical transmitter channels are defined as channels that have the same CMU PLL input reference clock source, have exactly the same CMU PLL configuration, and have exactly the same transmitter PMA and PCS configuration.

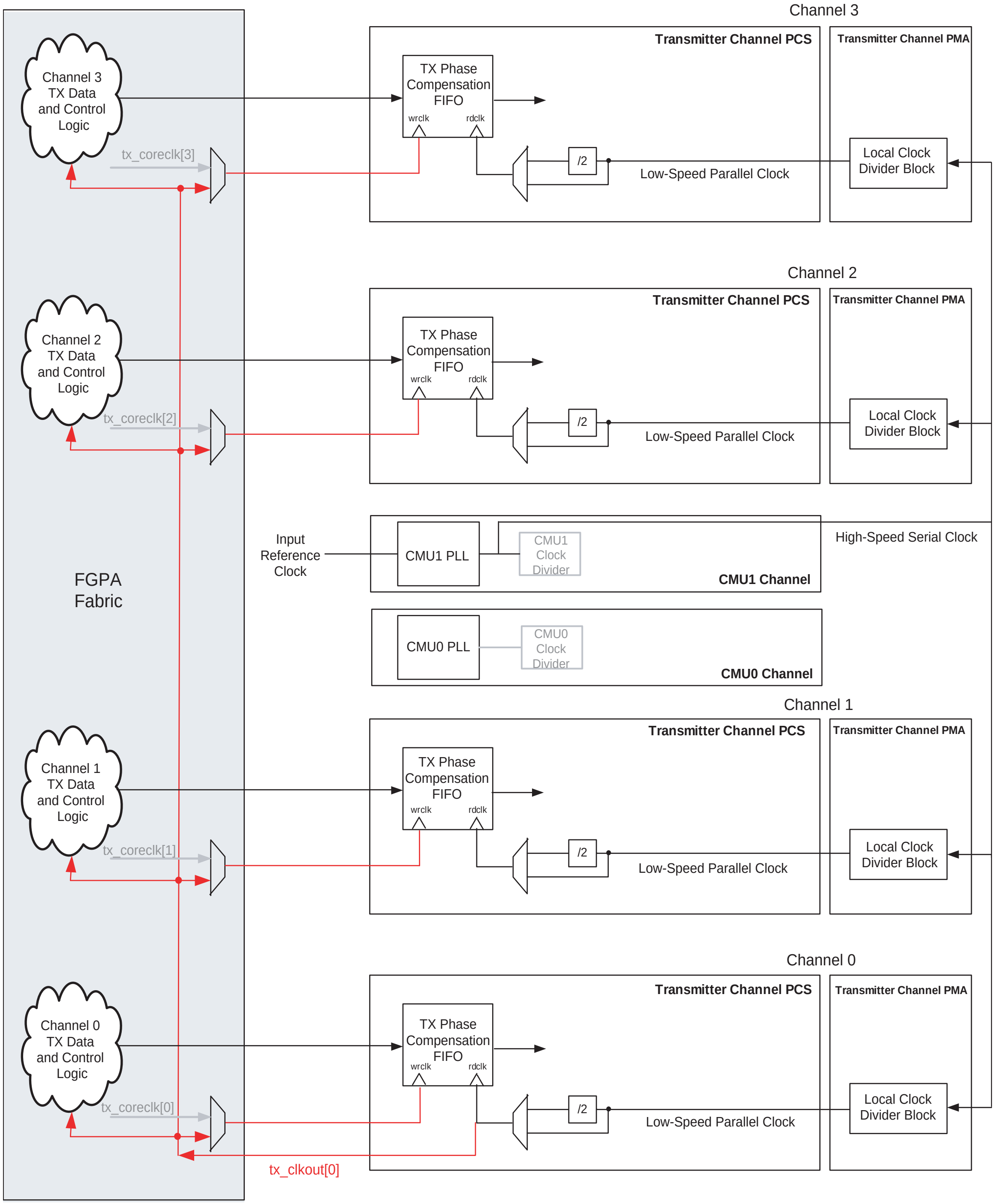
☞ Identical transmitter channels may have different transmitter voltage output differential (VOD), transmitter common mode voltage (VCM), or pre-emphasis setting.

Example 1: Four Identical Channels in a Transceiver Block

If all four channels within a transceiver block are identical, the Quartus II software automatically drives the write port of the transmitter phase compensation FIFO in all four channels with `tx_clkout[0]`, as shown in Figure 2–17. Use the `tx_clkout[0]` signal to clock the transmitter data and control logic for all four channels in the FPGA fabric.

This configuration uses only one FPGA GCLK, RCLK, or PCLK resource for `tx_clkout[0]`.

Figure 2–17. Four Identical Channels in a Transceiver Block for Example 1



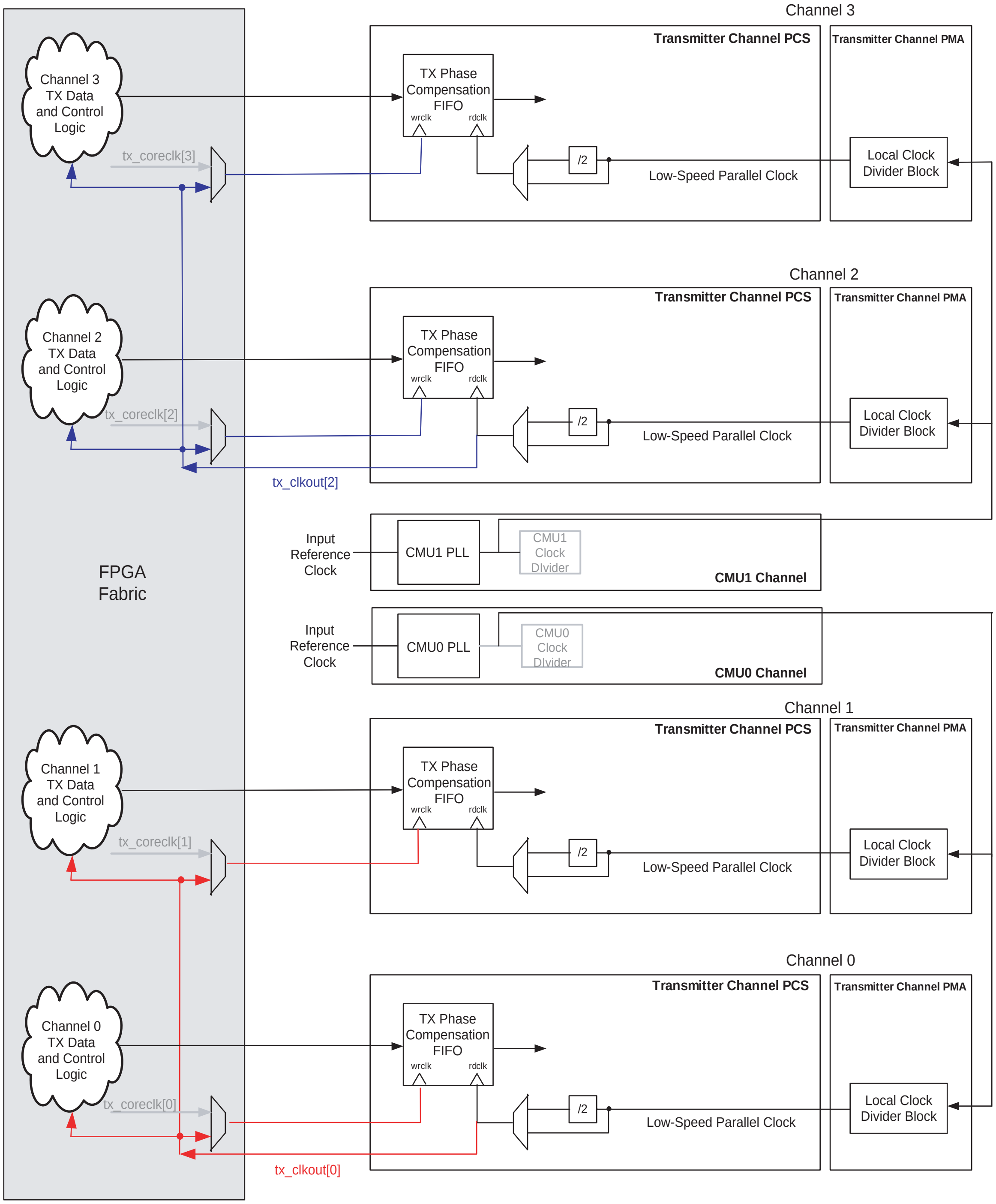
Example 2: Two Groups of Two Identical Channels in a Transceiver Block

Example 2 assumes channels 0 and 1, driven by `CMU0 PLL` in a transceiver block, are identical. Also, channels 2 and 3, driven by `CMU1 PLL` in the same transceiver block, are identical. In this case, the Quartus II software automatically drives the write port of the transmitter phase compensation FIFO in channels 0 and 1 with the `tx_clkout[0]` signal. It also drives the write port of the transmitter phase compensation FIFO in channels 2 and 3 with the `tx_clkout[2]` signal. Use the `tx_clkout[0]` signal to clock the transmitter data and control logic for channels 0 and 1 in the FPGA fabric. Use the `tx_clkout[2]` signal to clock the transmitter data and control logic for channels 2 and 3 in the FPGA fabric.

☞ This configuration uses two FPGA global and/or regional clock resources, one for the `tx_clkout[0]` signal and the other for the `tx_clkout[2]` signal.

Figure 2–18 shows the FPGA fabric-transmitter interface clocking for Example 2.

Figure 2–18. FPGA Fabric-Transmitter Interface Clocking for Example 2



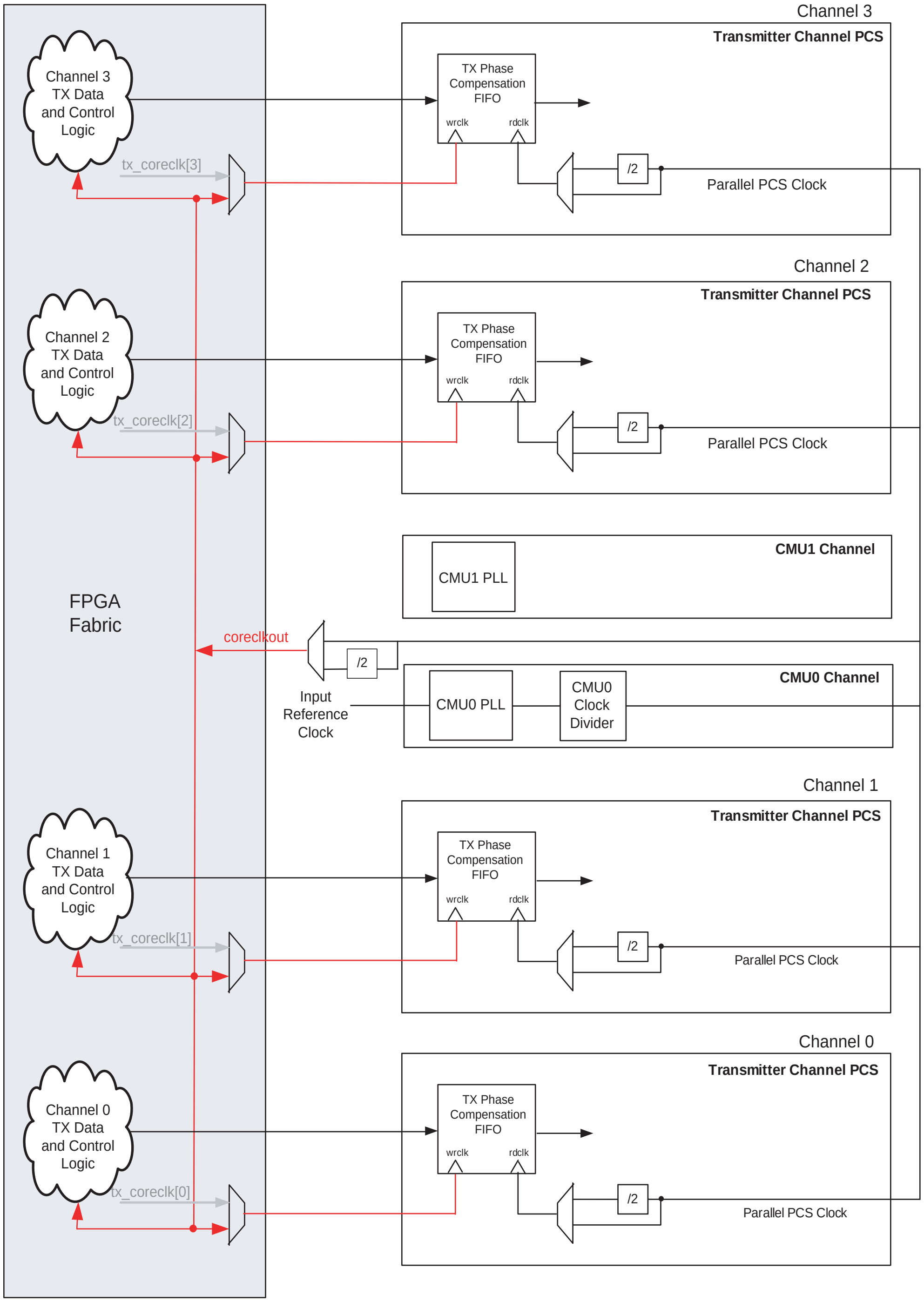
Bonded Channel Configuration

In ×4 bonded channel configuration, all four channels within the transceiver block are identical. The Quartus II software automatically drives the write port of the transmitter phase compensation FIFO in all four channels with the `coreclkout` signal. Use the `coreclkout` signal to clock the transmitter data and control logic for all four channels in the FPGA fabric.

In ×8 bonded channel configuration, all eight channels across two transceiver blocks are identical. The Quartus II software automatically drives the write port of the transmitter phase compensation FIFO in all eight channels with the `coreclkout` signal from the master transceiver block. Use the `coreclkout` signal to clock the transmitter data and control logic for all eight channels in the FPGA fabric.

Figure 2–19 shows the FPGA fabric-transmitter interface clocking in a ×4 bonded channel configuration.

Figure 2–19. FPGA Fabric-Transmitter Interface Clocking in a x4 Bonded Channel Configuration



Limitations of the Quartus II Software-Selected Transmitter Phase Compensation FIFO Write Clock

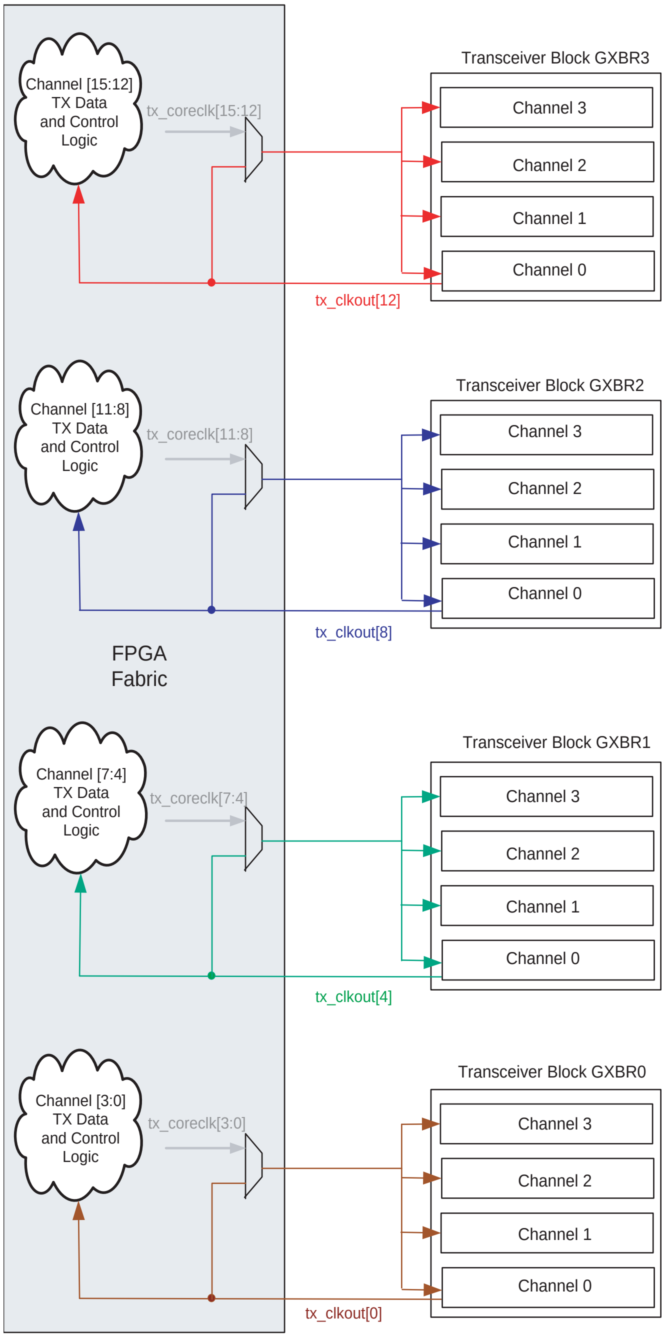
The Quartus II software uses a single `tx_clkout` signal to clock the transmitter phase compensation FIFO write port of all identical channels within a transceiver block. This results in one global and/or regional clock resource being used for each group of identical channels within a transceiver block.

For identical channels located across transceiver blocks, the Quartus II software does not use a single `tx_clkout` signal to clock the write port of the transmitter phase compensation FIFOs for all channels. It uses one `tx_clkout` signal for each group of identical channels per transceiver block. This results in higher global and/or regional clock resource utilization.

Example 3: Sixteen Identical Channels Across Four Transceiver Blocks

Consider 16 identical transmitter channels located across four transceiver blocks, as shown in Figure 2–20. The Quartus II software uses `tx_clkout` from Channel 0 in each transceiver block to clock the write port of the transmitter phase compensation FIFO in all four channels in that transceiver block. This results in four global and/or regional clocks resources being used, one for each transceiver block.

Figure 2–20. Sixteen Identical Channels Across Four Transceiver Blocks for Example 3



Because all 16 channels are identical, using a single `tx_clkout` to clock the transmitter phase compensation FIFO in all 16 channels results in only one global and/or regional clock resource being used instead of four. To achieve this, you must choose the transmitter phase compensation FIFO write clocks instead of the Quartus II software automatic selection, as discussed in "User-Selected Transmitter Phase Compensation FIFO Write Clock" on page 2–38.

User-Selected Transmitter Phase Compensation FIFO Write Clock

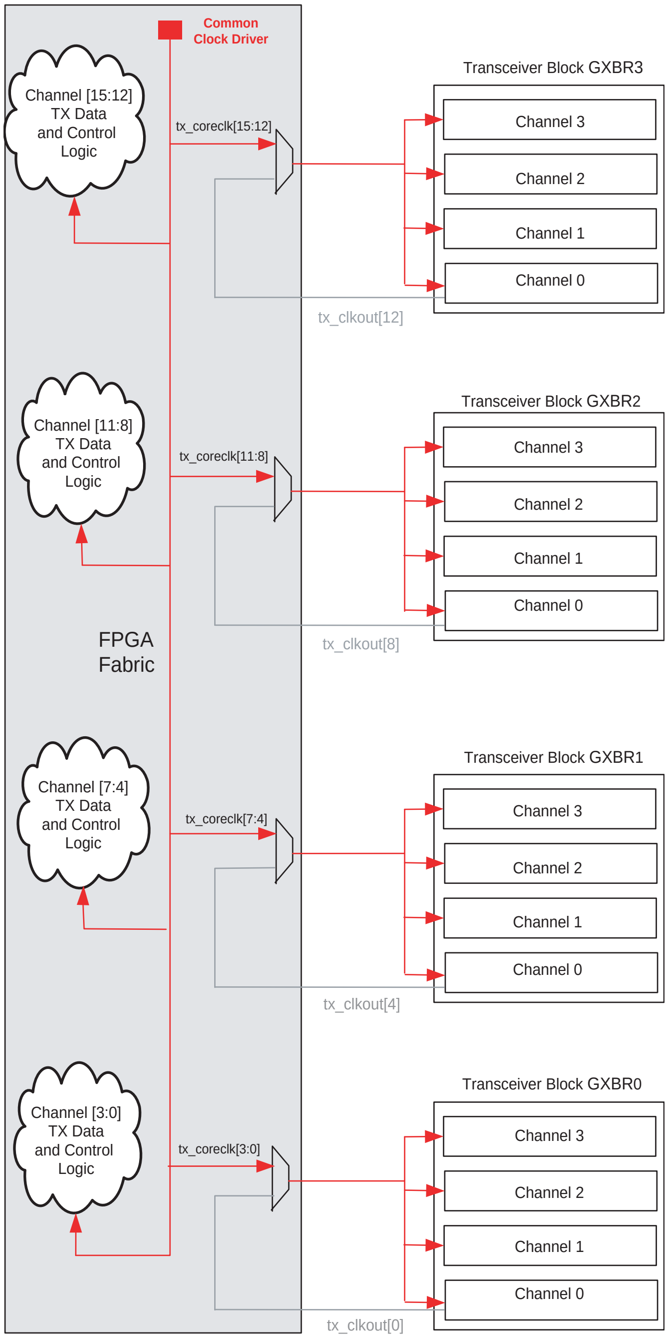
The ALTGX MegaWizard Plug-In Manager provides an optional port named `tx_coreclk` for each instantiated transmitter channel. If you enable this port, the Quartus II software does not automatically select the transmitter phase compensation FIFO write clock source. Instead, the signal that you drive on the `tx_coreclk` port of the channel clocks the write side of its transmitter phase compensation FIFO.

Use the flexibility of selecting the transmitter phase compensation FIFO write clock to reduce the global and/or regional clock resource utilization. You can connect the `tx_coreclk` ports of all identical channels in your design and drive them using a common clock driver that has 0 PPM frequency difference with respect to the FIFO read clocks of these channels. Use the common clock driver to clock the transmitter data and control logic in the FPGA fabric for all identical channels. This FPGA fabric-transceiver interface clocking scheme utilizes only one global and/or regional clock resource for all identical channels in your design.

Example 4: Sixteen Identical Channels Across Four Transceiver Blocks

Figure 2–21 shows 16 identical transmitter channels located across four transceiver blocks. The `tx_coreclk` ports of all 16 transmitter channels are connected together and driven by a common clock driver. This common clock driver also drives the transmitter data and control logic of all 16 transmitter channels in the FPGA fabric. Only one global and/or regional clock resource is used with this clocking scheme, compared to four global and/or regional clock resources needed without the `tx_coreclk` ports (the Quartus II software-selected transmitter phase compensation FIFO write clock).

Figure 2–21. Sixteen Identical Channels Across Four Transceiver Blocks for Example 4



Common Clock Driver Selection Rules

The common clock driver driving the `tx_coreclk` ports of all identical channels must have 0 PPM frequency difference with respect to the transmitter phase compensation FIFO read clocks of these channels. If there is any frequency difference between the FIFO write clock (`tx_coreclk`) and the FIFO read clock, the FIFO will overflow or under-run, resulting in corrupted data transfer between the FPGA fabric and the transmitter.

Table 2–10 shows the transmitter phase compensation FIFO read clocks that the Quartus II software selects in various configurations.

Table 2–10. Transmitter Phase Compensation FIFO Read Clocks

Configuration	Transmitter Phase Compensation FIFO Read Clock	
	Without Byte Serializer	**With Byte Serializer**
Non-Bonded Channel Configuration	Parallel transmitter PCS clock from the local clock divider in the associated channel (tx_clkout)	Divide-by-two version of the parallel transmitter PCS clock from the local clock divider in the associated channel (tx_clkout)
×4 Bonded Channel Configuration	Low-speed parallel clock from the CMU0 clock divider of the associated transceiver block (coreclkout)	Divide-by-two version of the low-speed parallel clock from the CMU0 clock divider of the associated transceiver block (coreclkout)
×8 Bonded Channel Configuration	Low-speed parallel clock from the CMU0 clock divider of the master transceiver block (coreclkout from master transceiver block)	Divide-by-two version of the low-speed parallel clock from the CMU0 clock divider of the master transceiver block (coreclkout from master transceiver block)

To ensure that you understand the 0 PPM clock driver rule, the Quartus II software expects the following set of user assignments whenever the tx_coreclk port is used to drive the transmitter phase compensation FIFO write clock:

- GXB 0 PPM Core Clock Setting

☞ Failing to make this assignment correctly when using the tx_coreclk port results in a Quartus II compilation error.

GXB 0 PPM Core Clock Setting

The GXB 0 PPM core clock setting is intended for advanced users who know the clocking configuration of the entire system and want to reduce the FPGA fabric global and regional clock resource utilization. The GXB 0 PPM core clock setting allows the following clock drivers to drive the tx_coreclk ports:

- tx_clkout in non-bonded channel configurations
- coreclkout in bonded channel configurations
- FPGA CLK input pins
- Transceiver REFCLK pins
- Clock output from left/right and top/bottom PLLs (PLL_L, PLL_R, and PLL_T, PLL_B)

☞ The Quartus II software does not allow gated clocks or clocks generated in FPGA logic to drive the tx_coreclk ports.

Because the GXB 0 PPM core clock setting allows FPGA CLK input pins and transceiver REFCLK pins as the clock driver, the Quartus II compiler cannot determine if there is a 0 PPM difference between the FIFO write clock and read clock for each channel.

☞ You must ensure that the clock driver for all connected tx_coreclk ports has a 0 PPM difference with respect to the FIFO read clock in those channels.

Table 2–11 shows the Quartus II assignments that you must make in the assignment editor.

Table 2–11. Quartus II Assignments

From:	Full design hierarchy name of one of the following clock drivers that you choose to drive the `tx_coreclk` ports of all identical channels *(1)*: ■ `tx_clkout` ■ `coreclkout` ■ FPGA CLK input pins ■ Transceiver `REFCLK` pins ■ Clock output from left and right or top and bottom PLLs
To:	`tx_dataout` pins of all identical channels whose `tx_coreclk` ports are connected together and driven by the 0 PPM clock driver.
Assignment Name:	GXB 0 PPM Core Clock Setting
Value:	ON

Note to Table 2–15:

(1) You can find the full hierarchy name of the 0 PPM clock driver using the **Node Finder** feature in the Quartus II Assignment Editor.

Example 5: Sixteen Identical Channels Across Four Transceiver Blocks

Figure 2–22 shows 16 identical transmitter channels located across four transceiver blocks. The `tx_coreclk` ports of all 16 transmitter channels are connected together and driven by the `tx_clkout[4]` signal from channel 0 in transceiver block GXBR1. The `tx_clkout[4]` signal also drives the transmitter data and control logic of all 16 transmitter channels in the FPGA fabric. Only one global clock resource is used by the `tx_clkout[4]` signal with this clocking scheme.

Figure 2–22. Sixteen Identical Channels Across Four Transceiver Blocks for Example 5

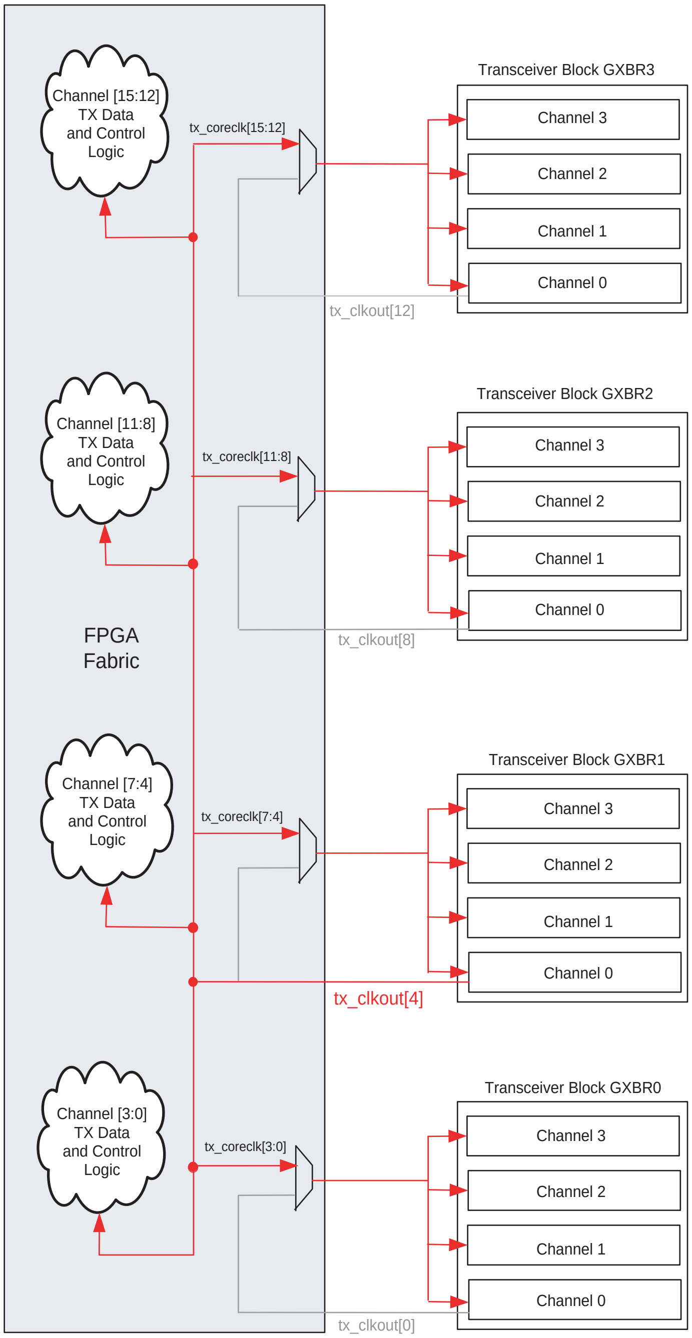


Table 2–12 shows the Quartus II assignments that you must make for the clocking scheme shown in Figure 2–22.

Table 2–12. Quartus II Assignments for Example 5 *(Note 1)* (Part 1 of 2)

From:	`top_level/top_xcvr_instance1/altgx_component/tx_clkout[4]` *(1)*
To:	`tx_dataout[15..0]`

Table 2–12. Quartus II Assignments for Example 5 *(Note 1)* (Part 2 of 2)

Assignment Name:	GXB 0 PPM Core Clock Setting
Value:	ON

Note to Table 2–15:

(1) This is an example design hierarchy path for the `tx_clkout[4]` signal.

FPGA Fabric-Receiver Interface Clocking

The receiver phase compensation FIFO compensates for the phase difference between the parallel receiver PCS clock (FIFO write clock) and the FPGA fabric clock (FIFO read clock). The receiver phase compensation FIFO read clock forms the FPGA fabric-receiver interface clock. The FIFO write clock and the read clock must have exactly the same frequency (0 PPM frequency difference).

Stratix IV GX transceivers provide the following two options for selecting the receiver phase compensation FIFO read clock:

- Quartus II software-selected receiver phase compensation FIFO read clock
- User-selected receiver phase compensation FIFO read clock

Quartus II Software-Selected Receiver Phase Compensation FIFO Read Clock

If you do not select the `rx_coreclk` port in the ALTGX MegaWizard Plug-In Manager, the Quartus II software automatically selects the receiver phase compensation FIFO read clock for each channel in that ALTGX instance. The Quartus II software selects the FIFO read clock depending on the channel configuration.

Non-Bonded Channel Configuration with Rate Matcher

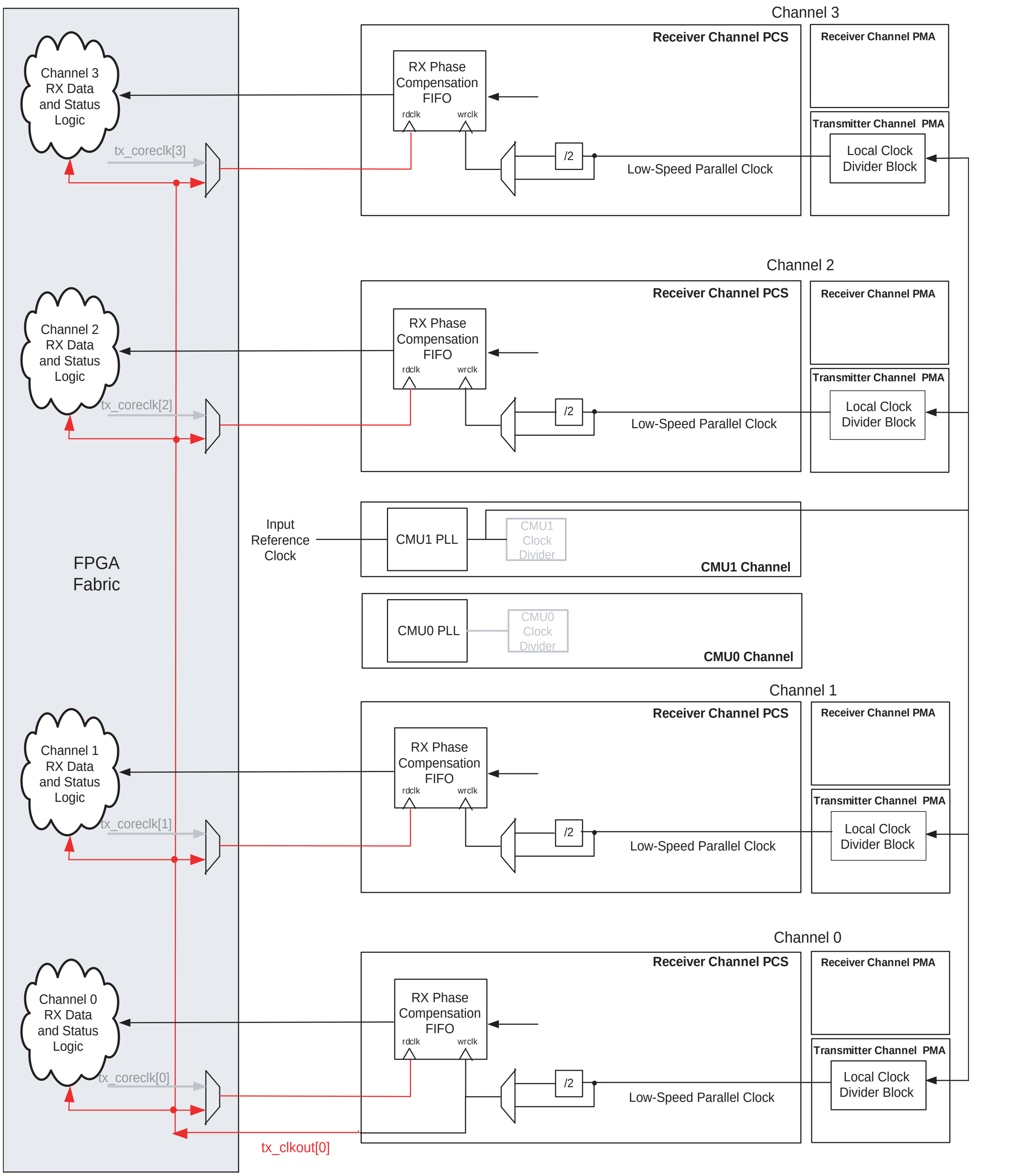
In non-bonded channel configuration, the transceiver channels may or may not be identical. Identical transceiver channels are defined as channels that have the same CMU PLL and receiver CDR input reference clock sources, have exactly the same CMU PLL and receiver CDR configuration, and have exactly the same PMA and PCS configuration.

Example 6: Four Identical Channels in a Transceiver Block

If all four channels within a transceiver block are identical, the Quartus II software automatically drives the read port of the receiver phase compensation FIFO in all four channels with `tx_clkout[0]`, as shown in Figure 2–23. Use the `tx_clkout[0]` signal to latch the receiver data and status signals from all four channels in the FPGA fabric.

☞ This configuration uses only one FPGA global and/or regional clock resource for `tx_clkout[0]`.

Figure 2–23. Four Identical Channels in a Transceiver Block for Example 6



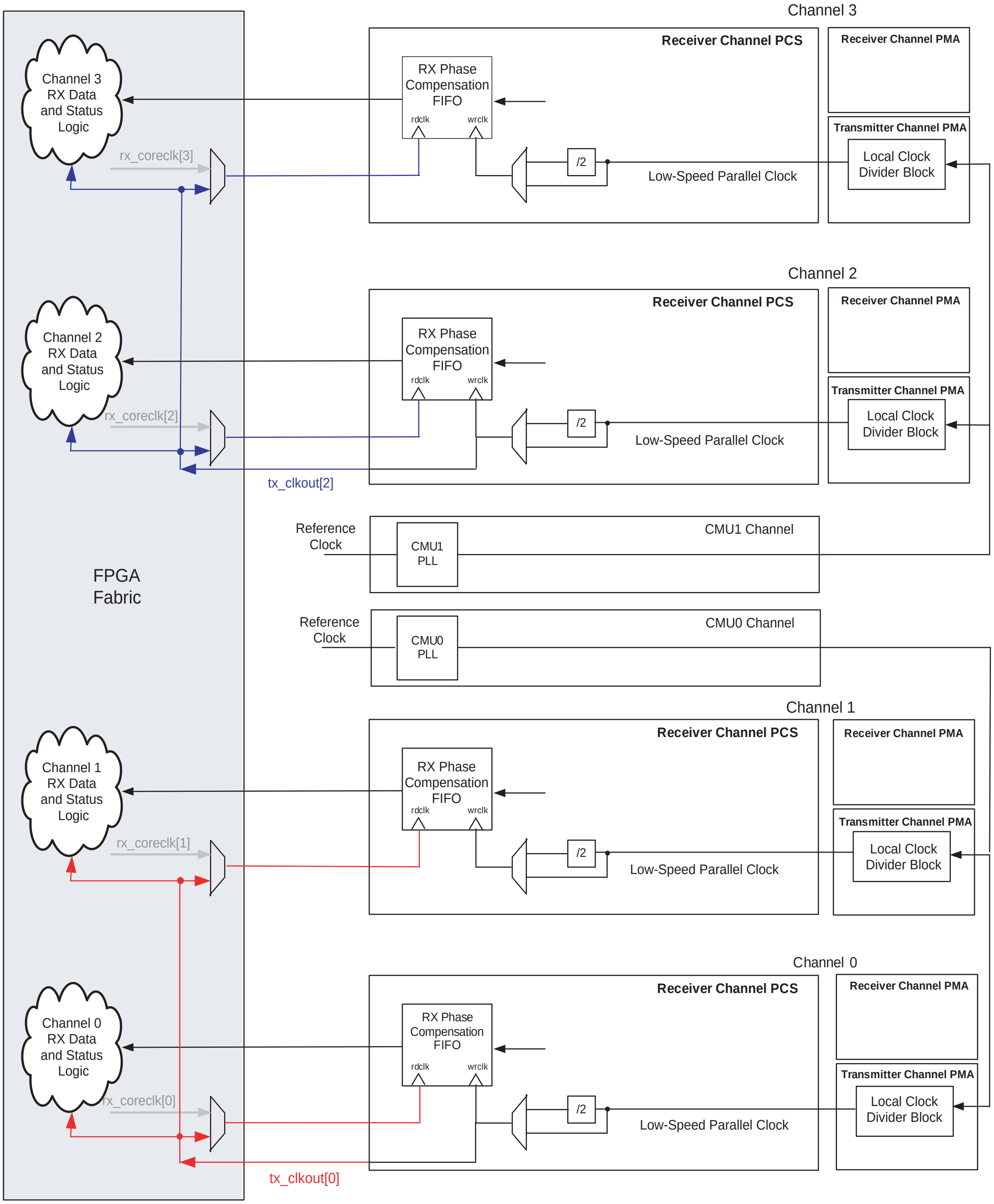
Example 7: Two Groups of Two Identical Channels in a Transceiver Block

Example 7 assumes channels 0 and 1, driven by the CMU0 PLL in a transceiver block, are identical. Also, channels 2 and 3, driven by the CMU1 PLL in the same transceiver block, are identical. In this case, the Quartus II software automatically drives the read port of the receiver phase compensation FIFO in channels 0 and 1 with the tx_clkout[0] signal. It also drives the read port of the receiver phase compensation FIFO in channels 2 and 3 with the tx_clkout[2] signal. Use the tx_clkout[0] signal to latch the receiver data and status signals from channels 0 and 1 in the FPGA fabric. Use the tx_clkout[2] signal to latch the receiver data and status signals from channels 2 and 3 in the FPGA fabric.

☞ This configuration uses two FPGA global and/or regional clock resources, one for the tx_clkout[0] signal and the other for the tx_clkout[2] signal.

Figure 2–24 shows the FPGA fabric-receiver interface clocking for Example 7.

Figure 2–24. FPGA Fabric-Receiver Interface Clocking for Example 7



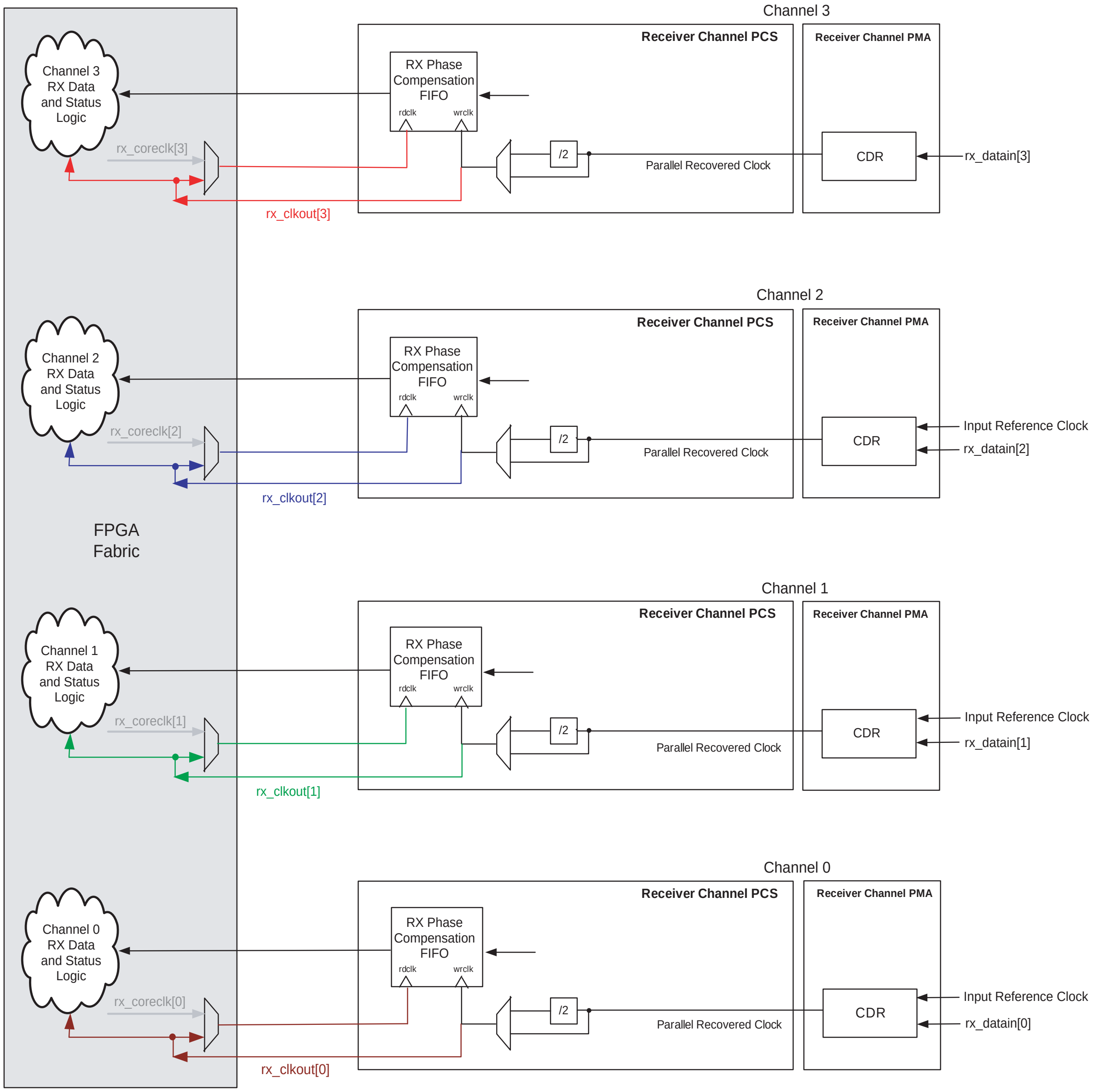
Non-Bonded Channel Configuration without Rate Matcher

In non-bonded channel configuration without rate matcher, the Quartus II software cannot determine if the incoming serial data in all channels have a 0 PPM frequency difference. The Quartus II software automatically drives the read port of the receiver phase compensation FIFO in each channel with the recovered clock driven on the `rx_clkout` port of that channel. Use the `rx_clkout` signal from each channel to latch its receiver data and status signals in the FPGA fabric.

☞ This configuration uses one FPGA global and/or regional clock resource per channel for the `rx_clkout` signal.

Figure 2–25 shows the FPGA fabric-receiver interface clocking for non-bonded channel configurations without rate matcher.

Figure 2–25. FPGA Fabric-Receiver Interface Clocking for Non-Bonded Channel Configurations without Rate Matcher



Bonded Channel Configuration

All bonded transceiver channel configurations have a rate matcher in the receiver data path.

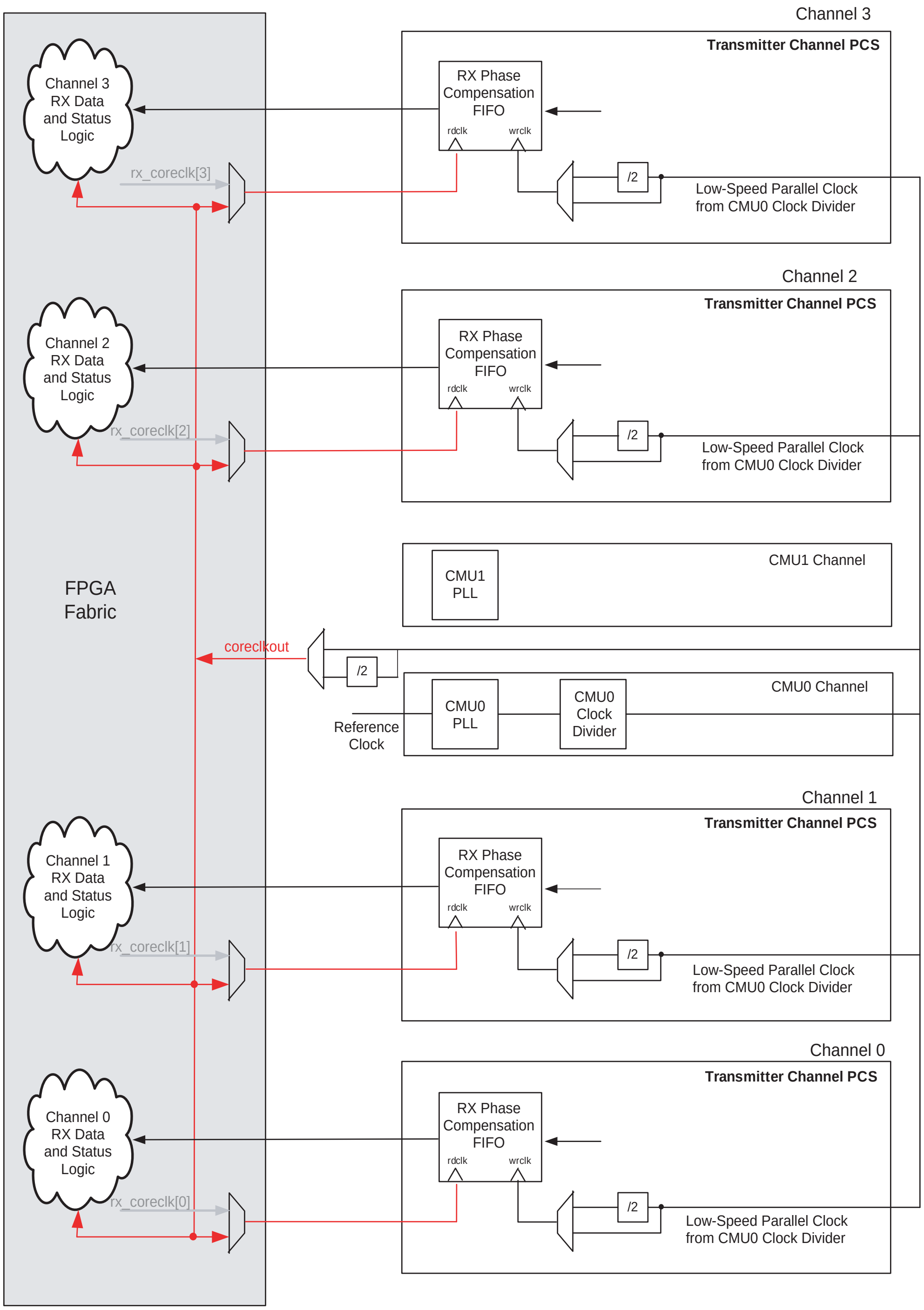
In ×4 bonded channel configurations, the Quartus II software automatically drives the read port of the receiver phase compensation FIFO in all four channels with the `coreclkout` signal. Use the `coreclkout` signal to latch the receiver data and status signals from all four channels in the FPGA fabric.

In ×8 bonded channel configurations, the Quartus II software automatically drives the read port of the receiver phase compensation FIFO in all eight channels with the `coreclkout` signal from the master transceiver block. Use the `coreclkout` signal to latch the receiver data and status signals from all eight channels in the FPGA fabric.

☞ This configuration uses one FPGA global and/or regional clock resource per bonded link for the `coreclkout` signal.

Figure 2–26 shows the FPGA fabric-receiver interface clocking in ×4 bonded channel configuration.

Figure 2–26. FPGA Fabric-Receiver Interface Clocking in a x4 Bonded Channel Configuration


Channel 3
Transmitter Channel PCS
Channel 3 RX Data and Status Logic
RX Phase Compensation FIFO
rdclk
wrclk
rx_coreclk[3]
/2
Low-Speed Parallel Clock from CMU0 Clock Divider
Channel 2
Transmitter Channel PCS
Channel 2 RX Data and Status Logic
RX Phase Compensation FIFO
rdclk
wrclk
rx_coreclk[2]
/2
Low-Speed Parallel Clock from CMU0 Clock Divider
CMU1 Channel
CMU1 PLL
FPGA Fabric
coreclkout
/2
CMU0 Channel
CMU0 PLL
CMU0 Clock Divider
Reference Clock
Channel 1
Transmitter Channel PCS
Channel 1 RX Data and Status Logic
RX Phase Compensation FIFO
rdclk
wrclk
rx_coreclk[1]
/2
Low-Speed Parallel Clock from CMU0 Clock Divider
Channel 0
Transmitter Channel PCS
Channel 0 RX Data and Status Logic
RX Phase Compensation FIFO
rdclk
wrclk
rx_coreclk[0]
/2
Low-Speed Parallel Clock from CMU0 Clock Divider

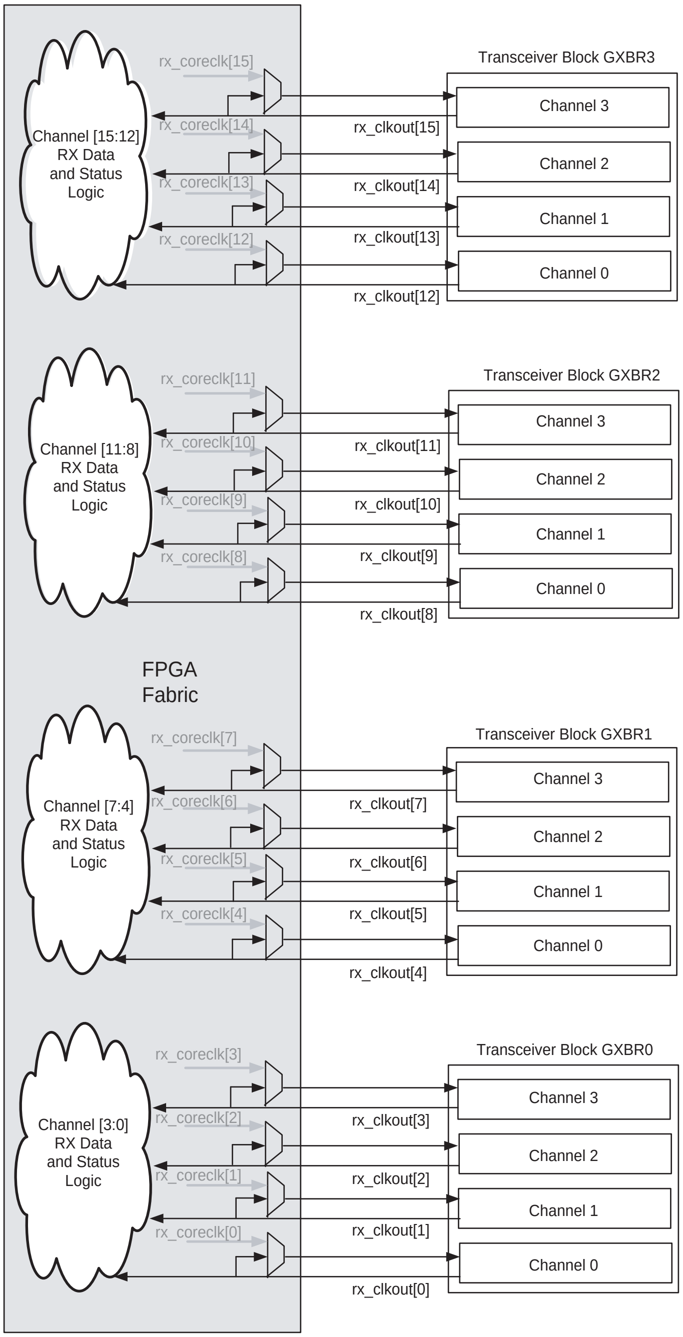
Limitations of Quartus II Software-Selected Receiver Phase Compensation FIFO Read Clock

In non-bonded channel configurations without rate matcher, the Quartus II software cannot determine if the incoming serial data in all channels have a 0 PPM frequency difference. The Quartus II software uses the recovered clock `rx_clkout` signal from each channel to clock the read port of its receiver phase compensation FIFO. This results in one global and/or regional clock resource being used per channel for the `rx_clkout` signal.

Example 8: Sixteen Channels Across Four Transceiver Blocks

Consider 16 non-bonded receiver channels without rate matcher located across four transceiver blocks, as shown in Figure 2–27. The incoming serial data to all 16 channels have a 0 PPM frequency difference with respect to each other. The Quartus II software uses `rx_clkout` from each channel to clock the read port of its receiver phase compensation FIFO. This results in 16 global and/or regional clocks resources being used, one for each channel.

Figure 2–27. Sixteen Non-Bonded Receiver Channels without Rate Match for Example 8



Since the recovered clock `rx_clkout` signals from all 16 channels have a 0 PPM frequency difference, you can use a single `rx_clkout` to clock the receiver phase compensation FIFO in all 16 channels. This results in only one global and/or regional clock resource being used instead of 16. To achieve this, you must select the receiver phase compensation FIFO read clocks instead of the Quartus II software automatic selection, as discussed in "User-Selected Receiver Phase Compensation FIFO Read Clock" on page 2–52.

User-Selected Receiver Phase Compensation FIFO Read Clock

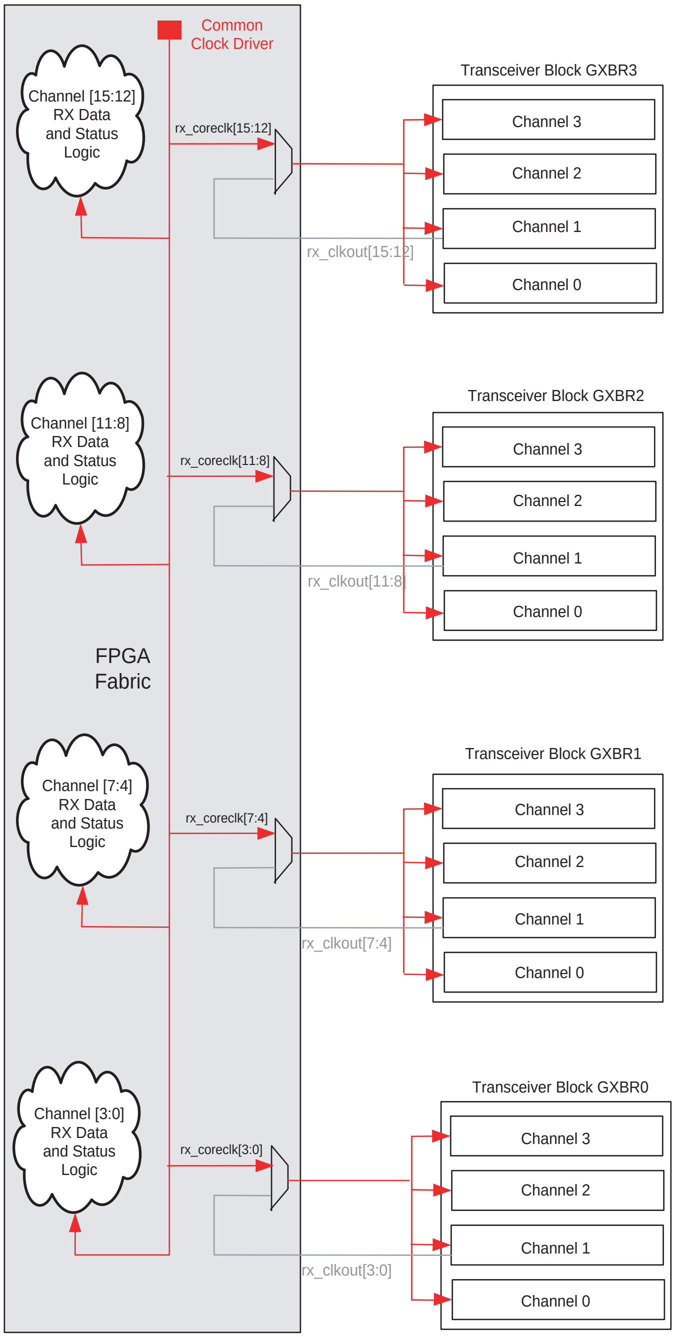
The ALTGX MegaWizard Plug-In Manager provides an optional port named `rx_coreclk` for each instantiated receiver channel. If you enable this port, the Quartus II software does not automatically select the receiver phase compensation FIFO read clock source. Instead, the signal that you drive on the `rx_coreclk` port of the channel clocks the read side of its receiver phase compensation FIFO.

You can use the flexibility of selecting the receiver phase compensation FIFO read clock to reduce the global and/or regional clock resource utilization. You can connect the `rx_coreclk` ports of all receiver channels in your design and drive them using a common clock driver that has a 0 PPM frequency difference with respect to the FIFO write clocks of these channels. Use this common clock driver to latch the receiver data and status signals in the FPGA fabric for these channels. This FPGA fabric-transceiver interface clocking scheme utilizes only one global and/or regional clock resource for all channels.

Example 9: Sixteen Identical Channels Across Four Transceiver Blocks

Figure 2–28 shows 16 channels located across four transceiver blocks. The incoming serial data to all 16 channels has a 0 PPM frequency difference with respect to each other. The `rx_coreclk` ports of all 16 channels are connected together and driven by a common clock driver. This common clock driver also latches the receiver data and status logic of all 16 receiver channels in the FPGA fabric. Only one global and/or regional clock resource is used with this clocking scheme, compared to 16 global and/or regional clock resources needed without the `rx_coreclk` ports (the Quartus II software-selected receiver phase compensation FIFO read clock).

Figure 2–28. Sixteen Identical Channels Across Four Transceiver Blocks for Example 9



Common Clock Driver Selection Rules

The common clock driver driving the `rx_coreclk` ports of all channels must have a 0 PPM frequency difference with respect to the receiver phase compensation FIFO write clocks of these channels. If there is any frequency difference between the FIFO read clock (`rx_coreclk`) and the FIFO write clock, the FIFO will overflow or under-run, resulting in corrupted data transfer between the FPGA fabric and the receiver.

Table 2–13 shows the receiver phase compensation FIFO write clocks that the Quartus II software selects in various configurations.

Table 2–13. Receiver Phase Compensation FIFO Write Clocks

Configuration	Receiver Phase Compensation FIFO Write Clock	
	Without Byte Serializer	**With Byte Serializer**
Non-Bonded Channel Configuration with rate matcher	Low-speed parallel clock from the local clock divider in the associated channel (`tx_clkout`)	Divide-by-two version of the low-speed parallel clock from the local clock divider in the associated channel (`tx_clkout`)
Non-Bonded Channel Configuration without rate matcher	Parallel recovered clock from the receiver PMA in the associated channel (`rx_clkout`)	Divide-by-two version of the parallel recovered clock from the receiver PMA in the associated channel (`rx_clkout`)
×4 Bonded Channel Configuration	Low-speed parallel clock from the `CMU0` clock divider of the associated transceiver block (`coreclkout`)	Divide-by-two version of the low-speed parallel clock from the `CMU0` clock divider of the associated transceiver block (`coreclkout`)
×8 Bonded Channel Configuration	Low-speed parallel clock from the `CMU0` clock divider of the master transceiver block (`coreclkout` from the master transceiver block)	Divide-by-two version of the low-speed parallel clock from the `CMU0` clock divider of the master transceiver block (`coreclkout` from the master transceiver block)

To ensure that you understand the 0 PPM clock driver rule, the Quartus II software expects the following set of user assignments whenever the `rx_coreclk` port is used to drive the receiver phase compensation FIFO read clock:

- GXB 0 PPM Core Clock Setting

☞ Failing to make this assignment correctly when using the `rx_coreclk` port results in a Quartus II compilation error.

GXB 0 PPM Core Clock Setting

The GXB 0 PPM core clock setting is intended for advanced users who know the clocking configuration of the entire system and want to reduce the FPGA fabric global and regional clock resource utilization. The GXB 0 PPM core clock setting allows the following clock drivers to drive the `rx_coreclk` ports:

- `tx_clkout` in non-bonded channel configurations with rate matcher
- `tx_clkout` and `rx_clkout` in non-bonded configurations without rate matcher
- `coreclkout` in bonded channel configurations
- FPGA CLK input pins
- Transceiver `REFCLK` pins
- Clock output from left/right and top/bottom PLLs (`PLL_L`, `PLL_R`, and `PLL_T`, `PLL_B`)

☞ The Quartus II software does not allow gated clocks or clocks generated in FPGA logic to drive the `tx_coreclk` ports.

Since the 0 PPM clock group assignment allows FPGA CLK input pins and transceiver REFCLK pins as the clock driver, the Quartus II compiler cannot determine if there is a 0 PPM difference between the FIFO write clock and read clock for each channel.

☞ You must ensure that the clock driver for all connected `rx_coreclk` ports has a 0 PPM difference with respect to the FIFO write clock in those channels.

Table 2–14 shows the Quartus II assignments that you must make.

Table 2–14. Quartus II Assignments

From:	Full design hierarchy name of one of the following clock drivers that you choose to drive the `rx_coreclk` ports of all identical channels *(1)*: ■ `tx_clkout` ■ `rx_clkout` ■ `coreclkout` ■ FPGA CLK input pins ■ Transceiver REFCLK pins ■ Clock output from left and right or top and bottom PLLs
To:	`rx_datain` pins of all channels whose `rx_coreclk` ports are connected together and driven by the 0 PPM clock driver.
Assignment Name:	GXB 0 PPM Core Clock Setting
Value:	ON

Note to Table 2–15:

(1) You can find the full hierarchy name of the 0 PPM clock driver using the Node Finder feature in the Quartus II Assignment Editor.

Example 10: Sixteen Channels Across Four Transceiver Blocks

Figure 2–29 shows 16 non-bonded channels without rate matcher located across four transceiver blocks. The incoming serial data to all 16 channels has a 0 PPM frequency difference with respect to each other. The `rx_coreclk` ports of all 16 channels are connected together and driven by `rx_clkout[9]` in transceiver block GXBR2. The `rx_clkout[9]` also clocks the receiver data and status signals of all 16 channels in the FPGA fabric. Only one global and/or regional clock resource is used by `rx_clkout[9]` with this clocking scheme.

Figure 2–29. Sixteen Channels Across Four Transceiver Blocks for Example 10

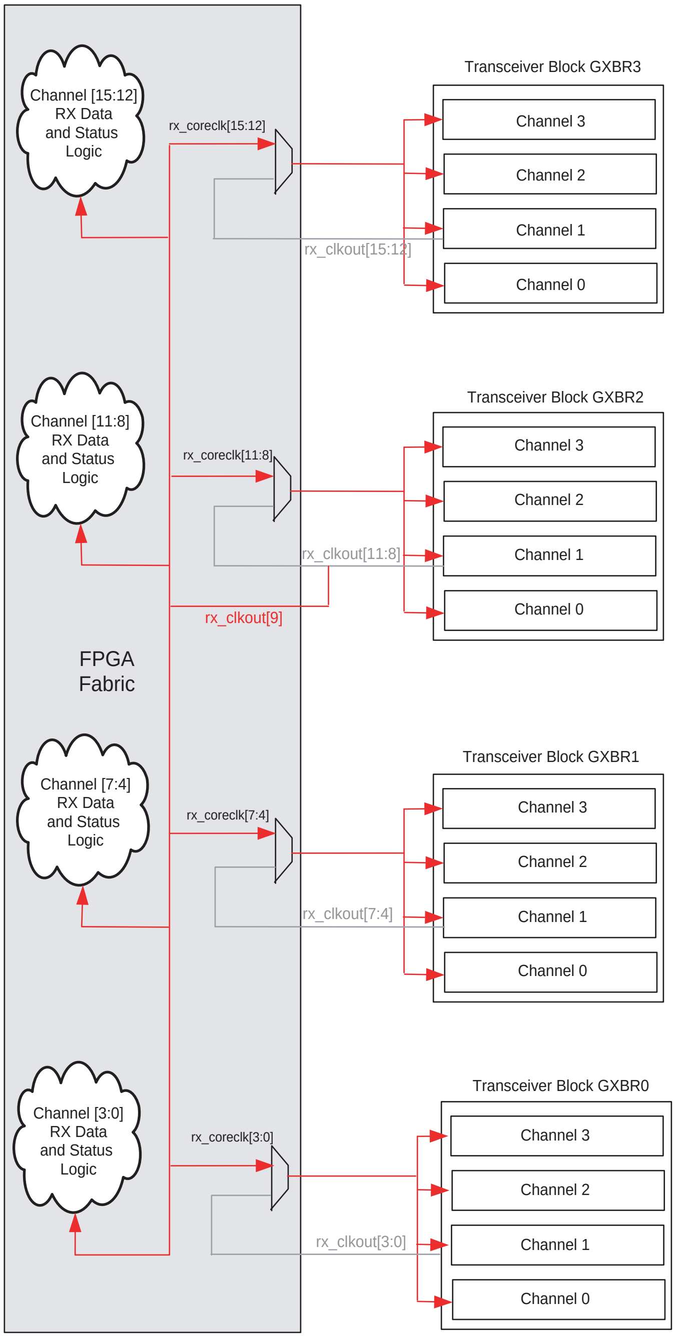


Table 2–15 shows the Quartus II assignments that you must make for the clocking scheme shown in Figure 2–29.

Table 2–15. Quartus II Assignments for Example 10 (Part 1 of 2)

From:	`top_level/top_xcvr_instance1/altgx_component/rx_clkout[9]` *(1)*
To:	`rx_datain[15..0]`

Table 2–15. Quartus II Assignments for Example 10 (Part 2 of 2)

Assignment Name:	GXB 0 PPM Core Clock Setting
Value:	ON

Note to Table 2–15:

(1) This is an example design hierarchy path for the `rx_clkout[9]` signal.

FPGA Fabric PLLs-Transceiver PLLs Cascading

The CMU PLL synthesizes the input reference clock to generate the high-speed serial clock used in the transmitter PMA. The receiver CDR synthesizes the input reference clock in lock-to-reference (LTR) mode to generate the high-speed serial clock.

This high-speed serial clock output from the CMU PLL and the receiver CDR runs at a frequency that is half the configured data rate. The CMU PLLs and the receiver CDRs only support multiplication factors (M) of 2, 4, 5, 8, 10, 16, 20, and 25. If you use an on-board crystal oscillator to provide the input reference clock through the dedicated `REFCLK` pins or ITB lines, the allowed crystal frequencies are limited by the CMU PLL and receiver CDR multiplication factors. The input reference clock frequencies are also limited by the allowed phase frequency detector (PFD) frequency range between 50 MHz and 325 MHz.

Example 11: Channel Configuration with 4 Gbps Data Rate

Consider a channel configured for 4 Gbps data rate. The high-speed serial clock output from the CMU PLL and the receiver CDR must run at 2 Gbps. Table 2–16 shows the allowed input reference clock frequencies for Example 11.

Table 2–16. Allowed Input Reference Clock Frequency for Example 11

Multiplication Factor (M)	On-Board Crystal Reference Clock Frequency (MHz)		Allowed
	With /N = 1	With /N = 2	
2	1000	2000	No. Violates the PFD frequency limit of 325 MHz.
4	500	1000	No. Violates the PFD frequency limit of 325 MHz.
5	400	800	No. Violates the PFD frequency limit of 325 MHz.
8	250	500	Yes
10	200	400	Yes
16	125	250	Yes
20	100	200	Yes
25	80	160	Yes

For a 4 Gbps data rate, the Quartus II software only allows an input reference clock frequency of 80, 100, 125, 160, 200, 250, 400, and 500 MHz. To overcome this limitation, Stratix IV GX devices allow the synthesized clock output from left and right PLLs in the FPGA fabric to drive the CMU PLL and receiver CDR input reference clock. The additional clock multiplication factors available in the left and right PLLs allow more options for on-board crystal oscillator frequencies.

Dedicated Left and Right PLL Cascade Lines Network

Stratix IV GX devices have a dedicated PLL cascade network on the left and right side of the device that connects to the input reference clock selection circuitry of the CMU PLLs and the receiver CDRs. The dedicated PLL cascade network on the left side of the device connects to the input reference clock selection circuitry of the CMU PLLs and receiver CDRs in transceiver blocks located on the left side of the device. The dedicated PLL cascade network on the right side of the device connects to the input reference clock selection circuitry of the CMU PLLs and receiver CDRs in transceiver blocks located on the right side of the device.

The dedicated PLL cascade networks are segmented by bidirectional tristate buffers located along the clock line. Segmentation of the dedicated PLL cascade network allows two left and right PLLs to drive the cascade clock line simultaneously to provide input reference clock to the CMU PLLs and receiver CDRs in different transceiver blocks.

Since the number of left and right PLLs and transceiver blocks vary from device to device, the capability of cascading a left and right PLL to the CMU PLLs and receiver CDRs also varies from device to device.

FPGA Fabric PLLs-Transceiver PLLs Cascading in the 780-Pin Package

The following Stratix IV GX devices are offered in the 780-pin package:

- EP4SGX70DF29
- EP4SGX110DF29
- EP4SGX230DF29
- EP4SGX290FH29
- EP4SGX360FH29

Stratix IV GX devices in 780-pin packages do not support FPGA fabric PLLs-Transceiver PLLs cascading.

FPGA Fabric PLLs-Transceiver PLLs Cascading in the 1152-Pin Package

The following Stratix IV GX devices are offered in the 1152-pin package:

- EP4SGX110FF35
- EP4SGX230FF35
- EP4SGX290FF35
- EP4SGX360FF35
- EP4SGX230HF35

- EP4SGX290HF35
- EP4SGX360HF35
- EP4SGX530HH35

Figure 2–30 shows the FPGA fabric PLLs-Transceiver PLLs cascading options allowed in the EP4SGX110FF35 device.

Figure 2–30. FPGA Fabric PLLs-Transceiver PLLs Cascading Options Allowed in the EP4SGX110FF35 Device

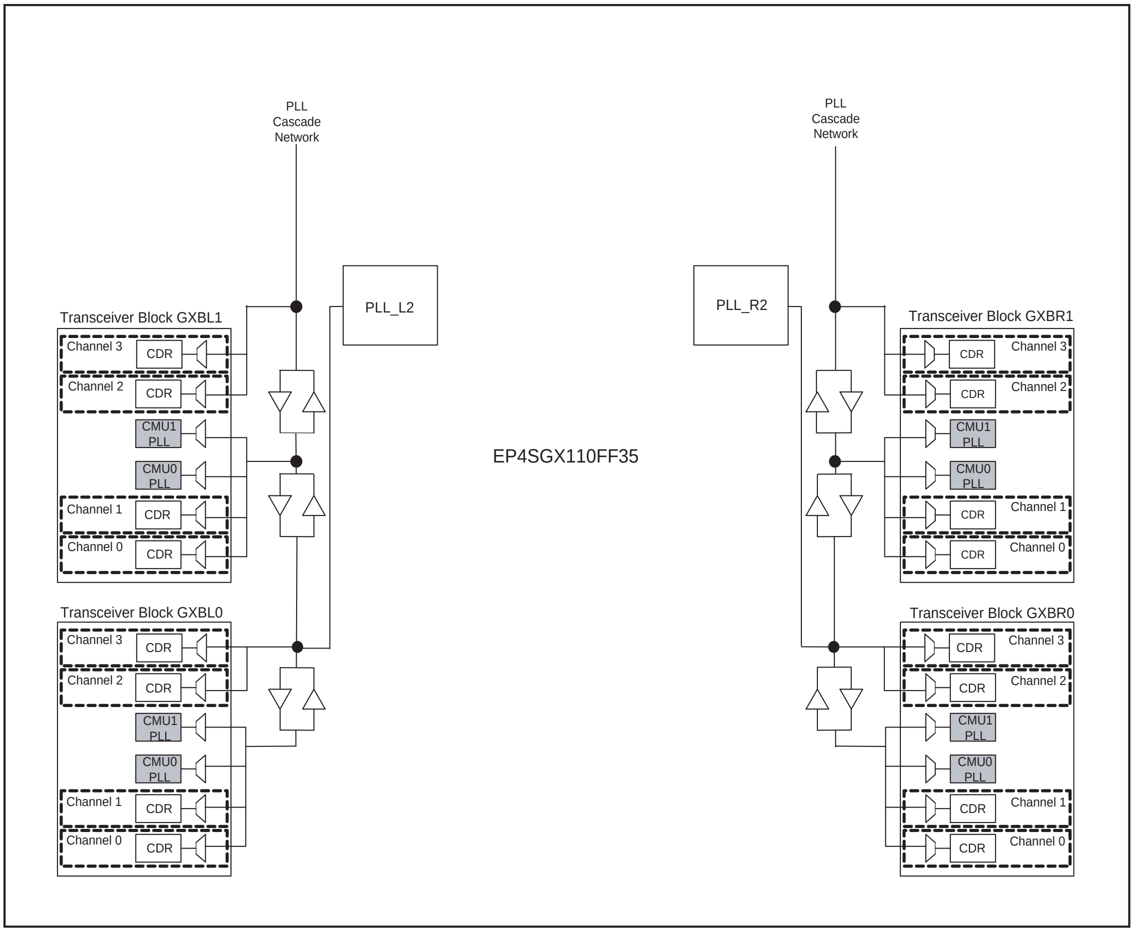


Figure 2–31 shows the FPGA fabric PLLs-Transceiver PLLs cascading options allowed in the EP4SGX230FF35, EP4SGX290FF35, and EP4SGX360FF35 devices.

Figure 2–31. FPGA Fabric PLLs-Transceiver PLLs Cascading Options Allowed in the EP4SGX230FF35, EP4SGX290FF35, and EP4SGX360FF35 Devices

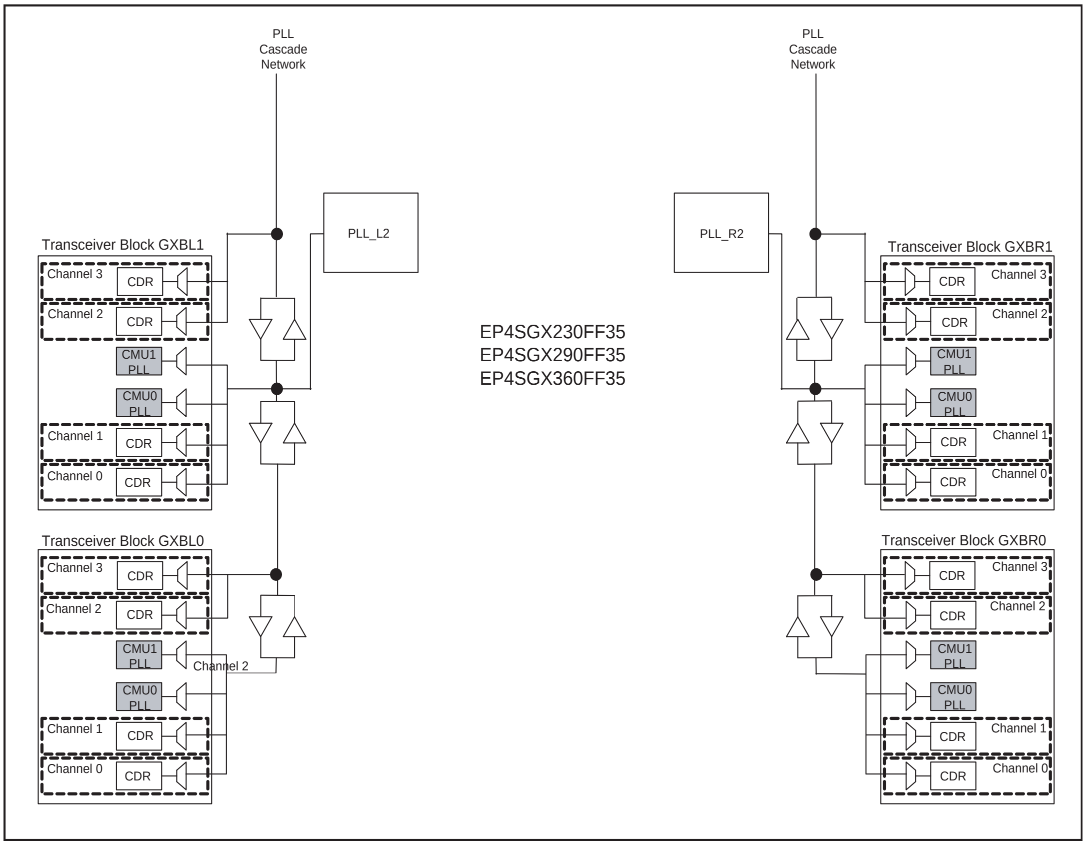


Figure 2–32 shows the FPGA fabric PLLs-Transceiver PLLs cascading options allowed in the EP4SGX230HF35, EP4SGX290HF35, and EP4SGX360HF35 devices.

Figure 2–32. FPGA Fabric PLLs-Transceiver PLLs Cascading Options Allowed in the EP4SGX230HF35, EP4SGX290HF35, and EP4SGX360HF35 Devices

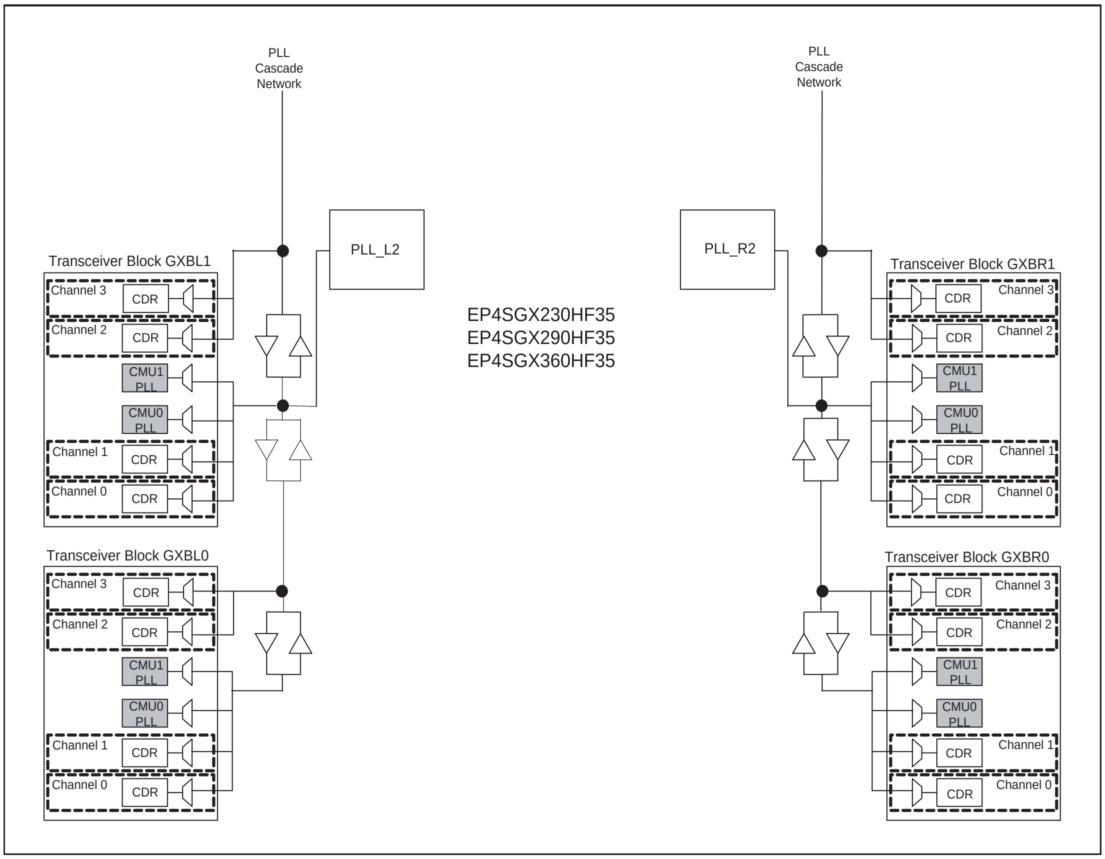
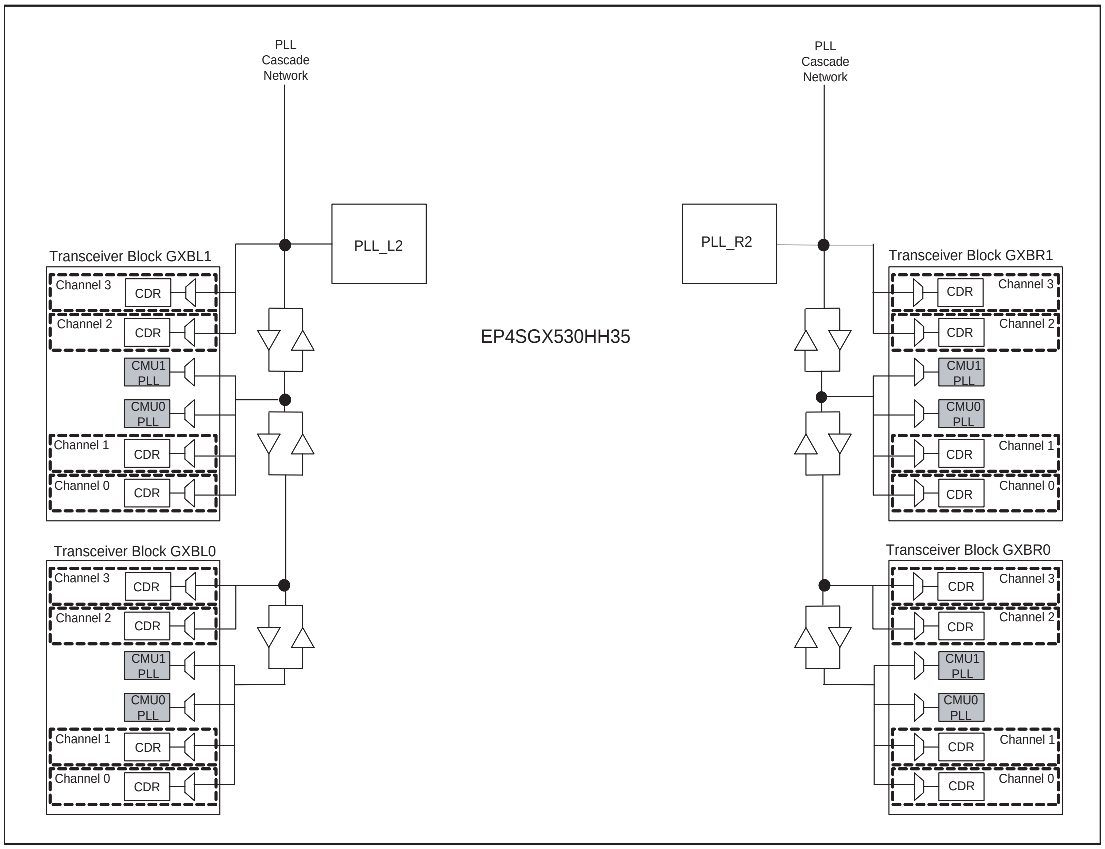


Figure 2–33 shows the FPGA fabric PLLs-Transceiver PLLs cascading options allowed in the EP4SGX530HH35 device.

Figure 2–33. FPGA Fabric PLLs-Transceiver PLLs Cascading Options Allowed in the EP4SGX530HH35 Device



FPGA Fabric PLLs-Transceiver PLLs Cascading in the 1517-Pin Package

The following Stratix IV GX devices are offered in the 1517-pin package:

- EP4SGX230KF40
- EP4SGX290KF40
- EP4SGX360KF40
- EP4SGX530KF40

Figure 2–34 shows the FPGA fabric PLLs-Transceiver PLLs cascading options allowed in the EP4SGX230KF40, EP4SGX290KF40, and EP4SGX360KF40 devices.

Figure 2–34. FPGA Fabric PLLs-Transceiver PLLs Cascading Options Allowed in the EP4SGX230KF40, EP4SGX290KF40, and EP4SGX360KF40 Devices

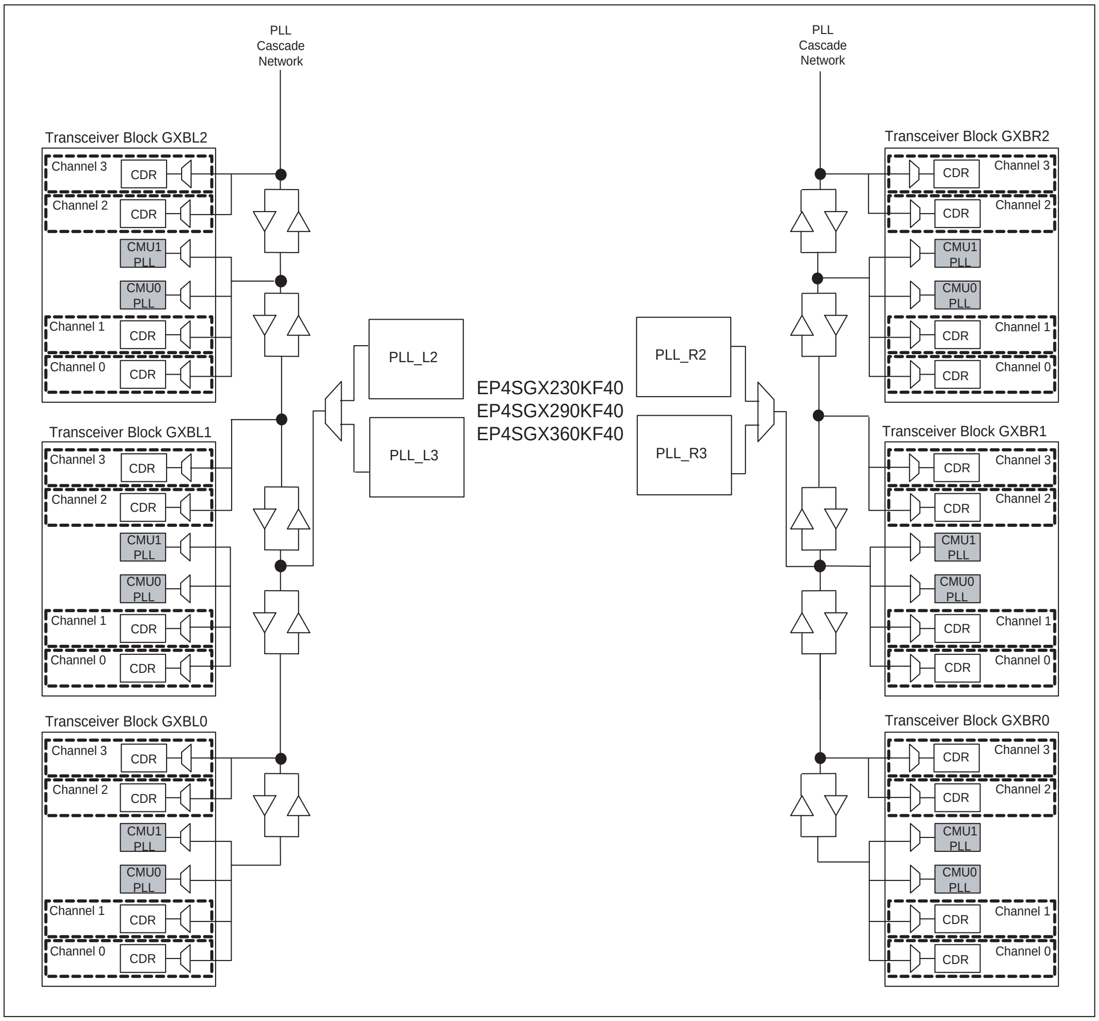
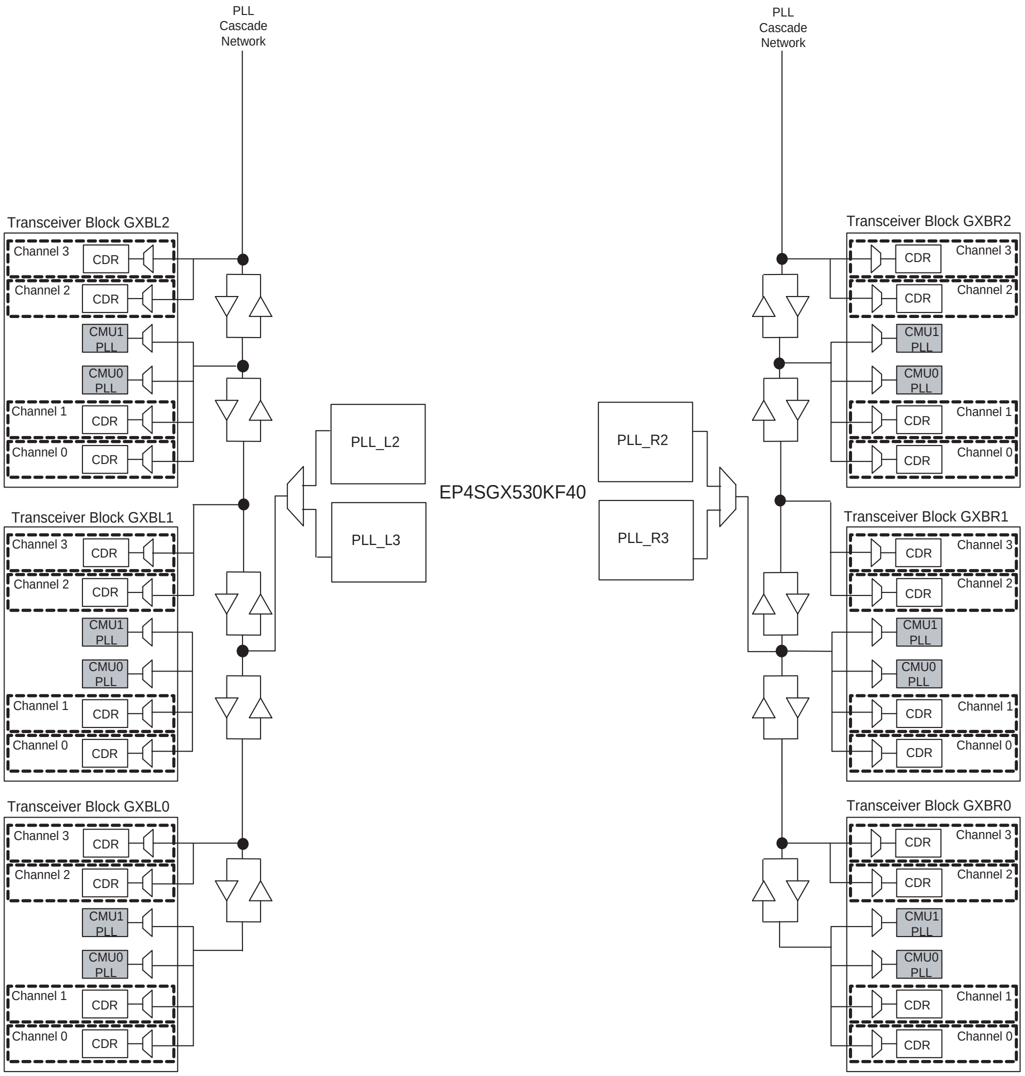


Figure 2–35 shows the FPGA fabric PLLs-Transceiver PLLs cascading options allowed in the EP4SGX530KF40 device.

Figure 2–35. FPGA Fabric PLLs-Transceiver PLLs Cascading Options Allowed in the EP4SGX530KF40 Device



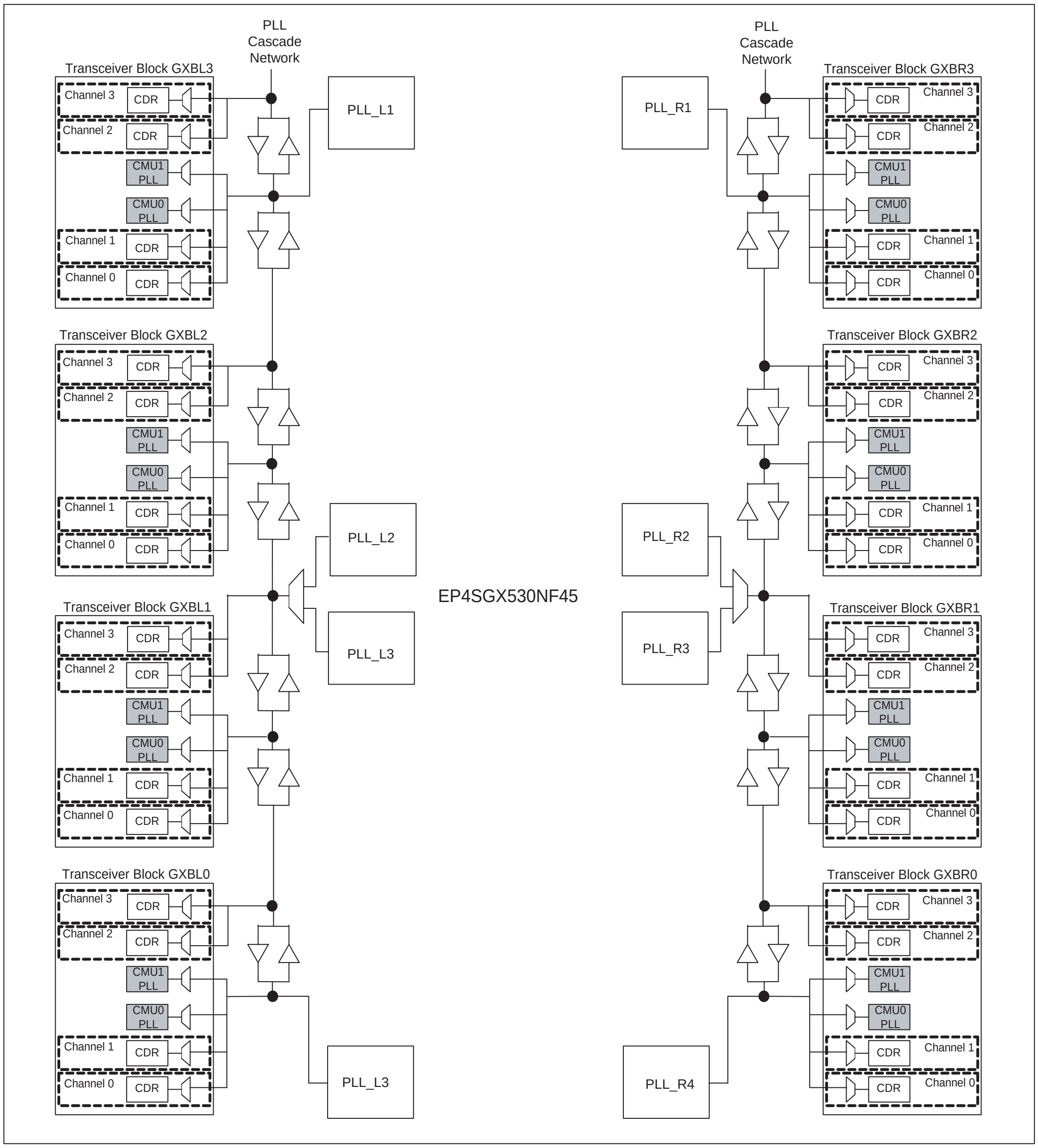
FPGA Fabric PLLs-Transceiver PLLs Cascading in the 1932-Pin Package

The following Stratix IV GX devices are offered in the 1932-pin package:

- EP4SGX530NF45

Figure 2–36 shows the FPGA fabric PLLs-Transceiver PLLs cascading options allowed in the EP4SGX530NF45 device.

Figure 2–36. FPGA Fabric PLLs-Transceiver PLLs Cascading Options Allowed in the EP4SGX530NF45 Device



FPGA Fabric PLLs-Transceiver PLLs Cascading Rules

You can only cascade the left PLLs (PLL_L1, PLL_L2, PLL_L3, and PLL_L4) to the transceiver blocks located on the left side of the device. Similarly, you can only cascade the right PLLs (PLL_R1, PLL_R2, PLL_R3, and PLL_R4) to the transceiver blocks located on the right side of the device.

The PLL cascade networks are single clock lines segmented by bidirectional tristate buffers located along the clock line. Segmentation of the PLL cascade network allows two left and right PLLs to drive the cascade clock line simultaneously to provide input reference clock to the CMU PLLs and receiver CDRs in different transceiver blocks. When cascading two or more FPGA fabric PLLs to the CMU PLLs and receiver CDRs, there must be no crossover in the cascaded clock paths on the PLL cascade network.

Example 12: Design Target—EP4SGX530NF45 Device

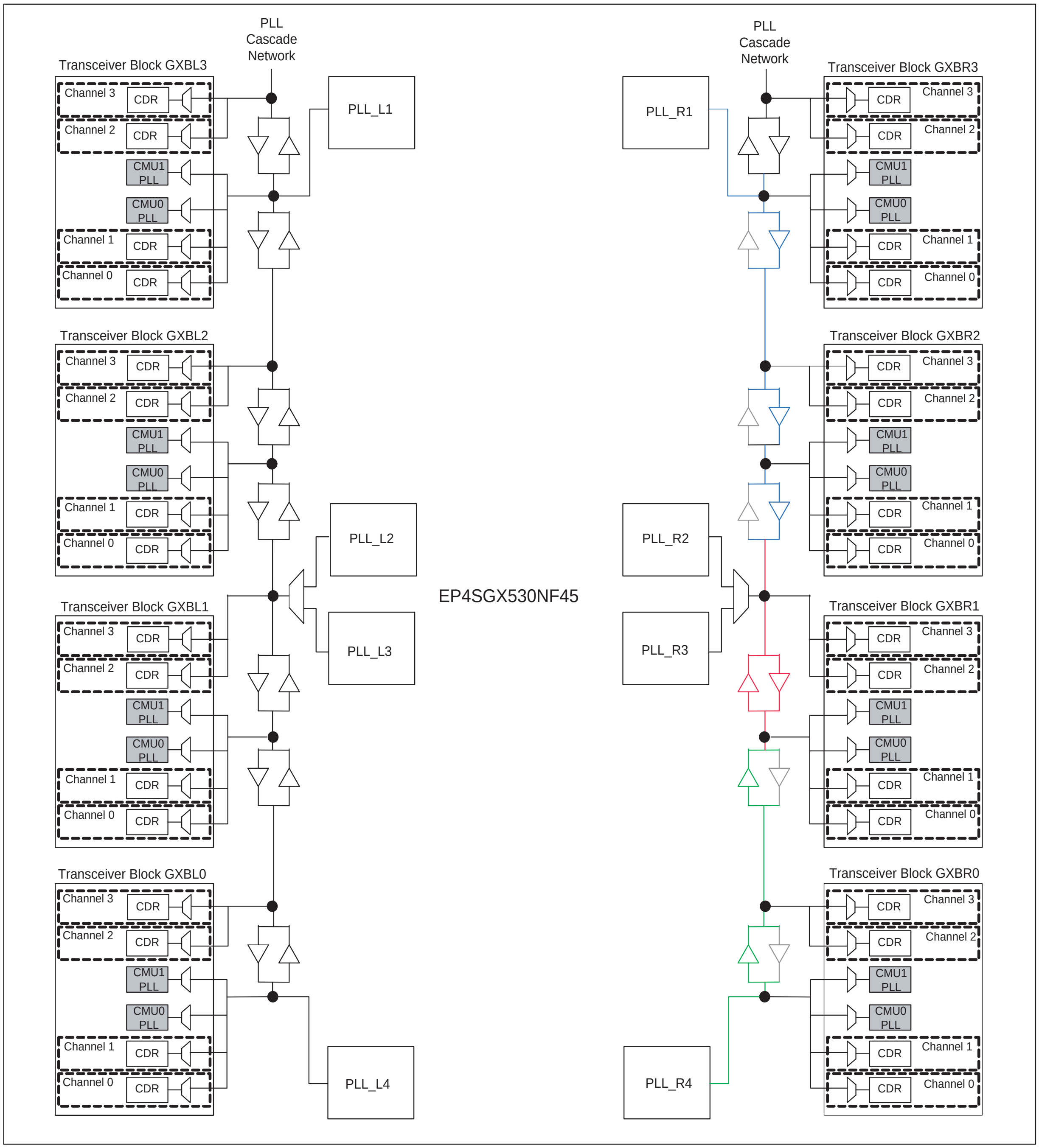
Consider that your design targeted toward the EP4SGX530NF45 device requires providing input reference clocks to the following CMU PLLs and receiver CDRs from two right PLLs in the FPGA fabric:

- `CMU0` PLL in Transceiver Block GXBR1
- Receiver CDRs in channel 2 and channel 3 in Transceiver Block GXBR1

Case 1: `PLL_R4` is used to provide the input reference clock to the receiver CDRs in channel 2 and channel 3 (shown in GREEN) and `PLL_R1` is used to provide the input reference clock to the `CMU0` PLL (shown in BLUE) in transceiver block GXBR1.

Figure 2–37 shows that this FPGA fabric-Transceiver PLL cascading configuration is illegal due to crossover (shown in RED) of cascade clock paths on the PLL cascade network.

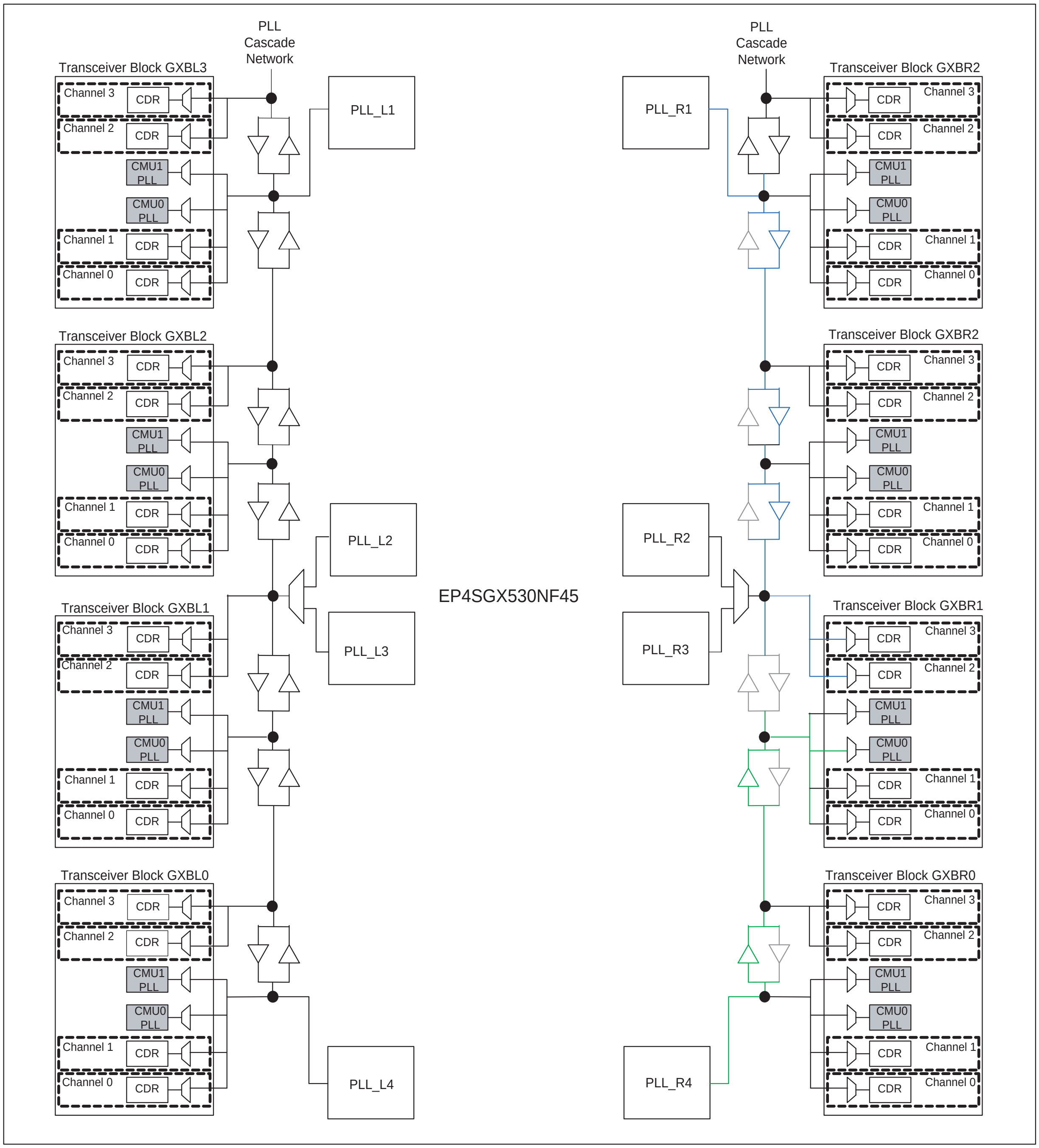
Figure 2–37. Illegal FPGA Fabric-Transceiver PLL Cascading Configuration



Case 2: PLL_R1 is used to provide input reference clock to the receiver CDRs in channel 2 and channel 3 (shown in BLUE) and PLL_R4 is used to provide input reference clock to the CMU0 PLL (shown in GREEN) in transceiver block GXBR1.

Figure 2–38 shows that this FPGA fabric-Transceiver PLL cascading configuration is legal as there is no crossover of the cascade clock paths on the PLL cascade network.

Figure 2–38. Legal FPGA Fabric-Transceiver PLL Cascading Configuration



Referenced Documents

This chapter references the following documents:

- *Clock Networks and PLLs in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook*
- *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook*

Document Revision History

Table 2–17 shows the revision history for this document.

Table 2–17. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	Initial release.	—
May 2008 v1.0	Initial release.	—

3. Configuring Multiple Protocols and Data Rates in a Transceiver Block

SIV52003-2.0

Overview

Each transceiver channel in a Stratix® IV GX device can run at an independent data rate or in an independent protocol mode. Within each transceiver channel, the transmitter and receiver channels can run at different data rates. Each transceiver block consists of two clock multiplier unit phase-locked loops (CMU PLLs) that provide clocks to all the transmitter channels within the transceiver block. Each receiver channel contains a dedicated clock data recovery (CDR) unit.

This chapter includes the following sections:

- "Creating Transceiver Channel Instances" on page 3–1
- "General Requirements to Combine Channels" on page 3–2
- "Sharing CMU PLLs" on page 3–4
- "Combining Receiver Only Channels" on page 3–12
- "Combining Transmitter Channel and Receiver Channel Instances" on page 3–12
- "Combining Channels Configured in Protocol Functional Modes" on page 3–15
- "Combining Transceiver Instances Using PLL Cascade Clocks" on page 3–19
- "Combining Transceiver Instances in Multiple Transceiver Blocks" on page 3–19

You can configure each transmitter channel to use one of the two CMU PLLs in the transceiver block. In addition, each transmitter channel has a local divider (/1, /2, or /4) that divides the high-clock output of the CMU PLL to provide high-speed serial and low-speed parallel clocks for its physical coding sublayer (PCS) and physical medium attachment (PMA) functional blocks.

You can configure the RX CDR present in the receiver channel to a distinct data rate and provide separate input reference clocks. Each receiver channel also contains a local divider that divides the high-speed clock output of the RX CDR and provides clocks for its PCS and PMA functional blocks. To enable transceiver channel settings, the Quartus® II software provides the ALTGX MegaWizard® Plug-In Manager interface. The ALTGX MegaWizard Plug-In Manager allows you to instantiate a single transceiver channel or multiple transceiver channels in **Receiver and Transmitter**, **Receiver only**, and **Transmitter only** configurations.

Creating Transceiver Channel Instances

The two ways you can instantiate multiple transceiver channels in the **General** screen of the ALTGX MegaWizard Plug-In Manager are:

- In the **What is the number of channels?** option, select the required value. This method creates all the transceiver channels with identical configurations. Refer to "Combining Transceiver Instances in Multiple Transceiver Blocks" on page 3–19 for examples.

- In the **What is the number of channels?** option, select 1 and create a single channel transceiver instance. To instantiate additional transceiver channels with an identical configuration, select the created ALTGX instance multiple times. If you need additional transceiver channels with different configurations, create separate ALTGX megafunction instances with different settings and use them in your design.

When you create instances using the above methods, you can force the placement of up to four transceiver channels within the same transceiver block. This can be done by assigning the `tx_dataout` and `rx_datain` ports of the channel instances to a single transceiver bank. If you do not assign pins to the `tx_dataout` and `rx_datain` ports, the Quartus II software chooses default pin assignments. When you compile the design, the Quartus II software combines multiple channel instances within the same transceiver block if the instances meet specific requirements. The following sections explain these requirements for different transceiver configurations.

General Requirements to Combine Channels

When you create multiple ALTGX instances, the Quartus II software requires you to set identical values for the following parameters and signals to combine the ALTGX instances within the same transceiver block or in transceiver blocks on the same side of the device. The following sections discuss these requirements.

Transmitter Buffer Voltage (V_{CCH})

The Stratix IV GX device provides you the option to select 1.4 V or 1.5 V for the V_{CCH} supply through the ALTGX MegaWizard Plug-In Manager. To combine the channel instances within the same transceiver block, the Quartus II software requires you to set the same V_{CCH} value in all the channel instances.

Note that if you select the 1.5 V option for a transceiver instance, the data rate range of all the transceiver channels within a transceiver block is restricted to between 600 Mbps and 4.25 Gbps.

Transceiver Analog Power ($V_{CCA\_L/R}$)

The Stratix IV GX device contains two different power supply pins, `VCCA_L` and `VCCA_R` that provide power to the PMA blocks in all the transceiver channels on the left and right sides of the device, respectively.

The Stratix IV GX device provides you the option to select 2.5 V or 3.0 V for the `VCCA_L/R` supply through the ALTGX MegaWizard Plug-In Manager. You must set the same `VCCA_L/R` value for all the transceiver channel instances to enable the Quartus II software to place them in the transceiver blocks on the same side of the device. For example, if you have two ALTGX instances that you would like to place on the left side transceiver banks GXBL0 and GXBL1, the `VCCA_L/R` values in the two ALTGX instances must be the same.

☞ Note that if you select the 2.5 V option on a side of the device, the data rate range of the CMU PLLs on the selected side is restricted to between 600 Mbps and 4.25 Gbps.

Control Signals

This section contains information about the gxb_powerdown, reconfig_fromgxb, and reconfig_togxb port.

gxb_powerdown Port

The gxb_powerdown port is an optional port that you can enable in the ALTGX MegaWizard Plug-In Manager. If enabled, you must drive the gxb_powerdown port in the ALTGX instances from the same logic or the same input pin to enable the Quartus II software to assign them in the same transceiver block.

reconfig_fromgxb and reconfig_togxb Ports

If you configure a transceiver channel in **Receiver only or transmitter** and **Receiver** configuration, the ALTGX Megawizard Plug-In Manager provides the reconfig_fromgxb and reconfig_togxb ports. These ports must be connected to the dynamic reconfiguration controller to perform offset cancellation on the receiver channels. These ports are also available if you select the **Analog Controls** option in the **Reconfig** screen to dynamically control the PMA settings such as the V_{OD}, pre-emphasis, equalization, or DC gain of a transceiver channel.

If these ports are present in one transceiver instance, you must also enable these ports in the other transceiver instances to combine them within the same transceiver block. Doing this enables the Quartus II software to combine these channel instances within the same transceiver block.

For further information about connecting these ports to the dynamic reconfiguration controller, refer to the “Connecting reconfig_fromgxb and reconfig_togxb Ports” section of the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*.

Calibration Clock and Power Down

Each calibration block in a Stratix IV GX device is shared by multiple transceiver blocks.

If your design uses multiple transceiver blocks, depending on the transceiver banks selected, you must connect the cal_blk_clk and cal_blk_powerdown ports of all channel instances to the same input pin or logic.

For more information about the calibration block and transceiver banks that are connected to a specific calibration block, refer to the “Calibration Block” section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

Asserting the cal_blk_powerdown port affects calibration on all transceiver channels connected to the calibration block.

Sharing CMU PLLs

Each Stratix IV GX transceiver block contains two CMU PLLs. When you create multiple transceiver channel instances and intend to combine them in the same transceiver block, the Quartus II software checks whether a single CMU PLL can be used to provide clock outputs for the transmitter side of the channel instances. If a single CMU PLL is not sufficient, the Quartus II software attempts to combine the channel instances using two CMU PLLs. Otherwise, the Quartus II software issues a Fitter error.

The following two sections discuss the ALTGX instance requirements to enable the Quartus II software to share the CMU PLL.

☞ Only the channels combined within the same transceiver block can share the two CMU PLLs available in a transceiver block.

Multiple Channels Sharing a CMU PLL

To enable the Quartus II software to share the same CMU PLL for multiple channels, the following parameters in the channel instantiations must be identical:

- "Base data rate" (the CMU PLL is configured for this data rate)
- CMU PLL bandwidth setting
- Reference clock frequency
- Input reference clock pin
- `pll_powerdown` port of the ALTGX instances must be driven from the same logic

Each channel instance can have a different local divider setting. This is a useful option when you intend to run each channel within the transceiver block at different data rates that are derived from the same base data rate using the local divider values /1, /2, and /4. This is shown in the following example.

Example 1

Consider an example design with four instances of **Receiver and Transmitter** configuration in the same transceiver block at various serial data rates. Assume that each instance contains a channel and is driven from the same clock source and has the same CMU PLL bandwidth settings. Table 3–1 shows the configuration for Example 1.

Table 3–1. Configuration for Example 1 (Part 1 of 2)

User-Created Instance Name	ALTGX MegaWizard Plug-In Manager Settings		
	Number of Channels	Configuration	Effective Data Rate
inst0	1	Receiver and Transmitter	4.25 Gbps
inst1	1	Receiver and Transmitter	2.125 Gbps

Table 3–1. Configuration for Example 1 (Part 2 of 2)

User-Created Instance Name	ALTGX MegaWizard Plug-In Manager Settings		
	Number of Channels	Configuration	Effective Data Rate
Inst2	1	Receiver and Transmitter	1.0625 Gbps
Inst3	1	Receiver and Transmitter	4.25 Gbps

For Example 1, you can share a single CMU PLL for all four channels because:

- One CMU PLL can be can be configured to run at 4.25 Gbps.
- Each channel can divide the CMU PLL clock output using the local divider and achieve the required data rates of 4.25 Gbps, 2.125 Gbps, and 1.0625 Gbps. Since each receiver channel has a dedicated CDR, the receiver side in each instance can be setup for these three data rates without any restrictions.

The following steps show you how to achieve the configuration.

To enable the Quartus II software to share a single CMU PLL for all four channels, set the following values in the **General** screen of the ALTGX MegaWizard Plug-In Manager.

- For inst0:
  - Set **What is the effective data rate?** to 4.25 Gbps
  - Set **Specify base data rate** to 4.25 Gbps
- For inst1:
  - Set **What is the effective data rate?** to 2.125 Gbps
  - Set **Specify base data rate** to 4.25 Gbps
- For inst2:
  - Set **What is the effective data rate?** to 1.0625 Gbps
  - Set **Specify base data rate** to 4.25 Gbps
- For inst3:
  - Set **What is the effective data rate?** to 4.25 Gbps
  - Set **Specify base data rate** to 4.25 Gbps

☞ The **Specify base data rate** option is 4.25 Gbps for all four instances. Given that the CMU PLL bandwidth setting and input reference clock are the same and that the `pll_powerdown` ports are driven from the same logic or pin, the Quartus II software shares a single CMU PLL that runs at 4.25 Gbps.

You can force the placement of transceiver channels to a specific transceiver block by assigning pins to `tx_dataout` and `rx_datain`. Otherwise, the Quartus II software selects a transceiver bank.

Figure 3–1 and Figure 3–2 show the scenario before and after the Quartus II software combines the transceiver channel instances. Since the RX CDR is not shared between channels, only the CMU PLL is shown.

☞ Each of the ALTGX instances has a `pll_powerdown` port. You must drive the `pll_powerdown` ports for all the instances from the same logic to enable the Quartus II software to share the same CMU PLL. If you drive the `pll_powerdown` ports of the ALTGX instance using different logic, the Quartus II software does not use the same CMU PLL even if all the other required parameters of all the ALTGX instances are identical.

Figure 3–1. ALTGX Instances before Compilation for Example 1

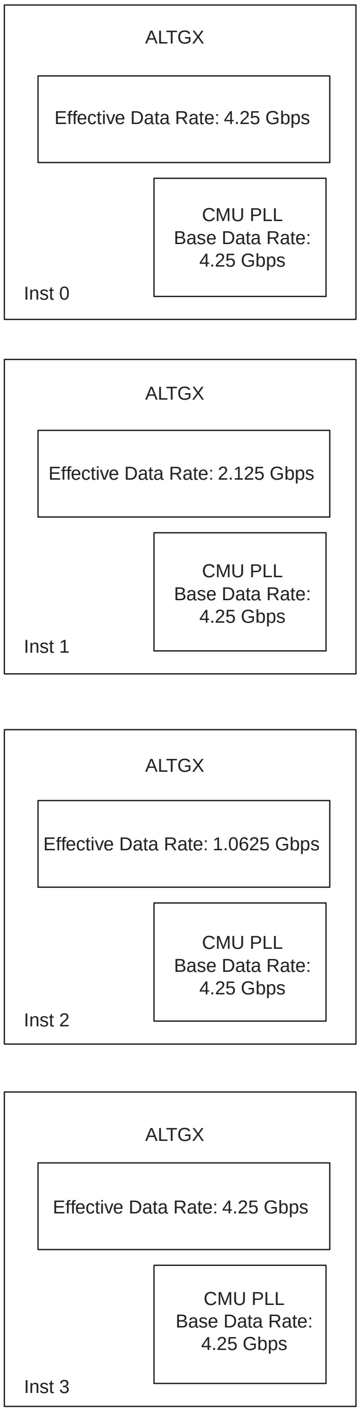
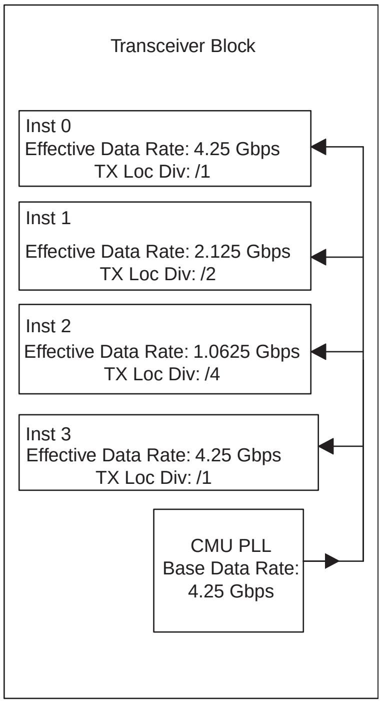


Figure 3–2. Combined Instances after Compilation for Example 1



Multiple Channels Sharing Two CMU PLLs

In some cases, a single CMU PLL is not sufficient to run the transmitter channels within a transceiver block at the desired data rates.

Using a second CMU PLL is useful if you want to combine channels that require different configurations, such as:

- Quartus II software-defined protocols (for example, Basic, gigabit ethernet [GIGE], SONET/synchronous digital hierarchy [SDH], serial digital interface [SDI], or PCI Express [PIPE] modes)
- CMU PLL bandwidth settings
- Different input reference clocks

Example 2

Consider an example design that requires four channels setup in the **Receiver and Transmitter** configuration, all in the same transceiver block, at the serial data rates shown in Table 3–2.

Table 3–2. Configuration for Example 2

User-Created Instance Name	ALTGX MegaWizard Plug-In Manager Settings		
	Number of Channels	**Configuration**	**Effective Data Rate**
inst0	1	Receiver and Transmitter	4.25 Gbps
inst1	1	Receiver and Transmitter	2.125 Gbps
Inst2	1	Receiver and Transmitter	1.0625 Gbps
Inst3	1	Receiver and Transmitter	5 Gbps

Assume that instance 0, 1, and 2 are driven from the same clock source and have the same CMU PLL bandwidth settings. In this case, you can use one CMU PLL for instance 0, 1, and 2. Refer to "Example 1" on page 3–4 for the ALTGX MegaWizard Plug-In Manger settings to enable the Quartus II software to share the same CMU PLL. A second CMU PLL is required for instance 3.

You can force the placement of transceiver channels to a specific transceiver block by assigning pins to the `tx_dataout` and `rx_datain` pins of the four ALTGX instances. Otherwise, the Quartus II software selects a transceiver bank.

Figure 3–3 and Figure 3–4 show the transceiver configuration before and after the Quartus II software combines the transceiver channels within the same transceiver block. Since the RX CDR is not shared between channels, only the CMU PLLs are shown.

☞ You must connect the `pll_powerdown` port of instance 0, 1, and 2 to the same logic output to share the same CMU PLL for these instances.

Figure 3–3. ALTGX Transceiver Channel Instances before Compilation for Example 2

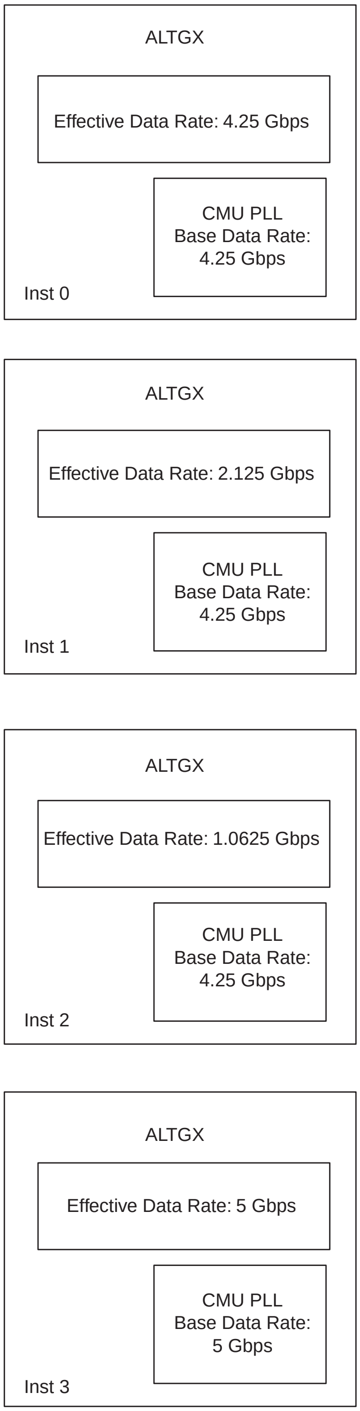
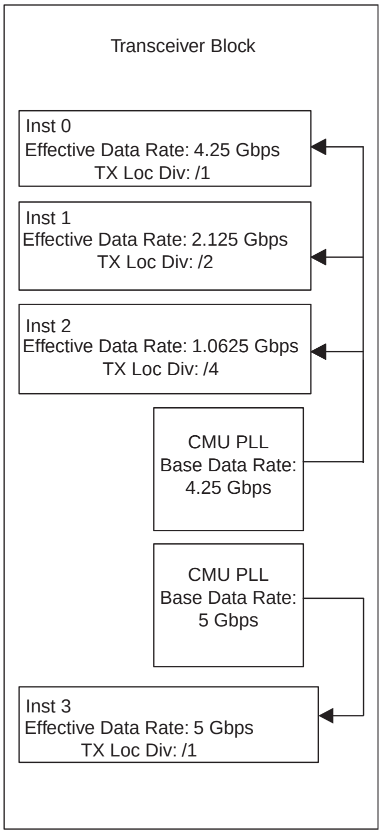


Figure 3–4. Combined Transceiver Channels Instances after Compilation for Example 2



In cases in which you have two instances with the same serial data rate but with different CMU PLL data rates, the Quartus II software creates a separate CMU PLL for the two instances. For example, consider the configuration shown in Table 3–3.

Table 3–3. Sample Configuration Using Two CMU PLLs

User-Created Instance Name	ALTGX MegaWizard Plug-In Manager Settings			
	Number of Channels	Configuration	Effective Data Rate	Base Data Rate
inst0	1	Receiver and Transmitter	2.5 Gbps	2.5 Gbps
inst1	1	Receiver and Transmitter	2.5 Gbps	5 Gbps
Inst2	1	Receiver and Transmitter	1 Gbps	1 Gbps

☞ Even though the effective data rate of inst1 is 2.5 Gbps (5 Gbps/2 = 2.5 Gbps), the same as inst0, when you compile the design, the Quartus II software requires two CMU PLLs to provide clocks for the transmitter side of the two instances because their base data rates are different. In this example, you have the third instance, inst2, that requires a third CMU PLL. Therefore, the Quartus II software cannot combine the above three instances within the same transceiver block.

Combining Receiver Only Channels

You can selectively use the receiver in the transceiver channel by selecting the **Receiver only** configuration in the **What is the Operating Mode?** option on the **General** screen of the ALTGX MegaWizard Plug-In Manager.

You can combine **Receiver only** channel instances of different configurations and data rates into the same transceiver block. Since each receiver channel contains its own dedicated CDR, each **Receiver only** instance (assuming one receiver channel per instance) can have different data rates. Note that for the Quartus II software to combine the **Receiver only** instances within the same transceiver block, you must connect `gxb_powerdown` (if used) of all the channel instances to the same logic or input pin. For more information, refer to "General Requirements to Combine Channels" on page 3–2.

It is possible to have up to four receiver channels that can run at different data rates by using separate input reference clocks, so long as there are enough clock routing resources available.

If you instantiate the **Receiver only** configuration, the ALTGX MegaWizard Plug-In Manager does not allow you to enable the rate-matching FIFO (clock rate compensation FIFO) in the receiver channel PCS because `tx_clkout` is not available in a **Receiver only** instance to clock the read-side of the rate-matching FIFO. If you need to perform clock rate compensation, implement the rate-matching FIFO in the FPGA fabric.

☞ If your design contains a receiver only instance, the Quartus II software disables all the settings for the unused transmitter channel present in the same physical transceiver channel. Therefore the unused transmitter channel is always powered down in hardware.

Combining Transmitter Channel and Receiver Channel Instances

You can create separate transmitter and receiver channel instances and assign the `tx_dataout` and `rx_datain` pins of the transmitter and receiver instances, respectively, to the same physical transceiver channel. This configuration is useful in cases where you intend to run the transmitter and receiver channel at different serial data rates. To create separate transmitter and receiver channel instances, select the **Transmitter only** and **Receiver only** options in the operating mode (**General** screen) of the ALTGX MegaWizard Plug-In Manager.

Multiple Transmitter Channel and Receiver Channel Instances

The Quartus II software allows you to combine multiple **Transmitter only** and **Receiver only** channel instances within the same transceiver block. Based on the pin assignments, the Quartus II software combines the corresponding **Transmitter only** and **Receiver only** channels in the same physical channel. To enable the Quartus II software to combine the transmitter channel and receiver channel instances in the same transceiver block, follow the rules and requirements outlined in:

- "General Requirements to Combine Channels" on page 3–2
- "Multiple Channels Sharing a CMU PLL" on page 3–4
- "Multiple Channels Sharing Two CMU PLLs" on page 3–8

- "Combining Receiver Only Channels" on page 3–12

Example 3

Consider that you want to create the four ALTGX instances shown in Table 3–4.

Table 3–4. Four ALTGX Instances for Example 3

Instance Name	Configuration	Serial Data Rate	Input Reference Clock Frequency
inst0	Transmitter only	3.125 bps	156.25 MHz
inst1	Receiver only	2.5 Gbps	156.25 MHz (same as inst0)
inst2	Transmitter only	1.25 Gbps	125 MHz
inst3	Receiver only	2 Gbps	125 MHz (same as inst2)

After you create the above instances, if you force the placement of the instances, as shown in Table 3–5, the Quartus II software combines inst0 and inst1 to physical channel 0, and inst2 and inst3 to physical channel 1.

Table 3–5. Forced Placement of the Instances for Example 3

Instance Name	Physical Channel Pin Assignments in the Same Transceiver Block
inst0	TX pin of channel 0
inst1	RX pin of channel 0
inst2	TX pin of channel 1
inst3	RX pin of channel 1

Figure 3–5 and Figure 3–6 show the transceiver channel instances before and after compilation.

Figure 3–5. ALTGX Transceiver Channel Instances before Compilation for Example 3

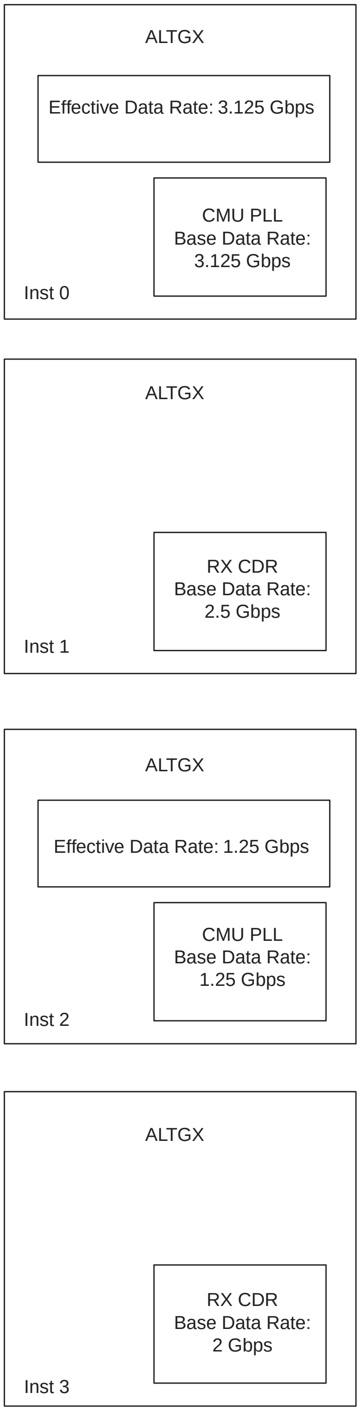
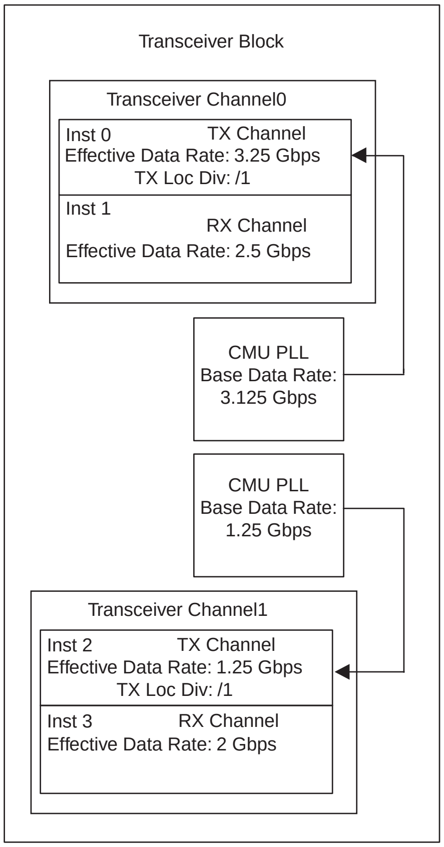


Figure 3–6. Combined Transceiver Instances after Compilation for Example 3



Combining Channels Configured in Protocol Functional Modes

Basic x4 Mode

The ALTGX MegaWizard Plug-In Manager provides a Basic mode with a ×4 option in the **Which sub-protocol will you be using?** option on the **General** screen. If you select this option, all the transmitter channels within the transceiver block receive the high-speed serial and low-speed parallel clock from the CMU0 clock divider block (present in the CMU0 channel). Each receiver channel within the transceiver block is clocked independently by the recovered clock from its receiver CDR.

When you use this mode, the ALTGX MegaWizard Plug-In Manager allows you to select one or more channels from the **What is the number of channels?** option on the **General** screen.

☞ If you select the number of channels to be fewer than four, the remaining transmitter channels within the transceiver block cannot be used. Therefore, if you have more than one instance configured in **Transmit only** mode or **Receiver and Transmitter** mode with the ×4 option enabled, the Quartus II software cannot combine the instances within the same transceiver block. You can use the available receiver channels in any configuration.

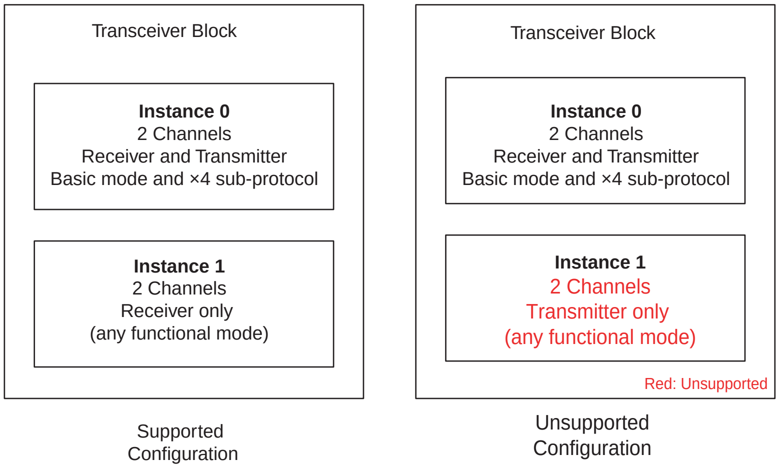
Note that only the transmitter channels share a common clock. The receiver channels are clocked independently. Therefore, you can configure the unused receiver channels within a transceiver block in any allowed configuration. For example, assume that you configure the ALTGX MegaWizard Plug-In Manager with the options shown in Table 3–6.

Table 3–6. Example of General Screen Options in Basic ×4 Mode

Option	Value
What protocol will you be using?	basic mode
Which subprotocol will you be using?	×4
What is the operating mode?	Receiver and Transmitter
What is the number of channels?	2

If you create the instance with the selection shown in Table 3–6, you cannot use the remaining two transmitter channels in the transceiver block. However, you can use the remaining two receiver channels in a different configuration. Figure 3–7 shows examples of supported and unsupported configurations.

Figure 3–7. Examples of Supported and Unsupported Configurations to Combine Instances in Basic ×4 Mode



Basic ×8 Mode

The ALTGX MegaWizard Plug-In Manager provides a Basic mode with a ×8 option in the **Which sub-protocol will you be using?** option on the **General** screen. This mode enables you to bond a maximum of eight transceiver channels from two adjacent transceiver blocks. If you select this option, all the transmitter channels in this configuration receive the high-speed serial and low-speed parallel clock from the CMU0 clock divider block (present in the CMU0 channel) of the master transceiver block.

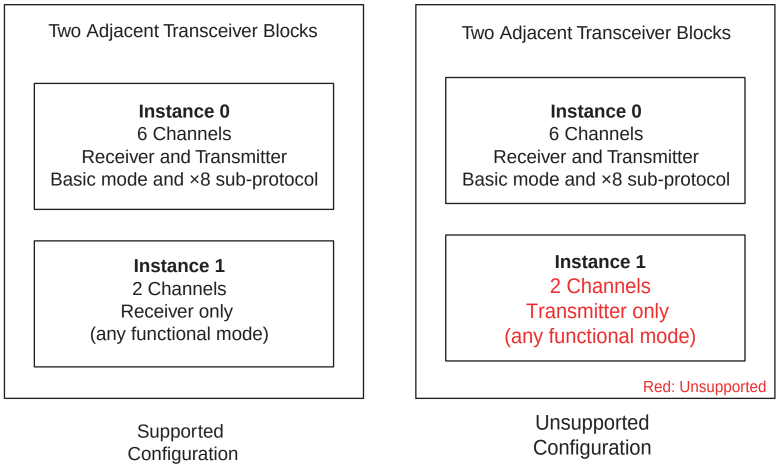
The master is the adjacent lower transceiver block. For further information about location requirements in Basic ×8 mode, refer to the *Bonded Channel Configurations* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Handbook*.

Each receiver channel configured in this mode is clocked independently by the recovered clock from its receiver CDR.

When you use the ×8 mode, the ALTGX MegaWizard Plug-In Manager allows you to select fewer than 8 channels from the **What is the number of channels?** option on the **General** screen.

☞ If you select the number of channels to be fewer than eight (for example, six), the remaining transmitter channels within the transceiver block cannot be used. Therefore, if you have more than one instance configured in **Transmit only** mode or **Receiver and Transmitter** mode with the **×8** option enabled, the Quartus II software cannot combine the instances within the same transceiver block. You can use the available receiver channels in any configuration. Figure 3–8 shows examples of supported and unsupported configuration in Basic ×8 mode.

Figure 3–8. Examples of Supported and Unsupported Configurations to Combine Instances in Basic ×8 Mode



(OIF) CEI Phy Interface Mode

The ALTGX MegaWizard Plug-In Manager provides a **Use central clock divider to improve transmitter jitter** option on the CEI screen (enabled by default) for the CEI functional mode. If you create a transmitter only or receiver and transmitter configuration in CEI mode with the **Use central clock divider to improve transmitter jitter** option enabled, the Quartus II software uses the CMU0 clock divider to provide high-speed clocks to all the transmitter channels within the transceiver block to improve transmitter jitter. The Quartus II software restricts you from using any unused transmitter channels within the transceiver block (the restrictions are similar to those explained in "Basic x4 Mode" on page 3–15). You can configure the unused receiver channels in any configuration.

Combining Channels Using the PCI Express Hard IP Block with Other Channels

The Stratix IV GX device contains an embedded PCI Express hard IP block that performs the phyMAC, datalink, and transaction layer functionality specified by PCI Express base specification 2.0. Each PCI Express hard IP block is shared by two transceiver blocks. The PCI Express Compiler Wizard provides you the options to configure the PCI Express hard IP block. When enabled, the transceiver channels associated with this block are enabled.

There are restrictions on combining transceiver channels with different functional and/or protocol modes (for example, Basic mode) within two contiguous transceiver blocks with the channels that use PCI Express hard IP block. The restrictions depend on the number of channels used (×1 or ×4) and the number of virtual channels (VCs) selected in the PCI Express compiler MegaWizard Plug-In Manager. Table 3–7 shows the restrictions.

When you use the PCI Express hard IP block there are placement restrictions on the locations of the transceiver channels. Refer to the *PCI Express Compiler User Guide* for these channel placement restrictions.

Table 3–7. PCI Express Hard IP Block Restrictions When Combining Transceiver Channels with Different Functional and/or Protocol Modes *(Note 1)*, *(2)*

PCI Express Configuration (PCI Express hard IP Options Enabled in the PCIE Compiler Wizard)			Transceiver Block 0 *(3)*				Transceiver Block 1 *(4)*			
Link Width	**Lane (Data Interface Width)**	**Virtual Channel (VC)**	**Ch0 *(5)***	**Ch1**	**Ch2**	**Ch3**	**Ch4**	**Ch5**	**Ch6**	**Ch7**
×1	64-bit	1	PCIe ×1	Avail.	Avail.	Avail.	Avail.	Avail.	Avail.	Avail.
		2	PCIe ×1	Avail.	N/A	N/A	Avail.	Avail.	Avail.	Avail.
×4	64-bit	1	PCIe ×4				Avail.	Avail.	Avail.	Avail.
		2	PCIe ×4				Avail.	Avail.	Avail.	Avail.
×4	128-bit	1	PCIe ×4				Avail.	Avail.	Avail.	Avail.
		2	PCIe ×4				N/A	N/A	Avail.	Avail.
×8	—	—	PCIe ×8							

Notes to Table 3–7:

(1) Avail.—the channels can be used in other configurations.

(2) N/A—the channels are NOT available for use.

(3) Transceiver block 0—the master transceiver block that provides high-speed serial and low-speed parallel clocks in a PCI Express (PIPE) ×4 or ×8 configuration.

(4) Transceiver block 1—the adjacent transceiver block that shares the same PCI Express hard IP block with transceiver block 0.

(5) The physical channel 0 in the transceiver block. For more information about physical to logical channel mapping in PCI Express (PIPE) functional mode, refer to the ×8 Channel Configuration section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*.

For more information about the *PCI Express Compiler MegaCore Functions* and hard IP implementation, refer to the *PCI Express Compiler User Guide.*

Combining Transceiver Instances Using PLL Cascade Clocks

The Stratix IV GX device provides multiple input reference clock sources to clock the CMU PLLs and RX CDRs in each transceiver block. The following are the input reference clock sources that can clock the CMU PLLs and RX CDRs:

- `refclks` from the same transceiver block
- Global clock lines
- `refclks` from transceiver blocks on the same side of the device using the inter-transceiver block (ITB) lines
- PLL cascade clock (this is the cascaded clock output from the PLLs in the FPGA fabric)

If you use the PLL cascade clock to provide input reference clocks to the CMU PLLs or RX CDRs, there are requirements for combining transceiver channels (as described in the following sections).

The Stratix IV GX transceiver has the ability to cascade the output of the general purpose PLLs (`PLL_L` and `PLL_R`) to the CMU PLLs and receiver CDRs. The left side PLLs can only be cascaded with the transceivers on the left side of the device. Similarly, the right side PLLs can only be cascaded with the transceivers on the right side of the device. Each side of the Stratix IV GX device contains a PLL cascade clock network; a single line network that connects the PLL cascade clock to the transceiver block. This clock line is segmented to allow different PLL cascade clocks to drive the transceiver CMU PLLs and RX CDRs.

The segmentation locations differ based on the device family. Therefore, there are restrictions when you want to combine transceiver channels that use different PLL cascade clocks as input reference clocks.

For more information about using the PLL cascade clock and segmentation, refer to the PLL Cascading section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*.

Combining Transceiver Instances in Multiple Transceiver Blocks

In “Creating Transceiver Channel Instances” on page 3–1, the method to instantiate multiple transceiver channels using a single ALTGX instance was discussed. The following section discusses the method to instantiate multiple transceiver channels using multiple transceiver blocks.

When you create a transceiver instance that has more than four transceiver channels, the Quartus II software attempts to combine the transceiver channels in multiple transceiver blocks. This is shown in the following examples.

Example 1

Consider that you want to create two ALTGX instances with the configuration shown in Table 3–8.

Table 3–8. Two ALTGX Instances for Example 1

Instance Name	Number of Transceiver Channels	Configuration	Serial Data Rate	Input Reference Clock
inst0	7	Receiver and Transmitter	4.25 Gbps	125 MHz from `refclk0`
inst1	1	Receiver and Transmitter	4.25 Gbps	125 MHz from `refclk0` (same as inst0)

In this case, assuming that all the required parameters specified in "Multiple Channels Sharing a CMU PLL" on page 3–4 are identical for inst0 and inst1, the Quartus II software fits inst0 and inst1 in two transceiver blocks.

Figure 3–9 and Figure 3–10 show the transceiver instances before and after compilation.

Figure 3–9. Transceiver Channel Instances before Compilation for Example 1

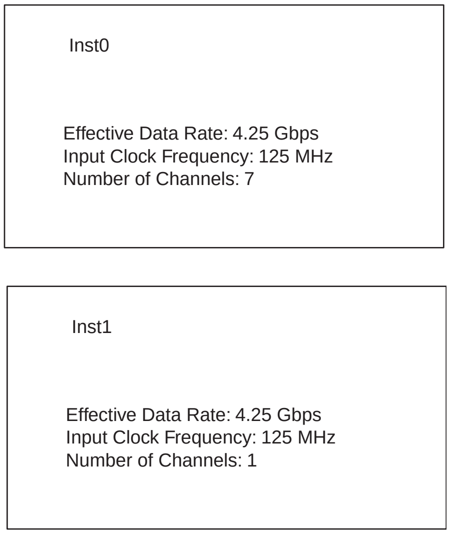
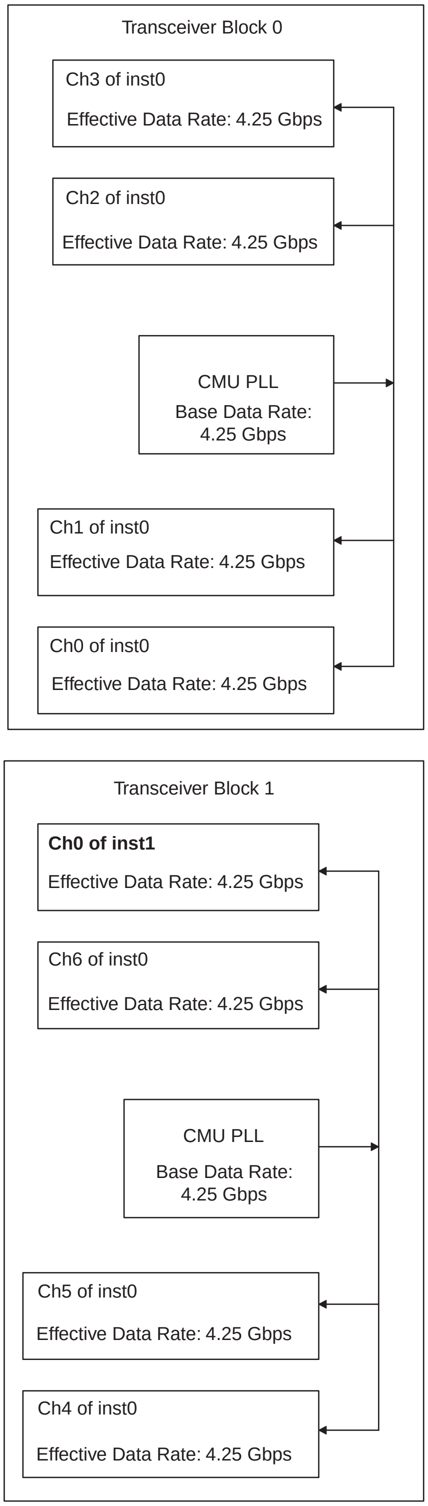

Inst0
Effective Data Rate: 4.25 Gbps
Input Clock Frequency: 125 MHz
Number of Channels: 7
Inst1
Effective Data Rate: 4.25 Gbps
Input Clock Frequency: 125 MHz
Number of Channels: 1

Figure 3–10. Combined Transceiver Instances after Compilation for Example 1


Transceiver Block 0
Ch3 of inst0
Effective Data Rate: 4.25 Gbps
Ch2 of inst0
Effective Data Rate: 4.25 Gbps
CMU PLL
Base Data Rate:
4.25 Gbps
Ch1 of inst0
Effective Data Rate: 4.25 Gbps
Ch0 of inst0
Effective Data Rate: 4.25 Gbps
Transceiver Block 1
Ch0 of inst1
Effective Data Rate: 4.25 Gbps
Ch6 of inst0
Effective Data Rate: 4.25 Gbps
CMU PLL
Base Data Rate:
4.25 Gbps
Ch5 of inst0
Effective Data Rate: 4.25 Gbps
Ch4 of inst0
Effective Data Rate: 4.25 Gbps

You can force the placement of the transceiver channels in specific transceiver banks by assigning pins to the `tx_dataout` and `rx_datain` ports of inst0 and inst1.

Note that even though inst0 instantiates seven transceiver channels, the ALTGX MegaWizard Plug-In Manager provides only one bit-wide `pll_inclk` port for inst0. In your design, provide only one clock input for the `pll_inclk` port. The Quartus II software uses two transceiver blocks to fit the seven channels and internally connects the input reference clock (connected to the `pll_inclk` port in your design) to the CMU PLLs of two transceiver blocks.

☞ For inst1, the ALTGX MegaWizard Plug-In Manager provides a `pll_inclk` port. In this example, it is assumed that a single reference clock is provided for inst0 and inst1. Therefore, connect the `pll_inclk` port of inst0 and inst1 to the same input reference clock pin. This enables the Quartus II software to share a single CMU PLL in transceiver block1 that has three channels of inst0 and one channel of inst1 (shown as ch5, ch6, and ch7 in transceiver block 1 in Figure 3–10).

For the RX CDRs in inst0, the ALTGX MegaWizard Plug-In Manager provides seven bits for the `rx_cruclk` port (if you do not select the **Train Receiver CDR from pll_inclk** option in the **PLL/Ports** screen). This allows separate input reference clocks to the RX CDRs of each channel.

Summary

The following is a summary for configuring multiple protocols and data rates in a transceiver block:

- You can run each transceiver channel at independent data rates or in independent protocol functional modes.
- Each transceiver block consists of two CMU PLLs that provide clocks to run the transmitter channels wibthin the transceiver block.
- To enable the Quartus II software to combine multiple instances of transceiver channels within a transceiver block, follow the rules specified in "General Requirements to Combine Channels" on page 3–2 and "Sharing CMU PLLs" on page 3–4.
- You can reset each CMU PLL within a transceiver block using a `pll_powerdown` signal. For each transceiver instance, the ALTGX MegaWizard Plug-In Manager provides an option to select the `pll_powerdown` port. If you want to share the same CMU PLL between multiple transceiver channels, connect the `pll_powerdown` ports of the instances and drive the signal from the same logic.
- If you enable the PCI Express hard IP block using the PCI Express compiler, the Quartus II software has certain requirements about using the remaining transceiver channels within the transceiver block in other configurations. Refer to "Combining Channels Using the PCI Express Hard IP Block with Other Channels" on page 3–18 for more information.

Referenced Documents

This chapter references the following documents:

- *PCI Express Compiler User Guide*
- *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*
- *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*
- *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*

Document Revision History

Table 3–9 shows the revision history for this document.

Table 3–9. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	■ Updated "Transmitter Buffer Voltage (V_{CCH})" on page 3–2 ■ Added "reconfig_fromgxb and reconfig_togxb Ports" on page 3–3 ■ Updated Figure 3–7 ■ Added "Basic ×8 Mode" on page 3–16 ■ Added Figure 3–8 ■ Updated Table 3–7	—
May 2008 v1.0	Initial release.	—

4. Reset Control and Power Down

SIV52004-2.0

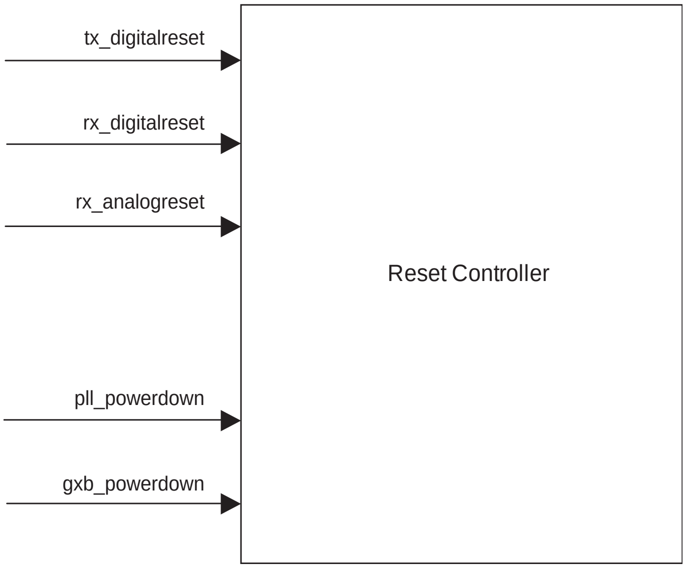
Introduction

Stratix® IV GX devices offer multiple reset signals to control transceiver channels and clock multiplier unit (CMU) phase-locked loops (PLLs) independently. The ALTGX Transceiver MegaWizard® Plug-In Manager provides individual reset signals for each channel instantiated in the design. It also provides one power-down signal for each transceiver block. Figure 4–1 shows the reset control and power-down block for a Stratix IV GX device.

This chapter includes the following sections:

- "Transceiver Reset Sequences" on page 4–4
- "PMA Direct Drive Mode Reset Sequences" on page 4–21
- "Dynamic Reconfiguration Reset Sequences" on page 4–27
- "Power Down" on page 4–30
- "Simulation Requirements" on page 4–31

Figure 4–1. Reset Control and Power-Down Block



User Reset and Power-Down Signals

Each transceiver channel in the Stratix IV GX device has individual reset signals to reset its physical coding sublayer (PCS) and physical medium attachment (PMA) blocks. Each CMU PLL in the transceiver block has a dedicated reset signal. The transceiver block also has a power-down signal that affects all the channels and CMU PLLs in the transceiver block.

☞ All reset and power-down signals are asynchronous.

The following reset signals are available for each transceiver channel:

- tx_digitalreset—provides asynchronous reset to all digital logic in the transmitter PCS, including the XAUI transmit state machine. This signal is available in the ALTGX MegaWizard Plug-In Manager in **Transmitter Only** and **Receiver and Transmitter** configurations. The minimum pulse width for this signal is two parallel clock cycles.
- rx_digitalreset—resets all digital logic in the receiver PCS, including the XAUI and GIGE receiver state machines, the XAUI channel alignment state machine, the BIST-PRBS verifier, and the BIST-incremental verifier. This signal is available in the ALTGX MegaWizard Plug-In Manager in **Receiver Only** and **Receiver and Transmitter** configurations. The minimum pulse width for this signal is two parallel clock cycles.

☞ The tx_digitalreset and rx_digitalreset signals need to be asserted until the clocks coming out of the transmitter PLL and receiver CDR are stabilized. Stable parallel clocks are essential for proper operation of transmitter and receiver phase-compensation FIFOs in the PCS.

- rx_analogreset—resets the receiver CDR present in the receiver channel. This signal is available in the ALTGX MegaWizard Plug-In Manager in **Receiver Only** and **Receiver and Transmitter** configurations. The minimum pulse width is two parallel clock cycles.

The following power-down signal is available for each CMU PLL in the transceiver block:

- PLL_powerdown—each transceiver block has two CMU PLLs. Each CMU PLL has a dedicated power-down signal called PLL_powerdown. The PLL_powerdown signal powers down the CMU PLLs that provide high-speed serial and low-speed parallel clocks to the transceiver channels.

The following power-down signal is common to the transceiver block:

- gxb_powerdown—powers down the entire transceiver block. When this signal is asserted, the PCS and PMA in all the transceiver channels and the CMU PLLs are powered down. This signal operates independently from the other reset signals.

☞ The refclk (refclk0 or refclk1) buffer is not powered down by any of the above signals.

The following status signals are available:

- pll_locked—indicates the status of the transmitter PLL. A high on this signal shows that the transmitter PLL is locked to the incoming reference clock frequency.
- rx_pll_locked—a high on this signal shows that the receiver CDR is locked to the incoming reference clock frequency.

- rx_freqlocked—indicates the status of the receiver CDR lock mode. A high level indicates that the receiver is in lock-to-data mode. A low level indicates that the receiver CDR is in lock-to-reference mode.
- busy—indicates the status of the dynamic reconfiguration controller. The busy signal remains low for the very first reconfig_clk clock cycle after power up. It then gets asserted from the second reconfig_clk clock cycle. Assertion on this signal indicates that the offset cancellation process is being executed on the receiver buffer as well as the receiver CDR. When this signal is de-asserted, it indicates that the offset cancellation is complete.

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

If none of the channels are instantiated in a transceiver block, the Quartus® II software automatically powers down the entire transceiver block.

Blocks Affected by Reset and Power-Down Signals

Table 4–1 shows the blocks that are affected by specific reset and power-down signals.

Table 4–1. Blocks Affected by Reset and Power-Down Signals

Transceiver Block	rx_digitalreset	rx_analogreset	tx_digitalreset	PLL_powerdown	gxb_powerdown
CMU PLLs	—	—	—	✓	✓
Transmitter Phase Compensation FIFO	—	—	✓	—	✓
Byte Serializer	—	—	✓	—	✓
8B/10B Encoder	—	—	✓	—	✓
Serializer	—	—	✓	—	✓
Transmitter Buffer	—	—	—	—	✓
Transmitter XAUI State Machine	—	—	✓	—	✓
Receiver Buffer	—	—	—	—	✓
Receiver CDR	—	✓	—	—	✓
Receiver Deserializer	—	—	—	—	✓
Receiver Word Aligner	✓	—	—	—	✓
Receiver Deskew FIFO	✓	—	—	—	✓
Receiver Clock Rate Compensation FIFO	✓	—	—	—	✓
Receiver 8B/10B Decoder	✓	—	—	—	✓
Receiver Byte Deserializer	✓	—	—	—	✓
Receiver Byte Ordering	✓	—	—	—	✓
Receiver Phase Compensation FIFO	✓	—	—	—	✓
Receiver XAUI State Machine	✓	—	—	—	✓

Transceiver Reset Sequences

You can configure transceiver channels in Stratix IV GX devices in various configurations. In all functional modes except XAUI functional mode, transceiver channels can be either bonded or non-bonded. In XAUI functional mode, transceiver channels must be bonded. In PCI Express (PIPE) functional mode, transceiver channels can be either bonded or non-bonded and need to follow a specific reset sequence.

The two categories of reset sequences for Stratix IV GX devices described in this chapter are:

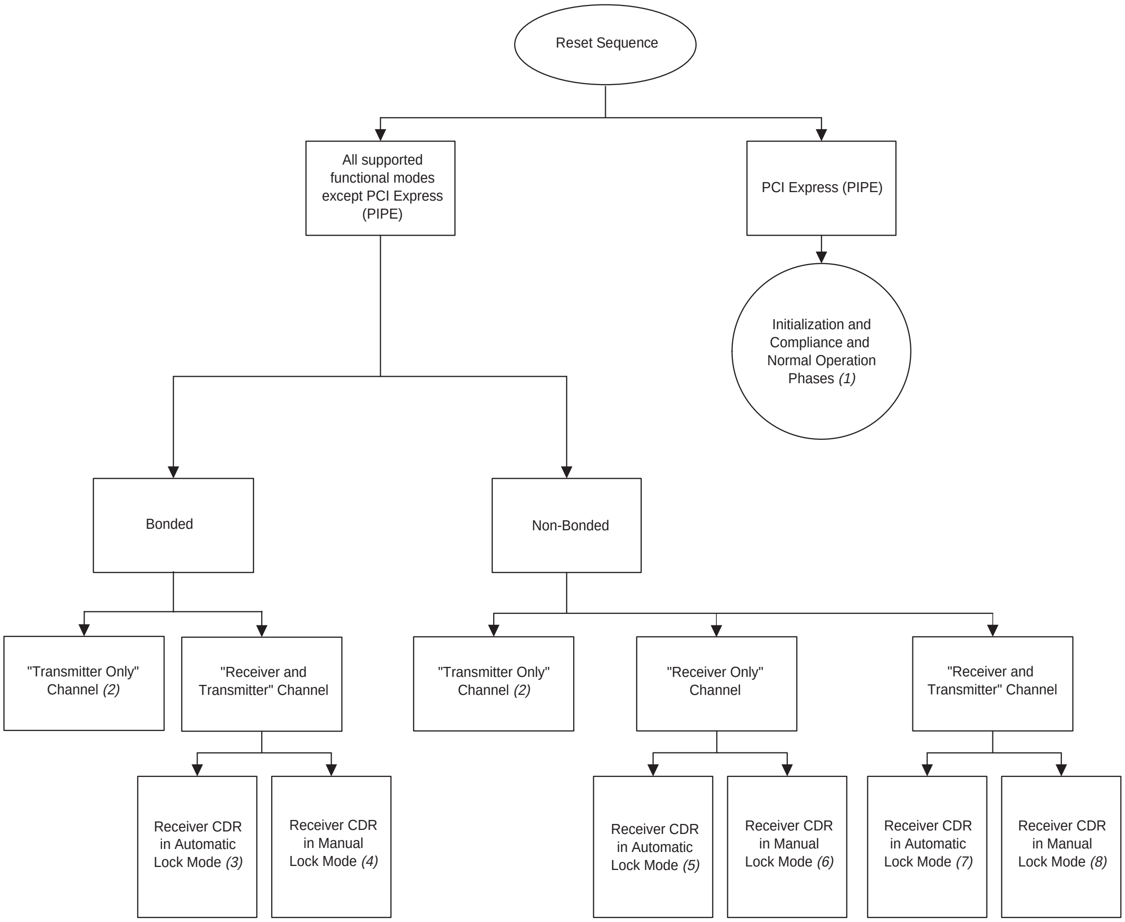
- "All Supported Functional Modes Except the PCI Express (PIPE) Functional Mode" on page 4–5—describes the reset sequences in bonded and non-bonded configurations.
- "PCI Express (PIPE) Functional Mode" on page 4–18—describes the reset sequence for the initialization/compliance phase and the normal operation phase in PCI Express (PIPE) functional modes.

☞ The busy signal remains low for the very first `reconfig_clk` clock cycle. It then gets asserted from the second `reconfig_clk` clock cycle. Subsequent deassertion of the busy signal indicates the completion of the offset cancellation process. This busy signal is required in transceiver reset sequences except for transmitter only channel configurations. Refer to the reset sequences shown in Figure 4–2 and the associated references listed in the notes for the figure.

☞ Altera strongly recommends adhering to these reset sequences for proper operation of the Stratix IV GX transceiver.

Figure 4–2 shows the transceiver reset sequences for Stratix IV GX devices.

Figure 4–2. Transceiver Reset Sequences Chart


Reset Sequence
All supported functional modes except PCI Express (PIPE)
PCI Express (PIPE)
Initialization and Compliance and Normal Operation Phases *(1)*
Bonded
Non-Bonded
"Transmitter Only" Channel *(2)*
"Receiver and Transmitter" Channel
"Transmitter Only" Channel *(2)*
"Receiver Only" Channel
"Receiver and Transmitter" Channel
Receiver CDR in Automatic Lock Mode *(3)*
Receiver CDR in Manual Lock Mode *(4)*
Receiver CDR in Automatic Lock Mode *(5)*
Receiver CDR in Manual Lock Mode *(6)*
Receiver CDR in Automatic Lock Mode *(7)*
Receiver CDR in Manual Lock Mode *(8)*

Notes to Figure 4–2:

(1) Refer to the Timing Diagram in Figure 4–10.
(2) Refer to the Timing Diagram in Figure 4–3.
(3) Refer to the Timing Diagram in Figure 4–4.
(4) Refer to the Timing Diagram in Figure 4–5.
(5) Refer to the Timing Diagram in Figure 4–6.
(6) Refer to the Timing Diagram in Figure 4–7.
(7) Refer to the Timing Diagram in Figure 4–8.
(8) Refer to the Timing Diagram in Figure 4–9.

All Supported Functional Modes Except the PCI Express (PIPE) Functional Mode

This section describes reset sequences for transceiver channels in bonded and non-bonded configurations. Timing diagrams of some typical configurations are shown to facilitate proper reset sequence implementation. In these functional modes, you can set the receiver CDR either in automatic lock or manual lock mode.

In manual lock mode, the receiver CDR locks to the reference clock (lock-to-reference) or the incoming serial data (lock-to-data), depending on the logic levels on the rx_locktorefclk and rx_locktodata signals. With the receiver CDR in manual lock mode, you can either configure the transceiver channels in the Stratix IV GX device in a non-bonded configuration or a bonded configuration. In a bonded configuration, for example in XAUI mode, four channels are bonded together.

Table 4–2 shows the lock-to-reference (LTR) and lock-to-data (LTD) controller lock modes for the rx_locktorefclk and rx_locktodata signals.

Table 4–2. Lock-To-Reference and Lock-To-Data Modes

rx_locktorefclk	rx_locktodata	LTR/LTD Controller Lock Mode
1	0	Manual, LTR Mode
—	1	Manual, LTD Mode
0	0	Automatic Lock Mode

Bonded Channel Configuration

In bonded channel configuration, you can reset all the bonded channels simultaneously. Examples of bonded channel configurations are the XAUI, PCI Express (PIPE), and BASIC ×4 functional modes. In BASIC ×4 functional mode, you can bond **Transmitter Only** channels together.

In XAUI mode, the receiver and the transmitter channels are bonded. Each of the receiver channels in this mode has its own output status signals, rx_pll_locked and rx_freqlocked. The timing of these signals is considered in the reset sequence.

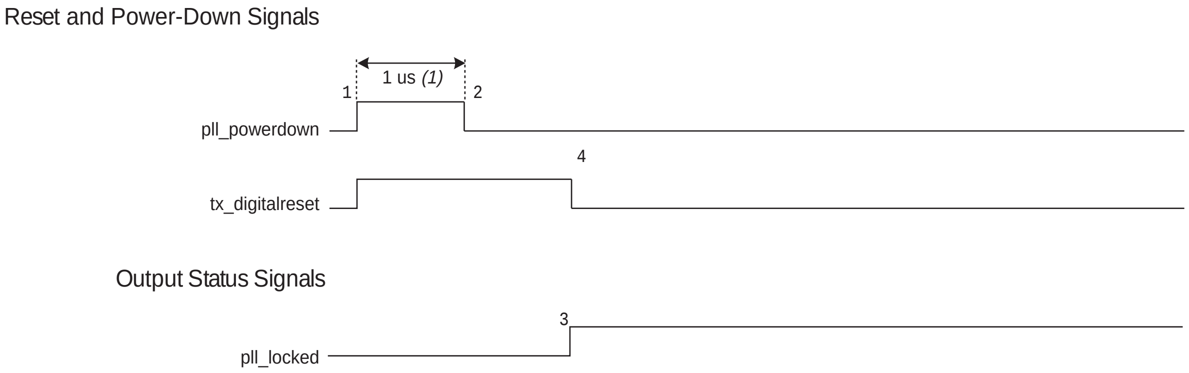
The following timing diagrams describe the reset and power-down sequences for bonded configurations under the following set-ups:

- **Transmitter Only** channel set-up—applicable to BASIC ×4 functional mode
- **Receiver and Transmitter** channel set-up—receiver CDR in automatic lock mode; applicable to XAUI functional mode
- **Receiver and Transmitter** channel set-up—receiver CDR in manual lock mode; applicable to XAUI functional mode

Transmitter Only Channel

This configuration contains only a transmitter channel. If you create a **Transmitter Only** instance in the ALTGX MegaWizard Plug-In Manager in BASIC ×4 functional mode, use the reset sequence shown in Figure 4–3.

Figure 4–3. Sample Reset Sequence for Four Transmitter Only Channels



Note to Figure 4–3:

(1) To be characterized.

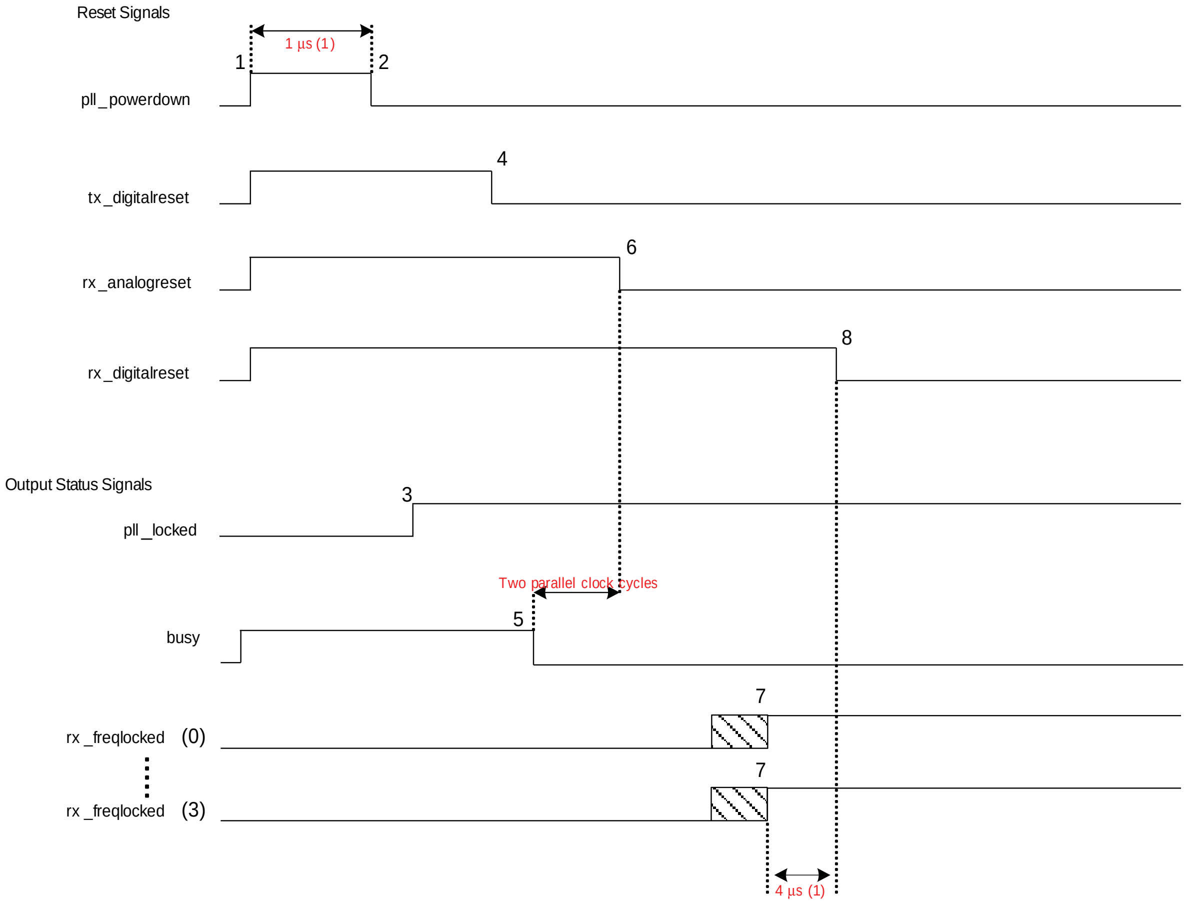
As shown in Figure 4–3, perform the following reset procedure for the **Transmitter Only** channel configuration:

1. After power up, assert `PLL_powerdown` for a minimum period of 1 µs (the time between markers 1 and 2).
2. Keep the `tx_digitalreset` signal asserted during this time period. After you de-assert the `PLL_powerdown` signal, the transmitter PLL starts locking to the transmitter input reference clock.
3. Once the transmitter PLL locks, as indicated by the `pll_locked` signal going high (marker 3), de-assert the `tx_digitalreset` signal (marker 4). At this point, the transmitter is ready for transmitting data.

Receiver and Transmitter Channel—Receiver CDR in Automatic Lock Mode

This configuration contains both a transmitter and receiver channel. For XAUI functional mode, with the receiver CDR in automatic lock mode, use the reset sequence shown in Figure 4–4.

Figure 4–4. Sample Reset Sequence for Four Receiver and Transmitter Channels—Receiver CDR in Automatic Lock Mode



Note to Figure 4–4:

(1) To be characterized.

As shown in Figure 4–4, perform the following reset procedure for the receiver CDR in automatic lock mode configuration:

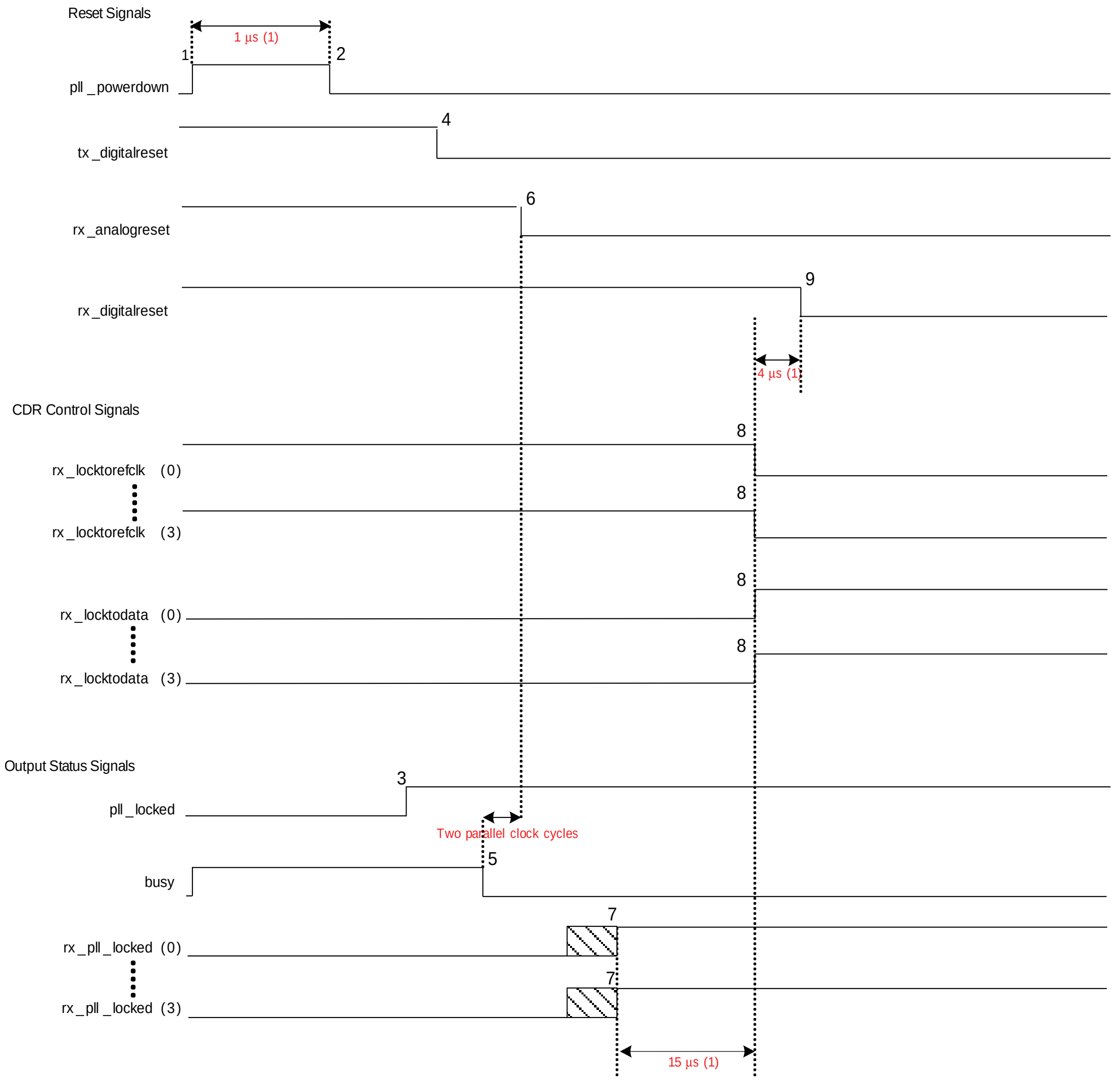
1. After power up, assert `PLL_powerdown` for a minimum period of 1 µs (the time between markers 1 and 2).
2. Keep the `tx_digitalreset`, `rx_analogreset`, and `rx_digitalreset` signals asserted during this time period. After you de-assert the `PLL_powerdown` signal, the transmitter PLL starts locking to the transmitter input reference clock.
3. After the transmitter PLL locks, as indicated by the `pll_locked` signal going high, de-assert the `tx_digitalreset` signal. At this point, the transmitter is ready for data traffic.
4. For the receiver operation, after de-assertion of `busy` signal, wait for two parallel clock cycles to de-assert the `rx_analogreset` signal. After `rx_analogreset` is de-asserted, the receiver CDR of each channel starts locking to the receiver input reference clock.

5. Wait for the `rx_freqlocked` signal from each channel to go high. The `rx_freqlocked` signal of each channel may go high at different times (indicated by the slashed pattern at marker 7).
6. In a bonded channel group, when the `rx_freqlocked` signals of all the channels has gone high, from that point onwards wait for at least 4 µs for the receiver parallel clock to be stable, then de-assert the `rx_digitalreset` signal (marker 8). At this point, all the receivers are ready for data traffic.

Receiver and Transmitter Channel—Receiver CDR in Manual Lock Mode

This configuration contains both a transmitter and receiver channel. For XAUI functional mode, with the receiver CDR in manual lock mode, use the reset sequence shown in Figure 4–5.

Figure 4–5. Sample Reset Sequence for Four Receiver and Transmitter Channels—Receiver CDR in Manual Lock Mode



Note to Figure 4–5:

(1) To be characterized.

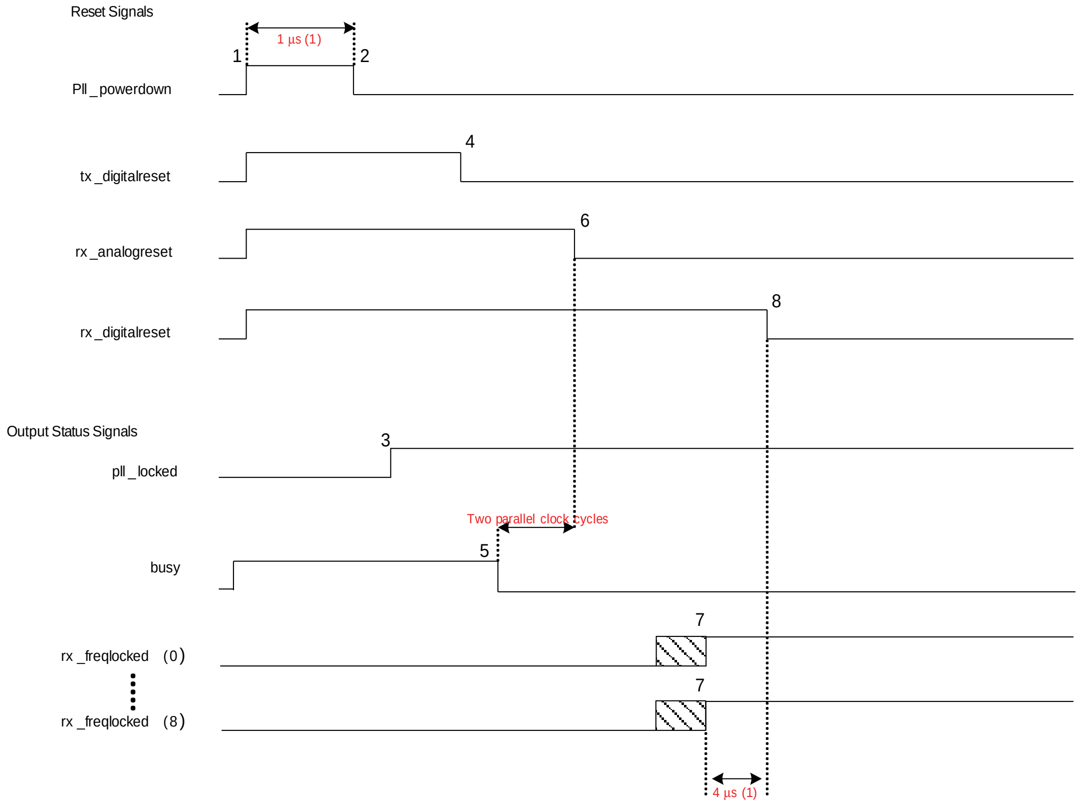
As shown in Figure 4–5, perform the following reset procedure for the receiver CDR in manual lock mode configuration:

1. After power up, assert `PLL_powerdown` for a minimum period of 1 µs (the time between markers 1 and 2).
2. Keep the `tx_digitalreset`, `rx_analogreset`, `rx_digitalreset`, and `rx_locktorefclk` signals asserted and the `rx_locktodata` signal de-asserted during this time period. After you de-assert the `PLL_powerdown` signal, the transmitter PLL starts locking to the transmitter input reference clock.
3. After the transmitter PLL locks, as indicated by the `pll_locked` signal going high (marker 3), de-assert the `tx_digitalreset` signal (marker 4). For the receiver operation, after de-assertion of the `busy` signal, wait for two parallel clock cycles to de-assert the `rx_analogreset` signal. After the `rx_analogreset` signal is de-asserted, the receiver CDR of each channel starts locking to the receiver input reference clock because `rx_locktorefclk` is asserted.
4. Wait for the `rx_pll_locked` signal from each channel to go high. The `rx_pll_locked` signal of each channel may go high at different times with respect to each other (indicated by the slashed pattern at the marker 7).
5. In a bonded channel group, when the `rx_pll_locked` signal of all the channels have gone high, from that point onwards, wait for at least 15 µs, then de-assert `rx_locktorefclk` and assert `rx_locktodata` (marker 8). At this point, the receiver CDR of all the channels enters into lock-to-data mode and starts locking to the received data.
6. De-assert `rx_digitalreset` at least 4 µs (the time between markers 8 and 9) after asserting the `rx_locktodata` signal.

Receiver and Transmitter Channel—Receiver CDR in Automatic Lock Mode

This configuration contains both a transmitter and a receiver channel. For BASIC ×8 functional mode, with the receiver CDR in automatic lock mode, use the reset sequence shown in Figure 4–6.

Figure 4–6. Sample Reset Sequence for Eight Receiver and Transmitter Channels—Receiver CDR in Automatic Lock Mode



Note to Figure 4–6:

(1) To be characterized.

As shown in Figure 4–6, perform the following reset procedure for the receiver CDR in automatic lock mode:

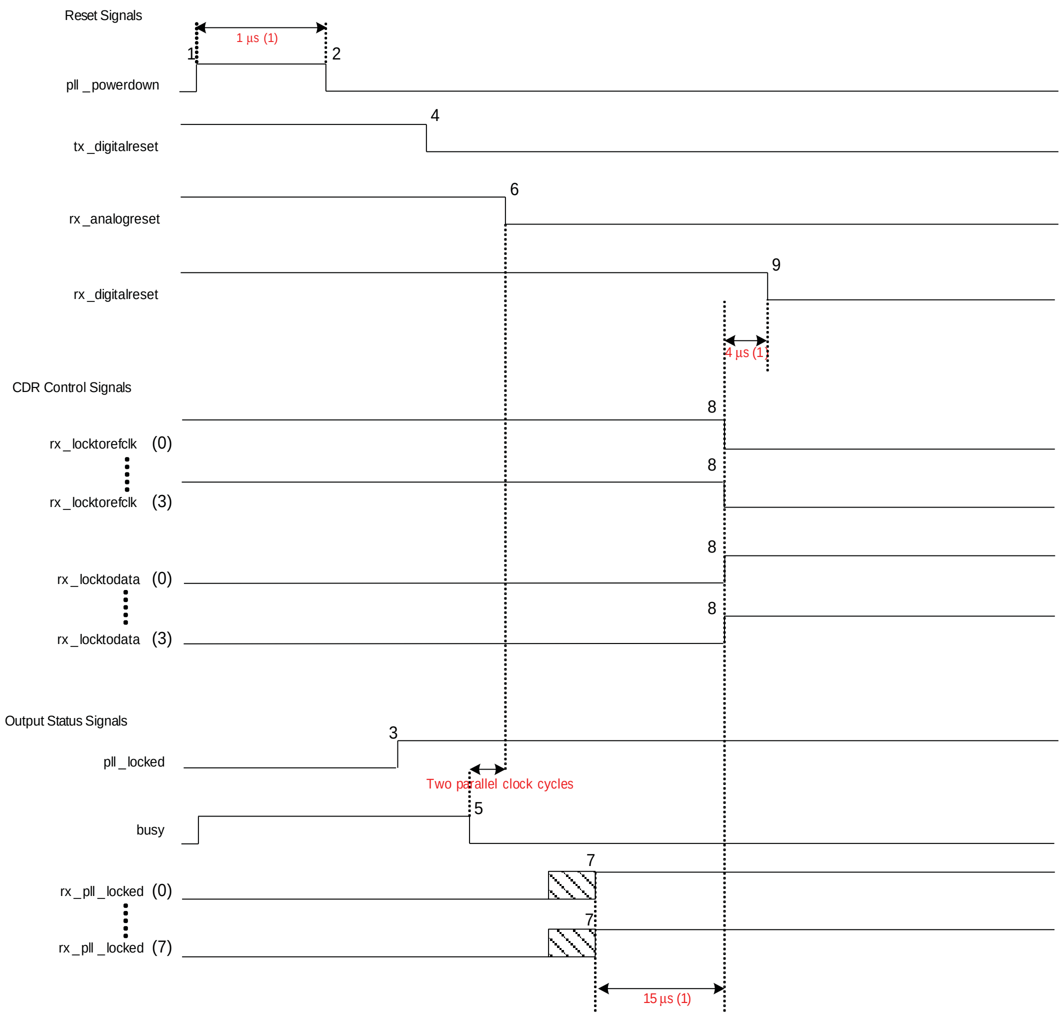
1. After power up, assert `PLL_powerdown` for a minimum period of 1 µs (the time between markers 1 and 2).
2. Keep the `tx_digitalreset`, `rx_analogreset`, and `rx_digitalreset` signals asserted during this time period. After you de-assert the `PLL_powerdown` signal, the transmitter PLL starts locking to the transmitter input reference clock.
3. After the transmitter PLL locks, as indicated by the `pll_locked` signal going high, de-assert the `tx_digitalreset` signal. At this point, the transmitter is ready for data traffic.
4. For the receiver operation, after de-assertion of the `busy` signal, wait for two parallel clock cycles to de-assert the `rx_analogreset` signal. After `rx_analogreset` is de-asserted, the receiver CDR of each channel starts locking to the receiver input reference clock.
5. Wait for the `rx_freqlocked` signal from each channel to go high. The `rx_freqlocked` signal of each channel may go high at different times (indicated by the slashed pattern at marker 7).

6. In a bonded channel group, when the `rx_freqlocked` signals of all the channels have gone high, from that point onwards, wait for at least 4 µs for the receiver parallel clock to stabilize, then de-assert the `rx_digitalreset` signal (marker 8). At this point, all the receivers are ready for data traffic.

Receiver and Transmitter Channel—Receiver CDR in Manual Lock Mode

This configuration contains both a transmitter and receiver channel. For BASIC ×8 functional mode, with the receiver CDR in manual lock mode, use the reset sequence shown in Figure 4–7.

Figure 4–7. Sample Reset Sequence for Eight Receiver and Transmitter Channels—Receiver CDR in Manual Lock Mode



Note to Figure 4–7:

(1) To be characterized.

As shown in Figure 4–7, perform the following reset procedure for the receiver CDR in manual lock mode:

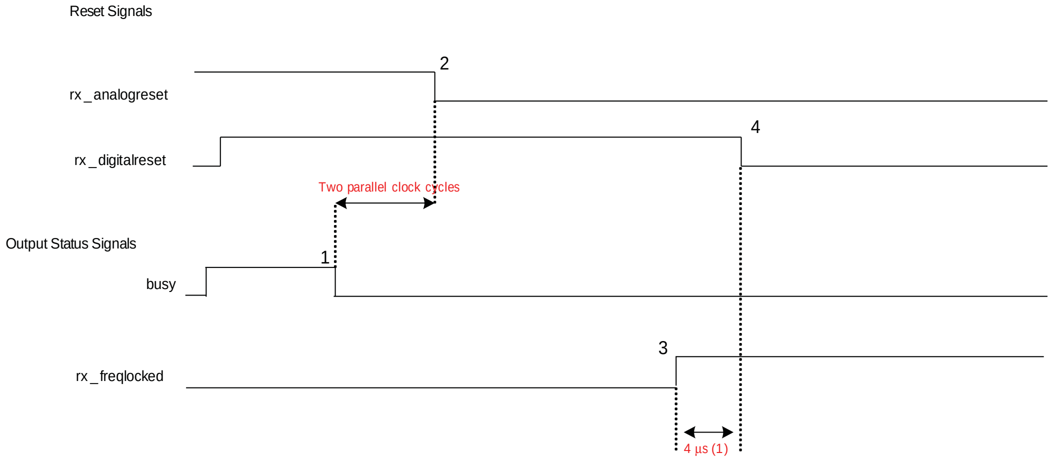
1. After power up, assert `PLL_powerdown` for a minimum period of 1 µs (the time between markers 1 and 2).

2. Keep the tx_digitalreset, rx_analogreset, rx_digitalreset, and rx_locktorefclk signals asserted and the rx_locktodata signal de-asserted during this time period. After you de-assert the PLL_powerdown signal, the transmitter PLL starts locking to the transmitter input reference clock.
3. After the transmitter PLL locks, as indicated by the pll_locked signal going high (marker 3), de-assert the tx_digitalreset signal (marker 4). For the receiver operation, after de-assertion of the busy signal, wait for two parallel clock cycles to de-assert the rx_analogreset signal. After the rx_analogreset signal is de-asserted, the receiver CDR of each channel starts locking to the receiver input reference clock because rx_locktorefclk is asserted.
4. Wait for the rx_pll_locked signal from each channel to go high. The rx_pll_locked signal of each channel may go high at different times with respect to each other (indicated by the slashed pattern at the marker 7).
5. In a bonded channel group, when the rx_pll_locked signal of all the channels has gone high, from that point onwards, wait for at least 15 µs, then de-assert rx_locktorefclk and assert rx_locktodata (marker 8). At this point, the receiver CDR of all the channels enters into lock-to-data mode and starts locking to the received data.
6. De-assert rx_digitalreset at least 4 µs (the time between markers 8 and 9) after asserting the rx_locktodata signal.

Receiver Only Channel—Receiver CDR in Automatic Lock Mode

This configuration contains only a receiver channel. If you create a **Receiver Only** instance in the ALTGX MegaWizard Plug-In Manager with the receiver CDR in automatic lock mode, use the reset sequence shown in Figure 4–8.

Figure 4–8. Sample Reset Sequence of Receiver Only Channel—Receiver CDR in Automatic Lock Mode



Note to Figure 4–8:

(1) To be characterized.

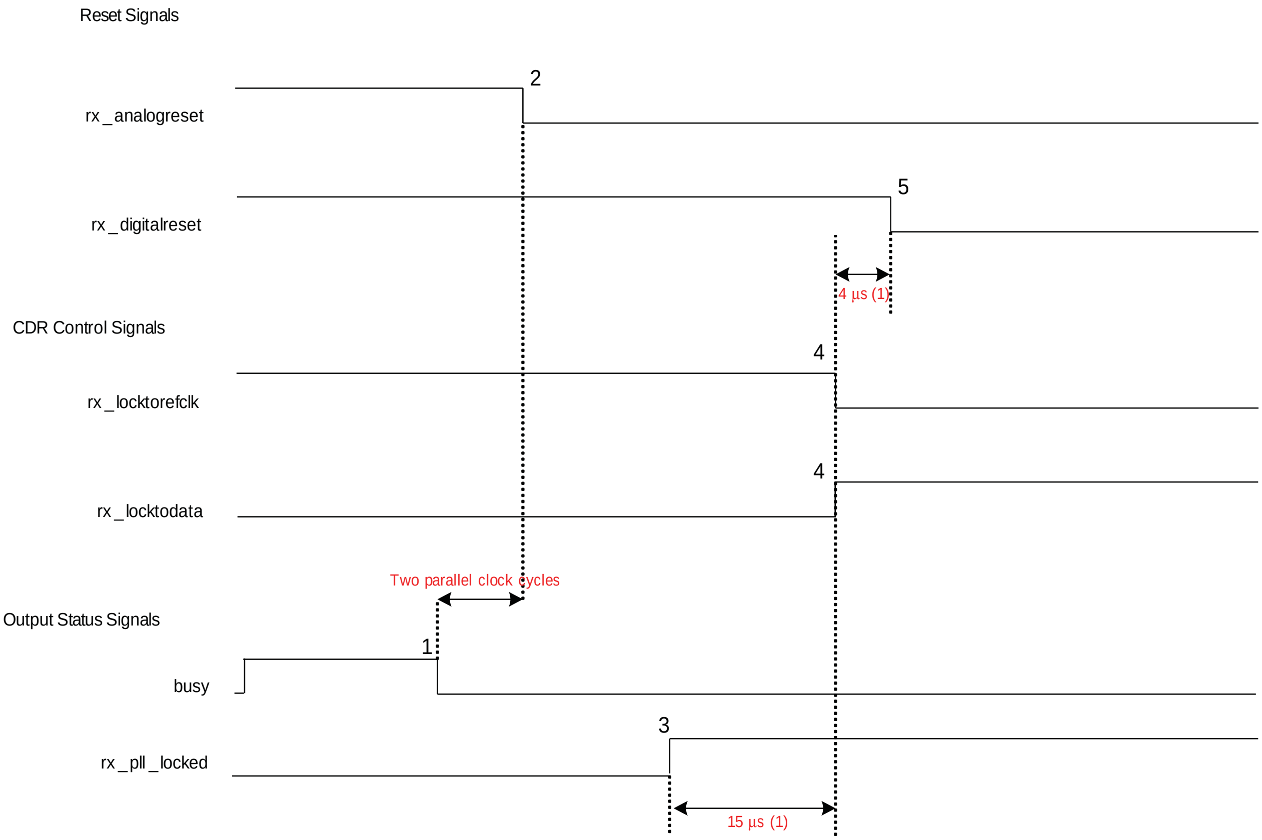
As shown in Figure 4–8, perform the following reset procedure for the receiver in CDR automatic lock mode:

1. After power up, wait for the `busy` signal to be de-asserted. After which, de-assert the `rx_analogreset` signal.
2. Keep the `rx_digitalreset` signal asserted during this time period. After you de-assert the `rx_analogreset` signal, the receiver CDR starts locking to the receiver input reference clock.
3. Wait for the `rx_freqlocked` signal to go high.
4. When `rx_freqlocked` goes high (marker 3), from that point onwards, wait for at least 4 µs, then de-assert the `rx_digitalreset` signal (marker 4). At this point, the receiver is ready to receive data.

Receiver Only Channel—Receiver CDR in Manual Lock Mode

This configuration contains only a receiver channel. If you create a **Receiver Only** instance in the ALTGX MegaWizard Plug-In Manager with receiver CDR in manual lock mode, use the reset sequence shown in Figure 4–9.

Figure 4–9. Sample Reset Sequence of Receiver Only Channel—Receiver CDR in Manual Lock Mode



Note to Figure 4–9:

(1) To be characterized.

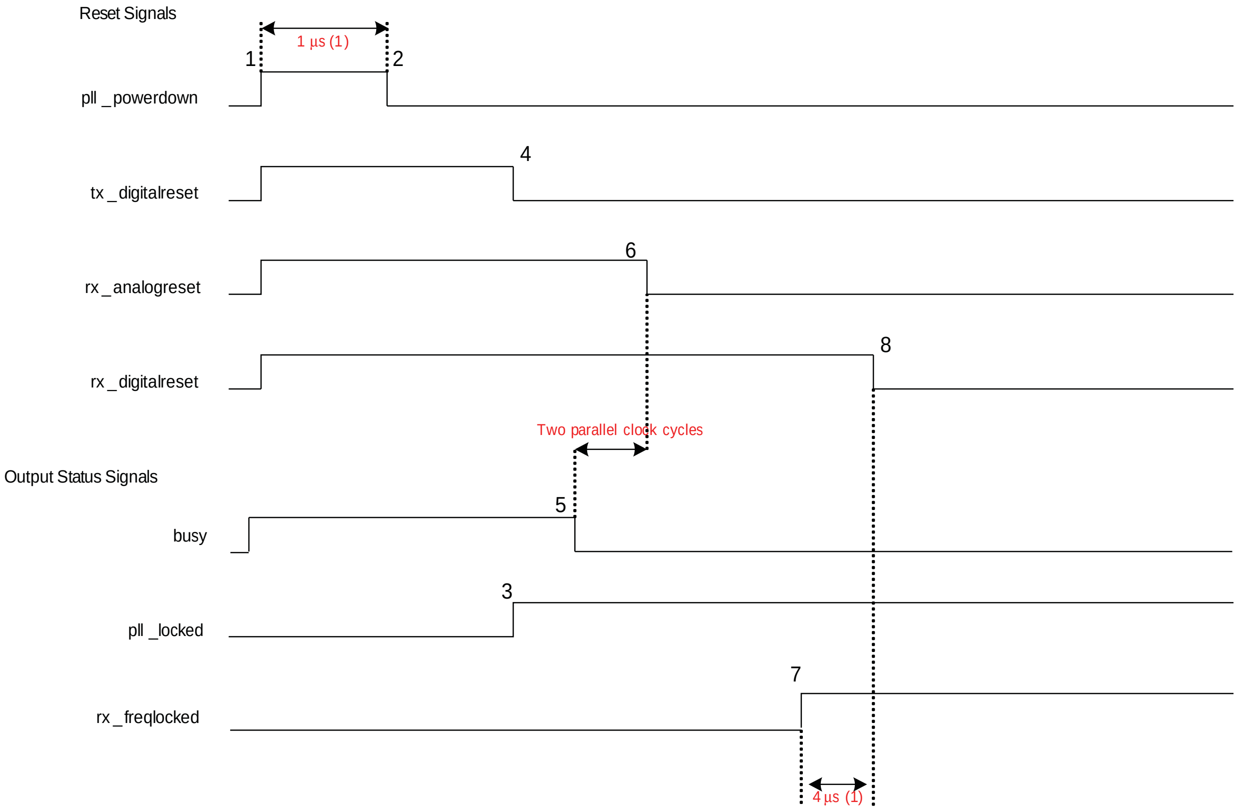
As shown in Figure 4–9, perform the following reset procedure for the receiver CDR in manual lock mode:

1. After power up, wait for the busy signal to be asserted.
2. Keep the rx_digitalreset and rx_locktorefclk signals asserted and the rx_locktodata signal de-asserted during this time period.
3. After de-assertion of the busy signal, de-assert the rx_analogreset signal, after which the receiver CDR starts locking to the receiver input reference clock because the rx_locktorefclk signal is asserted.
4. Wait for at least 15 µs (the time between markers 3 and 4) after the rx_pll_locked signal goes high and then de-assert the rx_locktorefclk signal. At the same time, assert the rx_locktodata signal (marker 4). At this point, the receiver CDR enters lock-to-data mode and the receiver PLL starts locking to the received data.
5. De-assert rx_digitalreset at least 4 µs (the time between markers 4 and 5) after asserting the rx_locktodata signal.

Receiver and Transmitter Channel—Receiver CDR in Automatic Lock Mode

This configuration contains both a transmitter and a receiver channel. If you create a **Receiver and Transmitter** instance in the ALTGX MegaWizard Plug-In Manager with receiver CDR in automatic lock mode, use the reset sequence shown in Figure 4–10.

Figure 4–10. Sample Reset Sequence of Receiver and Transmitter Channel—Receiver CDR in Automatic Lock Mode



Note to Figure 4–10:

(1) To be characterized.

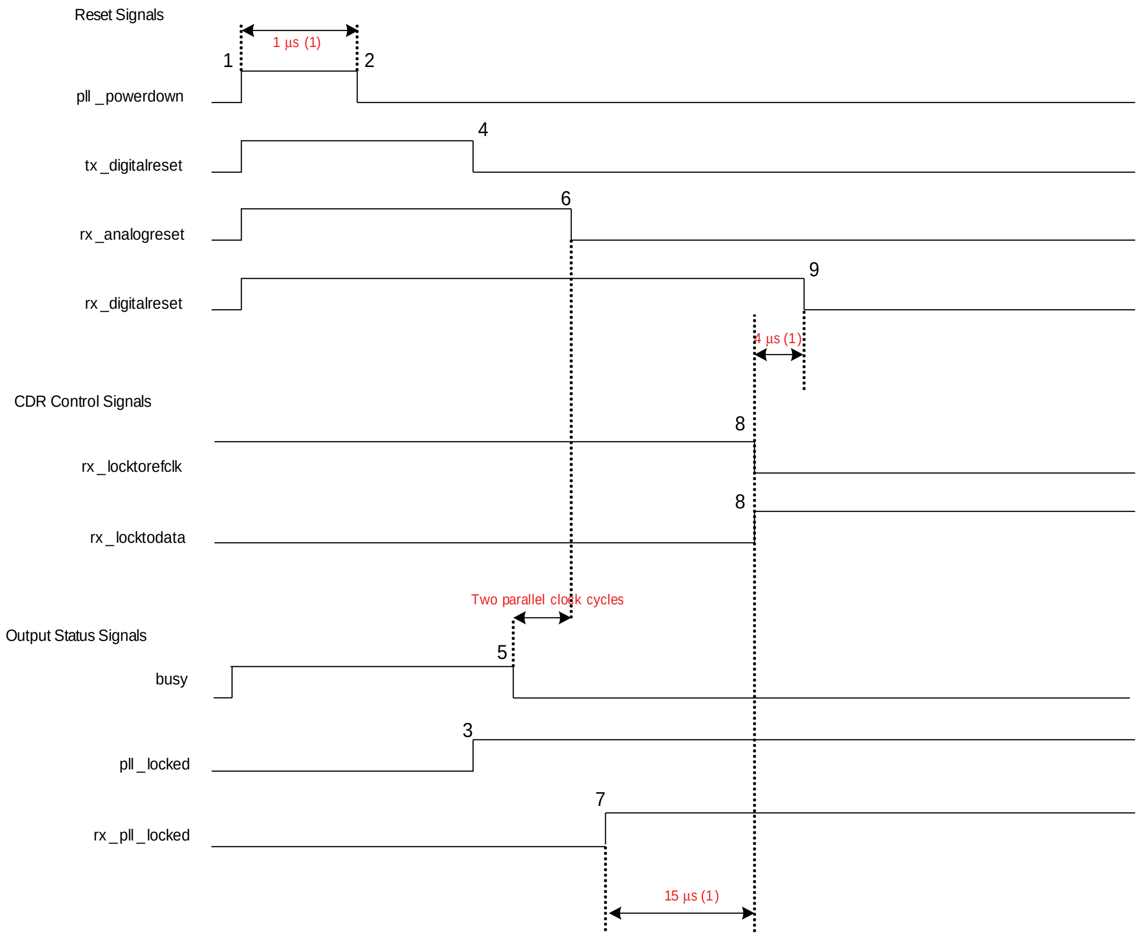
As shown in Figure 4–10, perform the following reset procedure for the receiver in CDR automatic lock mode:

1. After power up, assert `PLL_powerdown` for a minimum period of 1 µs (the time between markers 1 and 2).
2. Keep the `tx_digitalreset`, `rx_analogreset`, and `rx_digitalreset` signals asserted during this time period. After you de-assert the `PLL_powerdown` signal, the transmitter PLL starts locking to the transmitter input reference clock.
3. After the transmitter PLL locks, as indicated by the `pll_locked` signal going high (marker 3), de-assert `tx_digitalreset`. For receiver operation, wait for the `busy` signal to be de-asserted, after which `rx_analogreset` is de-asserted. After you de-assert `rx_analogreset`, the receiver CDR starts locking to the receiver input reference clock.
4. Wait for the `rx_freqlocked` signal to go high.
5. When the `rx_freqlocked` signal goes high (marker 7), from that point onwards, wait for at least 4 µs, then de-assert the `rx_digitalreset` signal (marker 8). At this point, the transmitter and receiver are ready for data traffic.

Receiver and Transmitter Channel—Receiver CDR in Manual Lock Mode

This configuration contains both a transmitter and receiver channel. If you create a **Receiver and Transmitter** instance in the ALTGX MegaWizard Plug-In Manager with the receiver CDR in Manual lock mode, use the reset sequence shown in Figure 4–11.

Figure 4–11. Sample Reset Sequence of Receiver and Transmitter Channel—Receiver CDR in Manual Lock Mode



Note to Figure 4–11:

(1) To be characterized.

As shown in Figure 4–11, perform the following reset procedure for the receiver in manual lock mode:

1. After power up, assert `PLL_powerdown` for a minimum period of 1 µs (the time between markers 1 and 2).
2. Keep the `tx_digitalreset`, `rx_analogreset`, `rx_digitalreset`, and `rx_locktorefclk` signals asserted and the `rx_locktodata` signal de-asserted during this time period. After you de-assert the `PLL_powerdown` signal, the transmitter PLL starts locking to the transmitter input reference clock.

3. After the transmitter PLL locks, as indicated by the pll_locked signal going high (marker 3), de-assert tx_digitalreset. For receiver operation, wait for the busy signal to be de-asserted. At this point rx_analogreset is de-asserted. Once rx_analogreset is de-asserted, the receiver CDR starts locking to the receiver input reference clock because rx_locktorefclk is asserted.
4. Wait for at least 15 µs (the time between markers 7 and 8) after the rx_pll_locked signal goes high, then de-assert the rx_locktorefclk signal. At the same time, assert the rx_locktodata signal (marker 8). At this point, the receiver CDR enters lock-to-data mode and the receiver CDR starts locking to the received data.
5. De-assert rx_digitalreset at least 4 µs (the time between markers 8 and 9) after asserting the rx_locktodata signal.

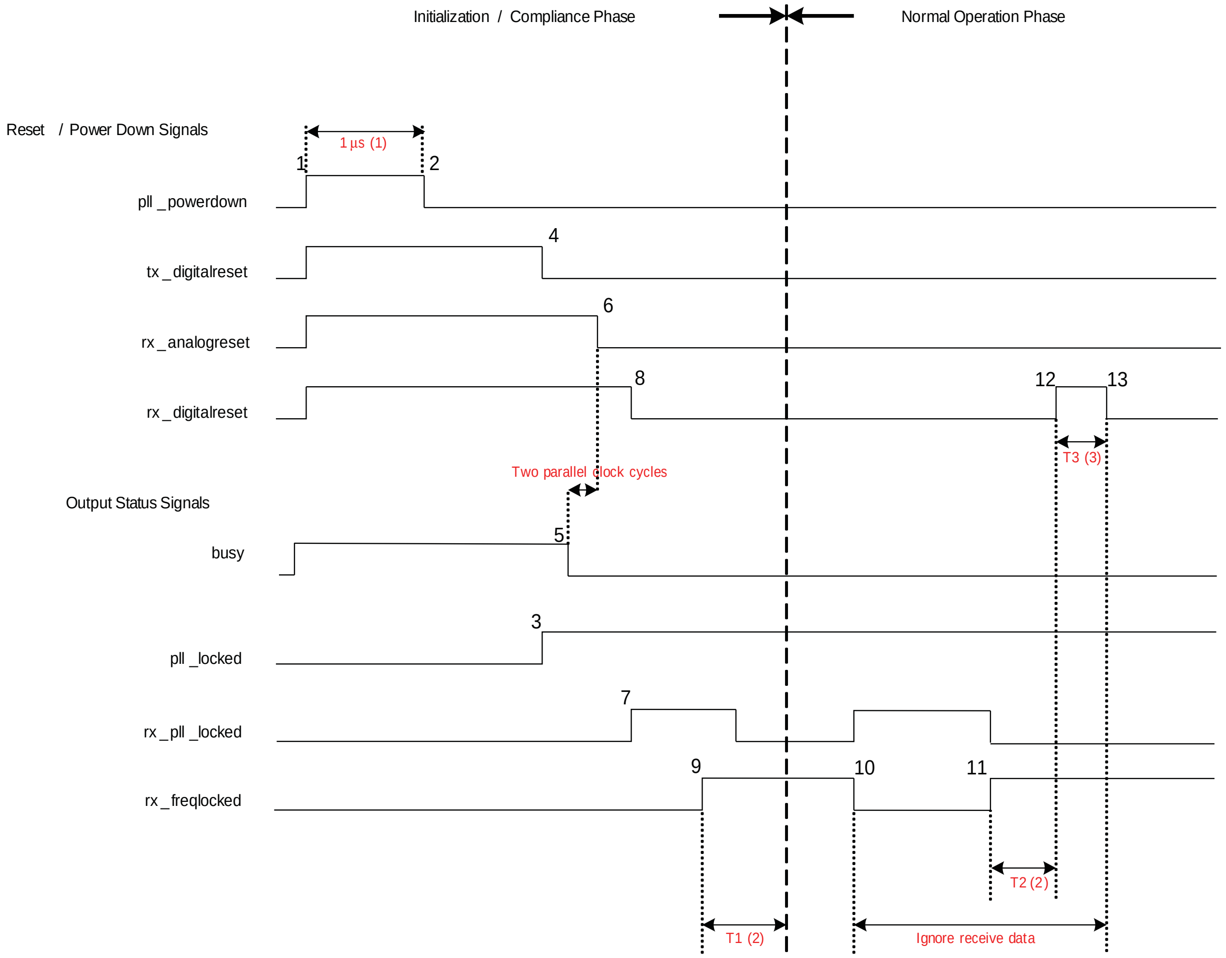
PCI Express (PIPE) Functional Mode

You can configure PCI Express (PIPE) functional mode with or without the receiver clock rate compensation FIFO in the Stratix IV GX device. The reset sequence remains the same irrespective of whether you use the receiver clock rate compensation FIFO or not.

PCI Express (PIPE) Reset Sequence

PCI Express (PIPE) protocol consists of an initialization/compliance phase and normal operation phase. The reset sequences for these two phases are described based on the timing diagram in Figure 4–12.

Figure 4–12. Reset Sequence of PCI Express (PIPE) Functional Mode



Notes to Figure 4–12:

(1) To be characterized.

(2) The minimum T1 and T2 period is 4 us.

(3) The minimum T3 period is two parallel clock cycles.

PCI Express (PIPE) Initialization/Compliance Phase

After the device is powered up, a PCI Express (PIPE)-compliant device goes through the compliance phase during initialization. In this phase, the PCI Express (PIPE) protocol requires the system to be operating at Gen 1 data rate. The `rx_digitalreset` signal must be de-asserted during this compliance phase to achieve transitions on the `pipephydonestatus` signal, as expected by the link layer. The `rx_digitalreset` signal is de-asserted based on the assertion of the `rx_freqlocked` signal.

During the initialization/compliance phase, do not use the rx_freqlocked signal to trigger a de-assertion of rx_digitalreset signal. Instead, perform the following reset sequence:

1. After power up, assert PLL_powerdown for a minimum period of 1 µs (the time between markers 1 and 2). Keep the tx_digitalreset, rx_analogreset, and rx_digitalreset signals asserted during this time period. After you de-assert the PLL_powerdown signal, the transmitter PLL starts locking to the transmitter input reference clock.
2. Once the transmitter PLL locks, as indicated by the pll_locked signal going high (marker 3), de-assert tx_digitalreset. For receiver operation, wait for the busy signal to be de-asserted. After which rx_analogreset is de-asserted. After rx_analogreset is de-asserted, the receiver CDR starts locking to the receiver input reference clock.
3. Once the receiver CDR locks to the input reference clock, as indicated by the rx_pll_locked signal going high at marker 7 in Figure 4–12, de-assert the rx_digitalreset signal (marker 8). After de-asserting rx_digitalreset, the pipephydonestatus signal transitions from the transceiver channel to indicate the status to the link layer. Depending on its status, pipephydonestatus helps with the continuation of the compliance phase. After successful completion of this phase, the device enters into the normal operation phase.

PCI Express Normal Phase

After completion of the Initialization/Compliance phase, during the normal operation phase at Gen 1 data rate, when the rx_freqlocked signal is de-asserted, (marker 10 in Figure 4–12), wait for the rx_pll_locked signal assertion signifying the lock-to-reference clock.

Next, wait for the rx_freqlocked signal to go high again. In this phase, the received data is valid (not electrical idle) and the receiver CDR locks to the incoming data. Therefore, proceed with the reset sequence after assertion of the rx_freqlocked signal. After the rx_freqlocked signal goes high, wait for at least 4 µs before asserting rx_digitalreset (marker 12 in Figure 4–12) for two parallel receive clock cycles so that the receiver phase compensation FIFO is initialized.

During normal operation, after you speed-negotiate to Gen 2 data rate, asserting the rx_digitalreset signal causes the PCI Express (PIPE) rate switch circuitry to switch the transceiver to Gen 1 data rate.

Data from the transceiver block is not valid from the time the rx_freqlocked signal goes low (marker 10 in Figure 4–12) to the time the rx_digitalreset is de-asserted (marker 13 in Figure 4–12). The PLD logic ignores the data during this period (between markers 10 and 13 in Figure 4–12).

☞ You can configure the Stratix IV GX device in ×1, ×4, and ×8 PIPE lane configurations. The reset sequence described in "PCI Express (PIPE) Reset Sequence" on page 4–18 applies to all these multi-lane configurations.

PMA Direct Drive Mode Reset Sequences

Stratix IV GX devices provide a PMA direct mode in which all the PCS blocks including the phase compensation FIFOs are bypassed in both the transmitter and receiver channel data paths. In this mode, the PMA block in the transmitter and receiver channels directly interface with the FPGA fabric.

In this mode, the transceiver channels can be configured as a single channel or in bonded configurations. Basic single and double width functional modes support bonding of PMA functional blocks across all transceiver channels on the same side of the device.

Since there are no PCS blocks available in this mode, `tx_digitalreset` and `rx_digitalreset` signals are not available.

PMA Direct Drive-xN Mode

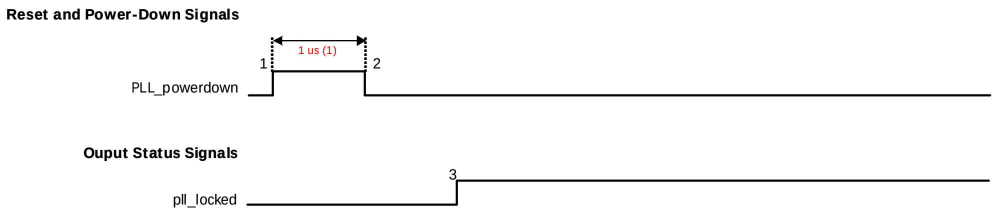
When bonding xN channels in PMA Direct Drive mode configuration, you can reset all the bonded channels simultaneously. There are three main configurations that describe the reset and power down sequences for Basic-PMA Direct Drive-xN functional mode:

- Transmitter only channel set-up
- Receiver and Transmitter channel set-up—receiver CDR in automatic lock mode
- Receiver and Transmitter channel set-up—receiver CDR in manual lock mode

Transmitter Only Channel

An example reset sequence timing diagram of four transmitter only channels in Basic-PMA Direct Drive-x4 functional mode is shown in Figure 4–13.

Figure 4–13. Reset Sequence Timing



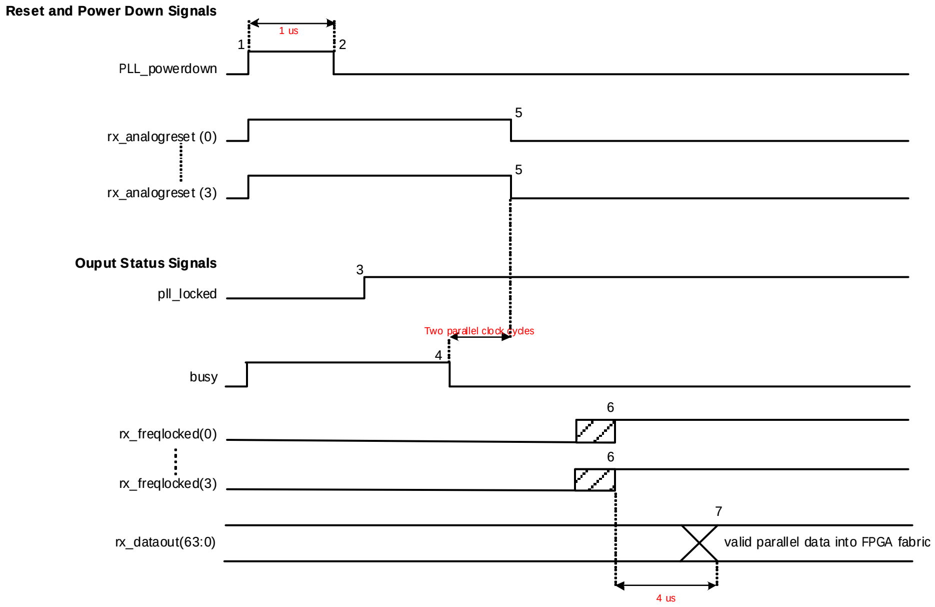
As shown in Figure 4–13, perform the following reset procedure for the **Transmitter only** channel in PMA direct drive mode configuration:

1. After power up, assert `PLL_powerdown` for a minimum of 1 us (the time between markers 1 and 2).
2. Once the transmitter PLL locks, as indicated by the `pll_locked` signal going high (marker 3), the transmitter is ready to accept parallel data from the FPGA fabric and subsequently transmitting serial data reliably.

Receiver and Transmitter Channel Set-up—Receiver CDR in Automatic Lock Mode

This configuration contains both a transmitter and receiver channel. For PMA direct drive-xN mode, with receiver CDR in automatic lock mode, use the reset sequence shown in Figure 4–14.

Figure 4–14. Reset Sequence in Automatic Lock Mode



As shown in Figure 4–14, perform the following reset procedure for the receiver and transmitter channel in PMA direct drive-x4 double width configuration with CDR in automatic lock mode:

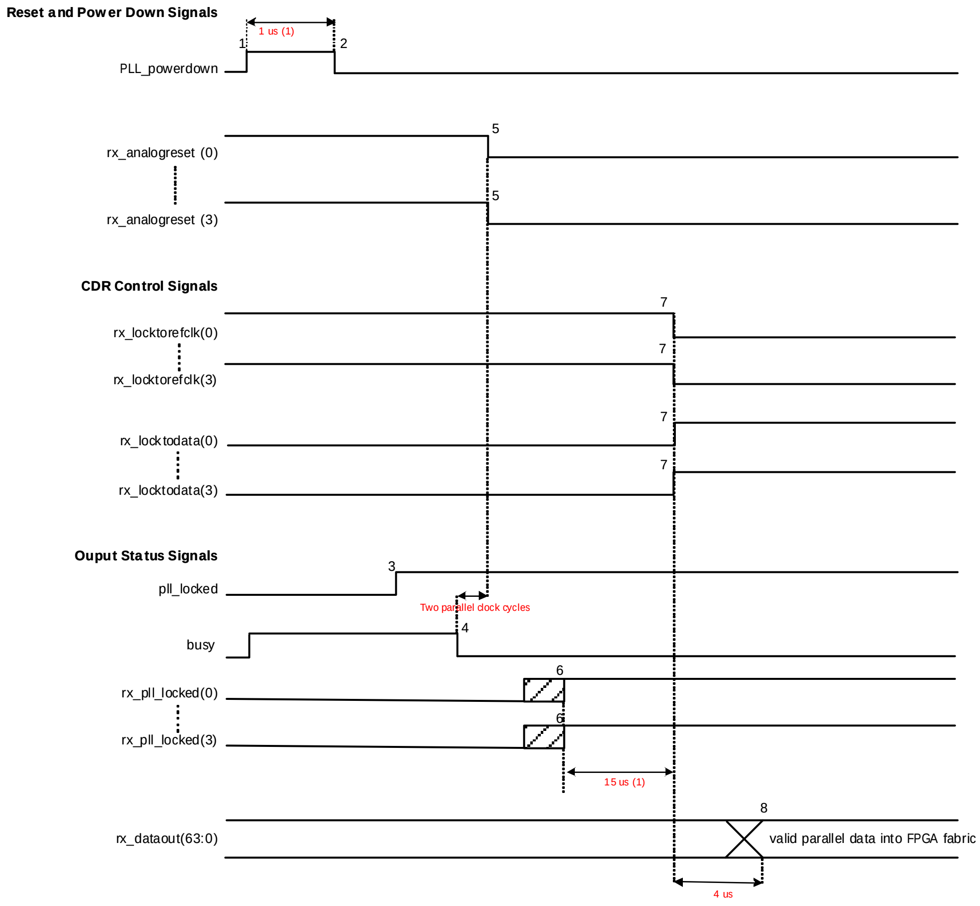
1. After power up, assert `PLL_powerdown` for a minimum period of 1 µs (the time between markers 1 and 2).
2. Keep the `rx_analogreset` signal asserted during this time period. After you de-assert the `PLL_powerdown` signal, the transmitter PLL starts locking to the transmitter input reference clock.
3. Once the transmitter PLL locks, as indicated by the `pll_locked` signal going high (marker 3), the transmitter is ready to accept parallel data from the FPGA fabric and subsequently transmitting serial data reliably.
4. For the receiver operation, after de-assertion of busy signal, wait for two parallel clock cycles to de-assert the `rx_analogreset` signals of each channel. After `rx_analogreset` is de-asserted, the receiver CDR of each channel starts locking to the receiver input reference clock.
5. Wait for the `rx_freqlocked` signal from each channel to go high. The `rx_freqlocked` signal of each channel may go high at different times (indicated by the slashed pattern at marker 6).

6. In a PMA direct drive-x4 double width configuration, when the `rx_freqlocked` signals of all the channels has gone high (marker 6), from that point onwards wait for at least 4 µs (marker 7) for the receiver parallel clock to become stable. At this point, all the receivers are ready for transferring valid parallel data into the FPGA fabric (until this time, it is recommended that the user logic that processes this data be under reset).

Receiver and Transmitter Channel Set-up—Receiver CDR in Manual Lock Mode

This configuration contains both a transmitter and receiver channel. For PMA direct drive-xN mode, with receiver CDR in manual lock mode, use the reset sequence shown in Figure 4–15.

Figure 4–15. Reset Sequence in Manual Lock Mode



As shown in Figure 4–15, perform the following reset procedure for the receiver and transmitter channel in PMA direct drive-x4 double width configuration with CDR in manual lock mode:

1. After power up, assert PLL_powerdown for a minimum period of 1 µs (the time between markers 1 and 2).

2. Keep the rx_analogreset and rx_locktorefclk signals asserted and the rx_locktodata signal de-asserted during this time period. After you de-assert the PLL_powerdown signal, the transmitter PLL starts locking to the transmitter input reference clock.
3. Once the transmitter PLL locks, as indicated by the pll_locked signal going high (marker 3), the transmitter is ready to accept parallel data from the FPGA fabric and subsequently transmitting serial data reliably.

   For the receiver operation, after de-assertion of the busy signal (marker 4), wait for two parallel clock cycles to de-assert the rx_analogreset signal. After the rx_analogreset signal is de-asserted, the receiver CDR of each channel starts locking to the receiver input reference clock because rx_locktorefclk is asserted.
4. Wait for the rx_pll_locked signal from each channel to go high. The rx_pll_locked signal of each channel may go high at different times with respect to each other (indicated by the slashed pattern at the marker 6).
5. In a PMA direct drive-x4 double width configuration, when the rx_pll_locked signal of all the channels have gone high, from that point onwards, wait for at least 15 µs, then de-assert rx_locktorefclk and assert rx_locktodata (marker 7). At this point, the receiver CDR of all the channels enters into lock-to-data mode and starts locking to the received data.
6. After assertion of rx_locktodata signal, from that point onwards, wait for at least 4 µs (marker 8) for the receiver parallel clock to become stable. At this point, all the receivers are ready for transferring valid parallel data into the FPGA fabric (until this time, it is recommended that the user logic that processes this data be under reset).

Non-Bonded Channel Configuration in PMA Direct Drive Mode

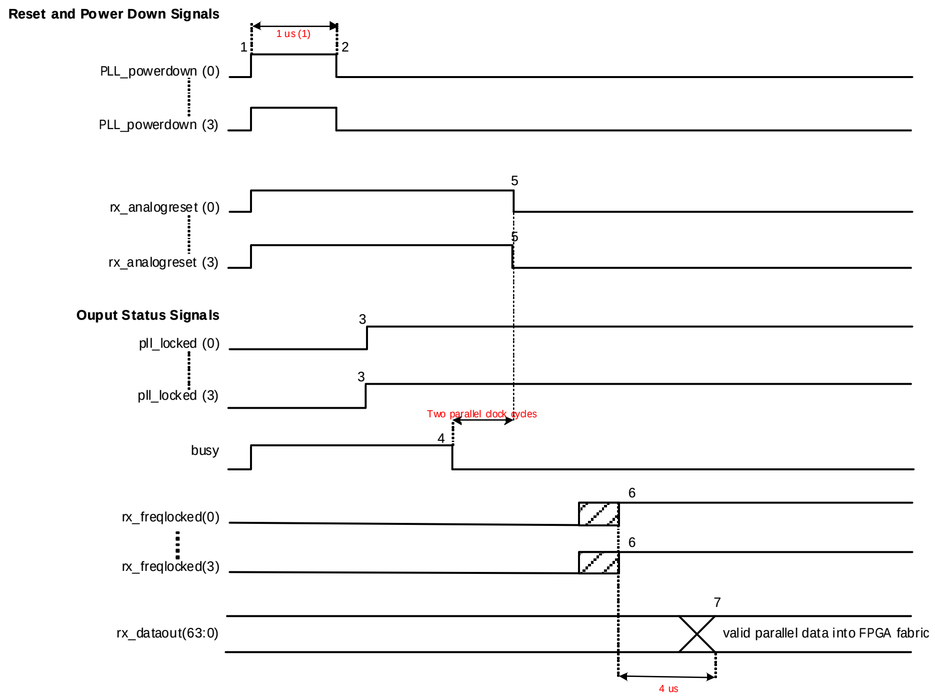
The following timing diagram examples are used to describe the reset and power down sequences for PMA direct drive mode without any bonding between the transceiver channels.

- Receiver and Transmitter channel set-up—receiver CDR in automatic lock mode
- Receiver and Transmitter channel set-up—receiver CDR in manual lock mode

Receiver and Transmitter Channel Set-Up—Receiver CDR in Automatic Lock Mode

This configuration contains both a transmitter and receiver channel. For PMA direct drive-xN mode, with receiver CDR in automatic lock mode, use the reset sequence shown in Figure 4–16.

Figure 4–16. Reset Sequence



As shown in Figure 4–16, perform the following reset procedure for the receiver and transmitter channel in PMA direct drive double width configuration, non-bonded with CDR in automatic lock mode:

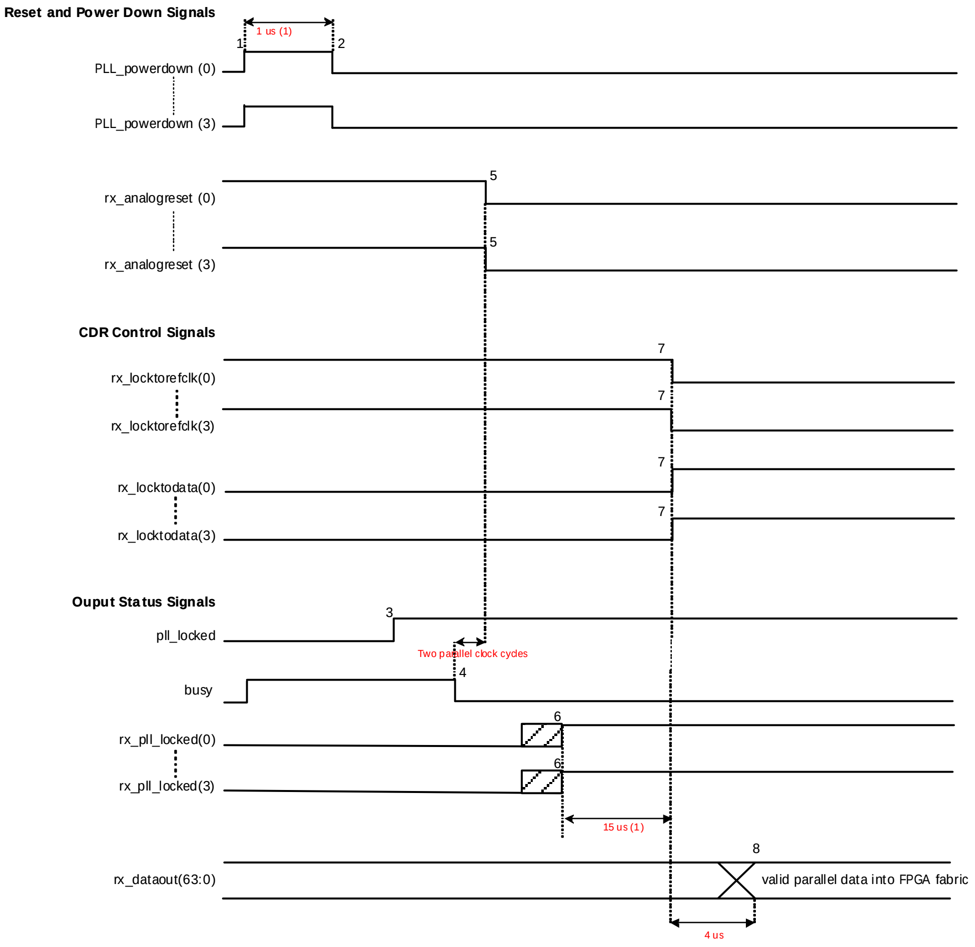
1. After power up, assert `PLL_powerdown` of each channel for a minimum period of 1 µs (the time between markers 1 and 2).
2. Keep the `rx_analogreset` signal of each channel asserted during this time period. After you de-assert the `PLL_powerdown` signal on all channels, the transmitter PLL of each channel starts locking to the transmitter input reference clock.
3. Once the transmitter PLL locks, as indicated by the `pll_locked` signal going high (marker 3), the transmitters are ready for accepting parallel data from the FPGA fabric and subsequently transmitting serial data reliably.
4. For the receiver operation, after de-assertion of busy signal, wait for two parallel clock cycles to de-assert the `rx_analogreset` signals of each channel. After `rx_analogreset` is de-asserted, the receiver CDR of each channel starts locking to the receiver input reference clock.
5. Wait for the `rx_freqlocked` signal from each channel to go high. The `rx_freqlocked` signal of each channel may go high at different times (indicated by the slashed pattern at marker 6).

6. In a PMA direct drive double width configuration without any bonding between channels, when the `rx_freqlocked` signals of all the channels has gone high (marker 6), from that point onwards, wait for at least 4 µs (marker 7) for the receiver parallel clock to become stable. At this point, all the receivers are ready for transferring valid parallel data into the FPGA fabric (until this time, it is recommended that the user logic that processes this data be under reset).

Receiver and Transmitter Channel Set-up—Receiver CDR in Manual Lock Mode

This configuration contains both a transmitter and receiver channel. For PMA direct drive-xN mode, with receiver CDR in manual lock mode, use the reset sequence shown in Figure 4–17.

Figure 4–17. Reset Sequence in Manual Lock Mode



As shown in Figure 4–17, perform the following reset procedure for the receiver and transmitter channel in PMA direct drive double width configuration, non-bonded with CDR in manual lock mode:

1. After power up, assert `PLL_powerdown` of each channel for a minimum period of 1 µs (the time between markers 1 and 2).

2. Keep the rx_analogreset and rx_locktorefclk signals of each channel asserted and the rx_locktodata signals de-asserted during this time period. After you de-assert the PLL_powerdown signal, the transmitter PLL starts locking to the transmitter input reference clock.
3. Once the transmitter PLL locks, as indicated by the pll_locked signal going high (marker 3), the transmitters are ready to accept parallel data from the FPGA fabric and subsequently transmitting serial data reliably.

   For the receiver operation, after de-assertion of the busy signal (marker 4), wait for two parallel clock cycles to de-assert the rx_analogreset signal of each channel. After the rx_analogreset signal is de-asserted, the receiver CDR of each channel starts locking to the receiver input reference clock because rx_locktorefclk is asserted.
4. Wait for the rx_pll_locked signal from each channel to go high. The rx_pll_locked signal of each channel may go high at different times with respect to each other (indicated by the slashed pattern at the marker 6).
5. In a PMA direct drive double width configuration without any bonding between channels, when the rx_pll_locked signal of all the channels have gone high, from that point onwards, wait for at least 15 µs, then de-assert rx_locktorefclk and assert rx_locktodata (marker 7). At this point, the receiver CDR of all the channels enters into lock-to-data mode and starts locking to the received data.
6. After assertion of rx_locktodata signal, from that point onwards wait for at least 4 µs (marker 8) for the receiver parallel clock to be stable. At this point, all the receivers are ready for transferring valid parallel data into the FPGA fabric (until this time, it is recommended that the user logic that processes this data be reset).

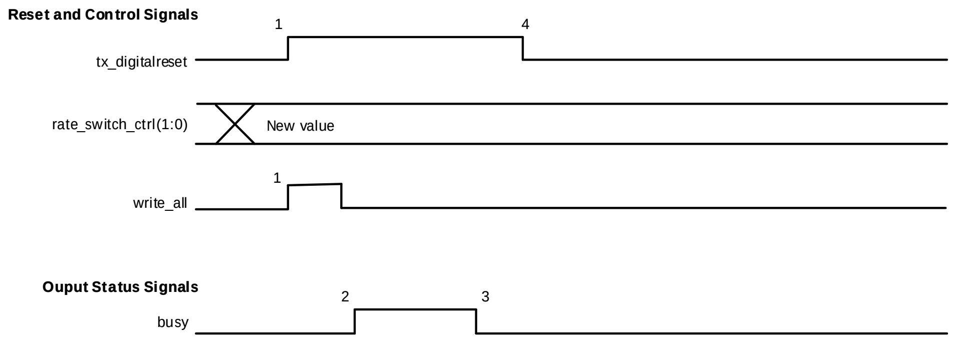
Dynamic Reconfiguration Reset Sequences

When using dynamic reconfiguration in data rate divisions in TX or channel and TX CMU PLL select/reconfig modes, follow the following reset sequences.

Reset Sequence when Using Dynamic Reconfiguration with 'data rate division in TX' Option

This section shows an example reset sequence to be followed when using the dynamic reconfig controller to change the data rate of the transceiver channel. In this example, the dynamic reconfiguration is used to dynamically reconfigure the data rate of the transceiver channel configured in Basic x1 mode with receiver CDR in automatic lock mode.

Figure 4–18. Reset Sequence



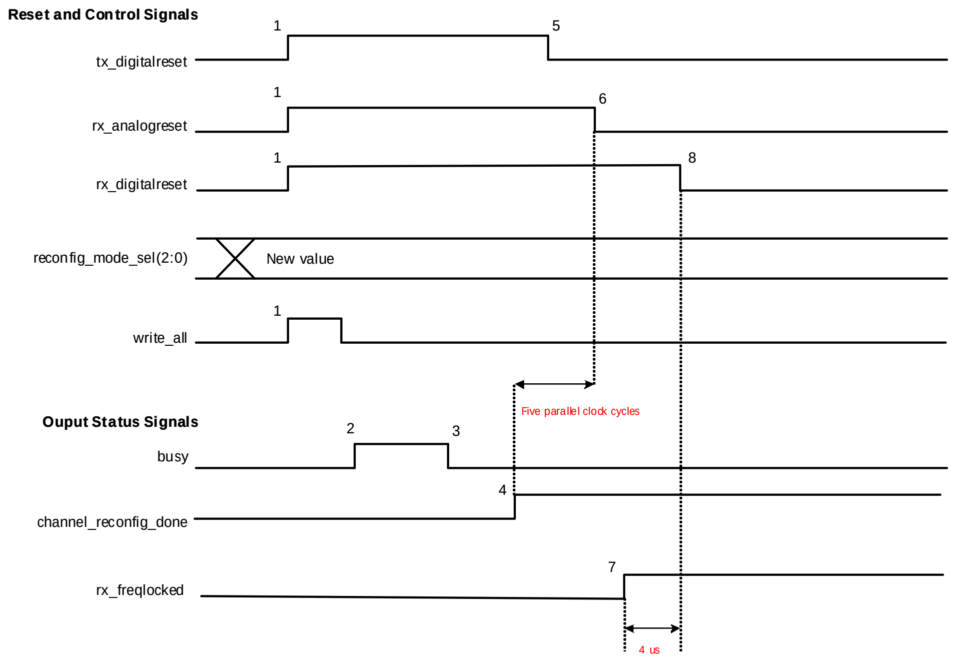
As shown in Figure 4–18, perform the following reset procedure when using the dynamic reconfig controller to change the configuration of the transmitter channel:

1. After power up and properly establishing that the transmitter is operating as desired, write the desired new value for a data rate in the appropriate register (in this example `rate_switch_ctrl[1:0]`) and subsequently assert `write_all` signal (marker 1) to initiate the dynamic reconfiguration. Refer to the *Stratix IV Dynamic Reconfiguration* chapter for additional information.
2. At this point, assert `tx_digitalreset` signal.
3. As soon as the `write_all` is asserted, the dynamic reconfiguration controller starts to execute its operation. This is indicated by the assertion of the busy (marker 2) signal.
4. After the completion of dynamic reconfiguration, the busy signal is de-asserted (marker 3).
5. Subsequently, the `tx_digitalreset` can be de-asserted to continue with transmitter operation (marker 4).

Reset Sequence when Using Dynamic Reconfiguration with 'Channel and TX PLL select/reconfig' Option

This section shows an example reset sequence to be followed when using the dynamic reconfig controller to change the TX PLL settings of the transceiver channel. In this example, the dynamic reconfiguration is used to dynamically reconfigure the data rate of the transceiver channel configured in Basic x1 mode with receiver CDR in automatic lock mode.

Figure 4–19. Reset Sequence in Basic x1 Mode with Receiver CDR in Automatic Lock Mode



As shown in Figure 4–19, perform the following reset procedure when using the dynamic reconfig controller to change the configuration of the transceiver channel:

1. After power up and properly establishing that the transceiver is operating as desired, write the desired new value in the appropriate registers including `reconfig_mode_sel[2:0]` and subsequently assert `write_all` signal (marker 1) to initiate the dynamic reconfiguration. Refer to the *Stratix IV Dynamic Reconfiguration* chapter for additional information.

2. At this point, assert `tx_digitalreset`, `rx_analogreset`, and `rx_digitalreset` signals.

3. As soon as the `write_all` is asserted the dynamic reconfiguration controller starts to execute its operation. This is indicated by the assertion of the busy (marker 2) signal.

4. Subsequently, wait for the assertion of `channel_reconfig_done` signal (marker 4) which indicates the completion of dynamic reconfiguration in this mode.

5. After assertion of `channel_reconfig_done` signal, de-assert `tx_digitalreset` (marker 5) and wait for at least five parallel clock cycles to de-assert `rx_analogreset` signal (marker 6).

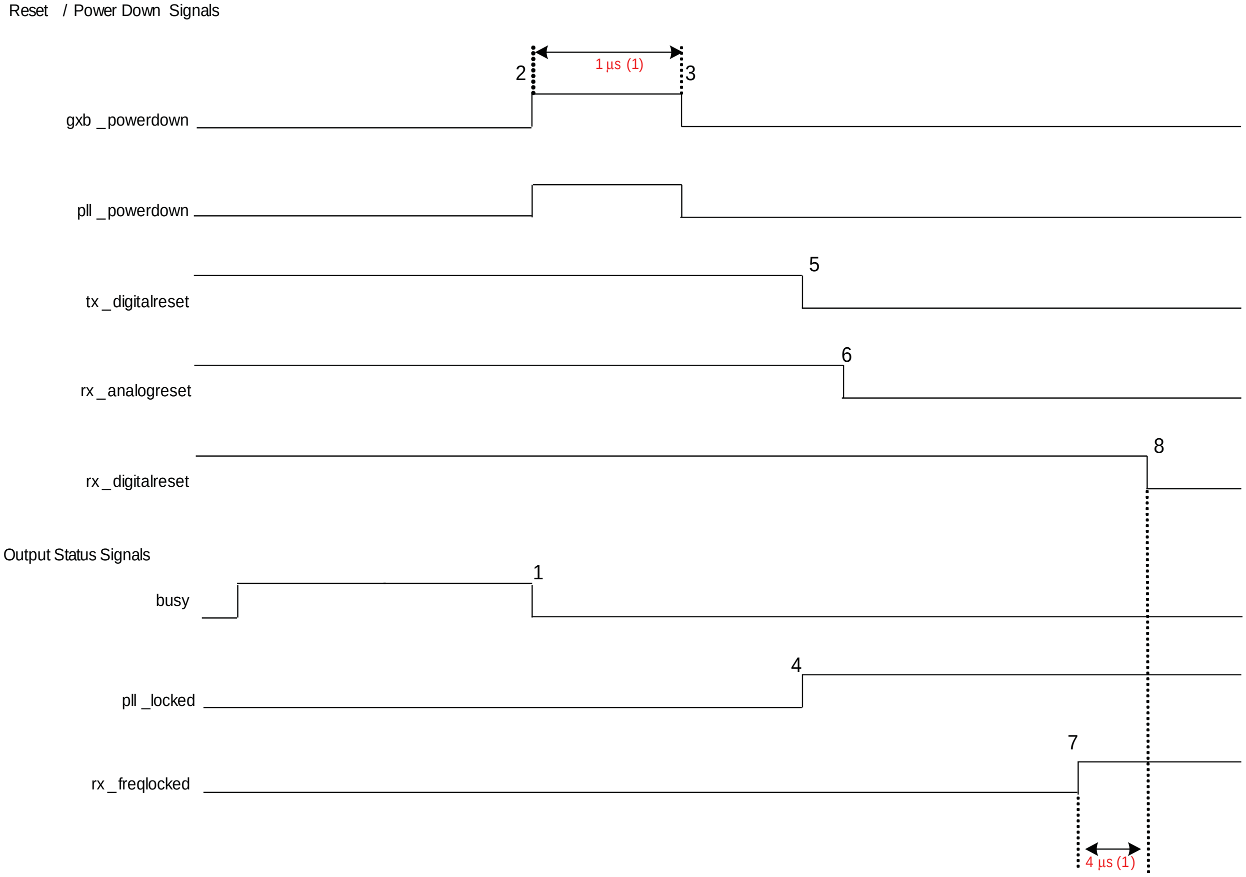
6. Subsequently, wait for `rx_freqlocked` signal to go high. After `rx_frqlocked` goes high (marker 7), wait for 4 µs to de-assert the `rx_digitalreset` signal (marker 8). At this point, the receiver is ready for data traffic.

Power Down

The Quartus II software automatically selects the power-down channel feature, which takes effect when you configure the Stratix IV GX device. All unused transceiver channels and blocks are powered down to reduce overall power consumption. The `gxb_powerdown` signal is an optional transceiver block signal. It powers down all the blocks in the transceiver block. The minimum pulse width for this signal is 1 µs. After power up, if you use the `gxb_powerdown` signal, wait for de-assertion of the `busy` signal, then assert the `gxb_powerdown` signal for a minimum of 1µs. Subsequently follow the sequence in Figure 4–20.

The de-assertion of the busy signal indicates proper completion of the offset cancellation process on the receiver channel.

Figure 4–20. Sample Reset Sequence of Four Receiver and Transmitter Channels-Receiver CDR in Automatic Lock Mode with Optional gxb_powerdown Signal



Notes to Figure 4–20:

(1) To be characterized.

Simulation Requirements

The following are simulation requirements:

- The gxb_powerdown port is optional. In simulation, if the gxb_powerdown port is not instantiated, you must assert the tx_digitalreset, rx_digitalreset, and rx_analogreset signals appropriately for correct simulation behavior.
- If the gxb_powerdown port is instantiated, and the other reset signals are not used, you must assert the gxb_powerdown signal for at least one parallel clock cycle for correct simulation behavior.
- You can de-assert the rx_digitalreset signal immediately after the rx_freqlocked signal goes high to reduce the simulation run time. It is not necessary to wait for 4 µs (as suggested in the actual reset sequence).
- The busy signal is de-asserted after about 20 parallel reconfig_clk clock cycles in order to reduce the simulation run time. For silicon behavior in hardware, the reset sequences discussed in the previous pages can be followed.
- In PCI Express (PIPE) mode simulation, you must assert the tx_forceelecidle signal for at least one parallel clock cycle before transmitting normal data for correct simulation behavior.

Documents Referenced

This chapter references the following documents:

- *Stratix IV Dynamic Reconfiguration* chapter in volume 4 of the the *Stratix IV Device Handbook*

Document Revision History

Table 4–3 shows the revision history for this chapter.

Table 4–3. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
February 2009, v3.0	Added “PMA Direct Drive Mode Reset Sequences” and “Dynamic Reconfiguration Reset Sequences” sections.	—
November 2008, v2.0	Added chapter to the Stratix IV Device Handbook.	—

5. Stratix IV Dynamic Reconfiguration

SIV52005-1.0

Introduction

Dynamic reconfiguration is a feature available for Stratix® IV GX transceivers. Each transceiver channel has multiple physical medium attachment (PMA) controls that can be programmed to achieve the desired bit error ratio (BER) for your system. When you enable the dynamic reconfiguration feature, you can modify the various PMA controls without powering down other transceiver channels or the FPGA fabric logic of the device.

Dynamic Reconfiguration Modes

The different modes of dynamic reconfiguration are PMA controls reconfiguration and offset cancellation.

PMA Controls Reconfiguration

You can dynamically reconfigure the following PMA controls:

- Pre-emphasis settings
- Equalization settings
- DC gain settings
- Voltage output differential (VOD) settings

Refer to "PMA Controls Reconfiguration" on page 5–37 for additional information.

Offset Cancellation

The Stratix IV GX device provides an offset cancellation circuit per receiver channel to counter the offset variations due to process, voltage, and temperature. These variations create an offset in the analog circuit voltages, pushing them out of the expected range. In addition to reconfiguring the transceiver channel, the dynamic reconfiguration controller performs offset cancellation on all the receiver channels connected to it on power up.

The **Offset cancellation for receiver channels** option is automatically enabled in both the ALTGX and ALTGX_RECONFIG MegaWizard® Plug-In Managers. for **Receiver and Transmitter** and **Receiver only** configurations. It is not available for **Transmitter only** configurations. For **Receiver and Transmitter** and **Receiver only** configurations, you must connect the necessary interface signals between the ALTGX_RECONFIG and ALTGX (with receiver channels) instances. Refer to "Offset Cancellation Control for Receiver Channels" on page 5–31 for additional information.

☞ For proper device operation, you must always connect the ALTGX_RECONFIG and ALTGX (with receiver channels) instances.

Conventions Used in this Chapter

The following conventions are used throughout this document:

- ALTGX_RECONFIG Instance

  This term represents the dynamic reconfiguration controller instance generated by the ALTGX_RECONFIG MegaWizard Plug-In Manager. This term is used when the various inputs, outputs and connections to the controller are explained.

- Dynamic Reconfiguration Controller

  This term represents the dynamic reconfiguration controller. This term is used when a concept related to the controller is explained.

- ALTGX Instance

  This term represents the transceiver instance generated by the ALTGX MegaWizard Plug-In Manager. This term is used when the various inputs, outputs and connections to the transceiver channels are explained.

- Logical Channel Addressing

  This term is used whenever the concept of logical channel addressing is explained. This term does not refer to the `logical_channel_address` port or the **Use 'logical_channel_address' port** option available in the ALTGX_RECONFIG MegaWizard Plug-In Manager.

- PMA controls

  This term represents the **Analog controls** (VOD, Pre-emphasis, Manual Equalization) as displayed in both the ALTGX and ALTGX_RECONFIG MegaWizard Plug-In Managers

Quartus II MegaWizard Plug-In Manager Interfaces to Support Dynamic Reconfiguration

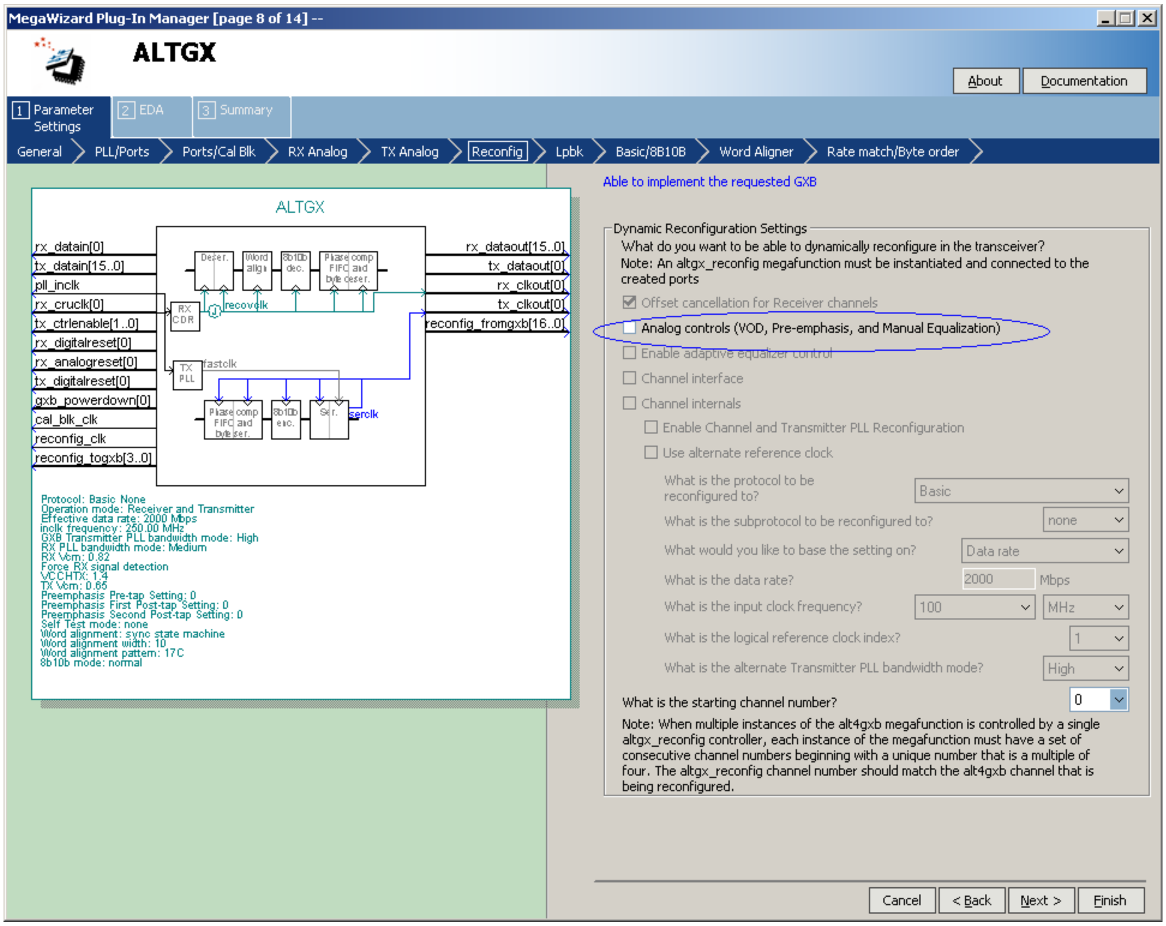
The Stratix IV GX device provides two MegaWizard Plug-In Manager interfaces to support dynamic reconfiguration: ALTGX and ALTGX_RECONFIG.

This section provides information about the dynamic reconfiguration options available in the ALTGX and ALTGX_RECONFIG MegaWizard Plug-In Managers to perform PMA controls reconfiguration on transceiver channels.

ALTGX MegaWizard Plug-In Manager

The ALTGX MegaWizard Plug-In Manager provides the **Analog controls (VOD, Pre-emphasis, and Manual Equalization)** option in the **Reconfig** screen, as shown in Figure 5–1. Select this option to enable PMA controls reconfiguration on the transceiver channels.

Figure 5–1. Dynamic Reconfiguration Settings in the ALTGX MegaWizard Plug-In Manager



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Dynamic Reconfiguration Settings

What do you want to be able to dynamically reconfigure in the transceiver?
Note: An altgx_reconfig megafunction must be instantiated and connected to the created ports

☑ Offset cancellation for Receiver channels
☐ Analog controls (VOD, Pre-emphasis, and Manual Equalization)
☐ Enable adaptive equalizer control
☐ Channel interface
☐ Channel internals
☐ Enable Channel and Transmitter PLL Reconfiguration
☐ Use alternate reference clock

What is the protocol to be reconfigured to? Basic
What is the subprotocol to be reconfigured to? none
What would you like to base the setting on? Data rate
What is the data rate? 2000 Mbps
What is the input clock frequency? 100 MHz
What is the logical reference clock index? 1
What is the alternate Transmitter PLL bandwidth mode? High

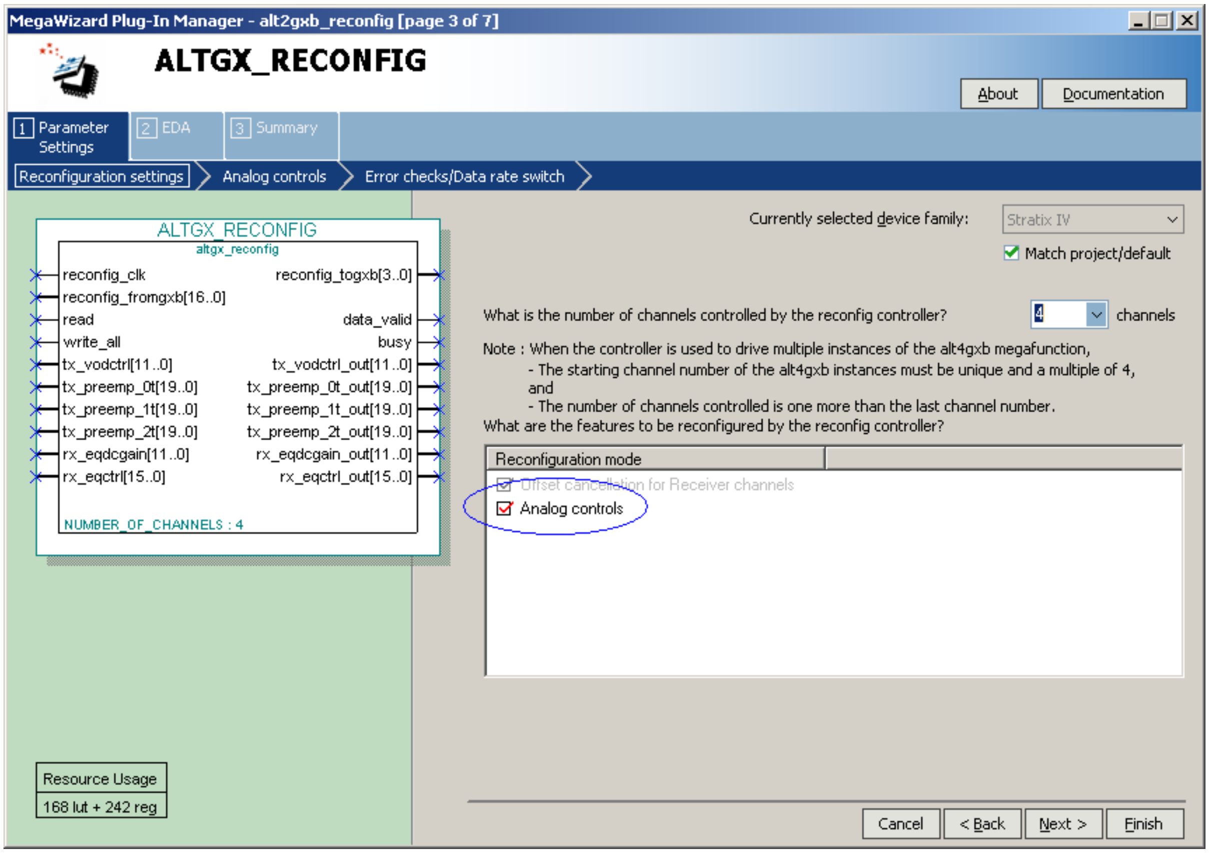
What is the starting channel number? 0

Note: When multiple instances of the alt4gxb megafunction is controlled by a single altgx_reconfig controller, each instance of the megafunction must have a set of consecutive channel numbers beginning with a unique number that is a multiple of four. The altgx_reconfig channel number should match the alt4gxb channel that is being reconfigured.

ALTGX_RECONFIG MegaWizard Plug-In Manager

The Quartus® II software provides the ALTGX_RECONFIG MegaWizard Plug-In Manager to instantiate the dynamic reconfiguration controller. To dynamically reconfigure the PMA controls, enable the **Analog controls** option in the **Reconfiguration settings** screen, as shown in Figure 5–2, and enable at least one of the PMA control ports in the **Analog controls** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager.

Figure 5–2. Dynamic Reconfiguration Settings in the ALTGX_RECONFIG MegaWizard Plug-In Manager



Currently selected device family: Stratix IV

☑ Match project/default

What is the number of channels controlled by the reconfig controller? 4 channels

Note : When the controller is used to drive multiple instances of the alt4gxb megafunction,
- The starting channel number of the alt4gxb instances must be unique and a multiple of 4, and
- The number of channels controlled is one more than the last channel number.

What are the features to be reconfigured by the reconfig controller?

Reconfiguration mode

☑ Offset cancellation for Receiver channels

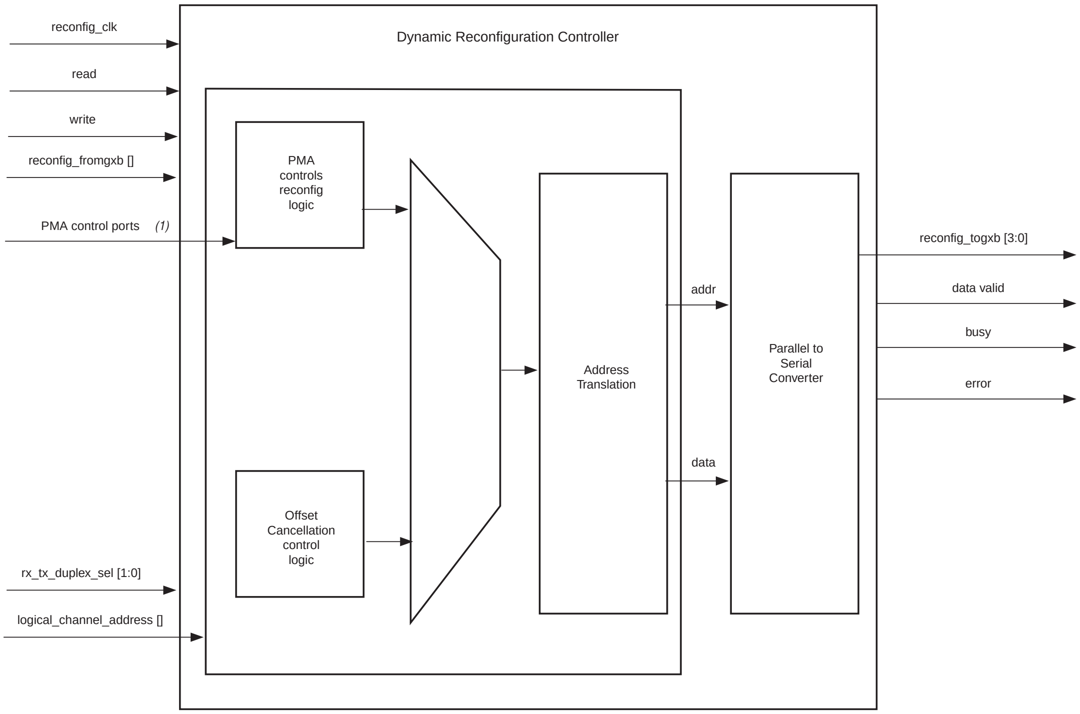
☑ Analog controls

The ALTGX_RECONFIG instance does not have backward compatibility with Stratix II GX devices or Stratix GX devices.

Dynamic Reconfiguration Controller Architecture

The dynamic reconfiguration controller is a soft IP which utilizes the FPGA-fabric resources. You can use only one controller per transceiver block. You cannot use the dynamic reconfiguration controller to control multiple Stratix IV GX devices or any off-chip interfaces. Figure 5–3 shows the conceptual view of the dynamic reconfiguration controller architecture

Figure 5–3. Block Diagram of the Dynamic Reconfiguration Controller



Notes to Figure 5–3:

(1) The PMA control ports consist of the VOD controls, Pre-emphasis controls, DC gain controls and Manual Equalization controls. Refer to "Dynamic Reconfiguration Controller Port List" on page 5–7 for the detailed description of all the inputs and outputs of the ALTGX_RECONFIG instance.

The dynamic reconfiguration controller comprises of two control logic modules:

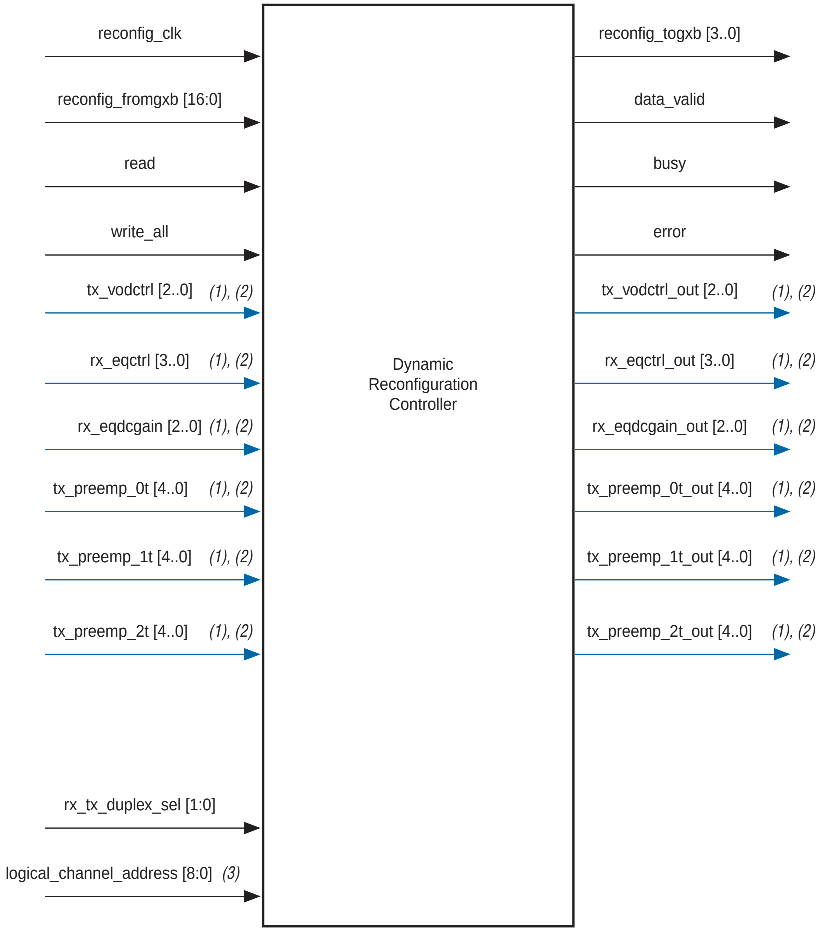
- PMA controls reconfiguration control logic
- Offset cancellation control logic for receiver channels

The dynamic reconfiguration control inputs to the controller are translated into an address bus and data bus within. The address bus and data bus are then converted into serial data and forwarded to the selected transceiver channel.

Dynamic Reconfiguration Controller Interface

The dynamic reconfiguration controller interface consists of certain control inputs and outputs certain status signals. Figure 5–4 shows the dynamic reconfiguration interface list, which comprises of all the inputs and outputs to the dynamic reconfiguration controller.

Figure 5–4. Dynamic Reconfiguration Controller Interface


reconfig_clk
reconfig_fromgxb [16:0]
read
write_all
tx_vodctrl [2..0] (1), (2)
rx_eqctrl [3..0] (1), (2)
rx_eqdcgain [2..0] (1), (2)
tx_preemp_0t [4..0] (1), (2)
tx_preemp_1t [4..0] (1), (2)
tx_preemp_2t [4..0] (1), (2)
rx_tx_duplex_sel [1:0]
logical_channel_address [8:0] (3)
Dynamic Reconfiguration Controller
reconfig_togxb [3..0]
data_valid
busy
error
tx_vodctrl_out [2..0] (1), (2)
rx_eqctrl_out [3..0] (1), (2)
rx_eqdcgain_out [2..0] (1), (2)
tx_preemp_0t_out [4..0] (1), (2)
tx_preemp_1t_out [4..0] (1), (2)
tx_preemp_2t_out [4..0] (1), (2)

Notes to Figure 5–4:

(1) These ports assume that the dynamic reconfiguration controller is connected to a single channel in the design.

(2) These are the optional PMA control input signals and the optional PMA control output status signals. You must select at least one of these PMA control ports if you want to dynamically configure the PMA controls of a transceiver channel. Refer to “Dynamic Reconfiguration Controller Port List” on page 5–7 for the detailed description of all the inputs and outputs of the ALTGX_RECONFIG instance.

(3) The `logical_channel_address` port is available for selection only when the number of channels controlled by the dynamic reconfiguration controller is more than one. It is shown in Figure 5–4 to represent the complete port list.

Dynamic Reconfiguration Controller Port List

Table 5–1 describes the input control ports and output status ports of the dynamic reconfiguration controller.

Table 5–1. Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 1 of 5)

Port Name	Input/ Output	Description
Clock Inputs to ALTGX_RECONFIG Instance		
`reconfig_clk`	Input	The frequency range of this clock depends on the following transceiver channel configuration modes: ■ Receiver only (37.5 MHz to 50 MHz) ■ Recevier and Transmitter (37.5 MHz to 50 MHz) ■ Transmitter only (2.5 MHz to 50 MHz) Refer to Table 5–3 on page 5–12 for additional details. By default, the Quartus II software assigns a global clock resource to this port.
ALTGX - ALTGX_RECONFIG Interface Signals		
`reconfig_fromgxb`	Input	The width of this signal is determined by the value you set in the **What is the number of channels controlled by the reconfig controller?** option in the **Reconfiguration settings** screen. Refer to "Connecting reconfig_from_gxb/reconfig_to_gxb Ports" on page 5–28 for additional details.
`reconfig_togxb[3..0]`	Output	The width of this signal is fixed to four bits. It is independent of the value you set in the **What is the number of channels controlled by the reconfig controller?** option in the **Reconfiguration settings** screen. Refer to "Connecting reconfig_from_gxb/reconfig_to_gxb Ports" on page 5–28 for additional details.
FPGA Fabric - ALTGX_RECONFIG Interface Signals		
`write_all`	Input	Assert this signal for one `reconfig_clk` clock cycle to initiate a write transaction from the ALTGX_RECONFIG instance to the ALTGX instance. Refer to "Dynamically Reconfiguring PMA Controls" on page 5–38 for additional information.
`busy`	Output	This signal is used to indicate the busy status of the dynamic reconfiguration controller during two conditions: ■ Dynamic Reconfiguration of PMA Controls This signal is high when the dynamic reconfiguration controller performs a read or write transaction. ■ Offset Cancellation After the device powers up, this signal remains low for the first `reconfig_clk` clock cycle. It then gets asserted and remains high when the dynamic reconfiguration controller performs offset cancellation on all the receiver channels connected to the ALTGX_RECONFIG instance. The de-assertion of the `busy` signal indicates the successful completion of the offset cancellation process. Refer to "Operation" on page 5–31 for additional details.

Table 5–1. Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 2 of 5)

Port Name	Input/ Output	Description
`read`	Input	Assert this signal for one `reconfig_clk` clock cycle to initiate a read transaction. The `read` port is available when you select **Analog controls** in the **Reconfiguration settings** screen and select atleast one of the PMA control ports in the **Analog controls** screen. Refer to "Dynamically Reconfiguring PMA Controls" on page 5–38 for additional information.
`data_valid`	Output	It indicates the validity of the data read from the transceiver by the dynamic reconfiguration controller. ■ ONLY if `data_valid` is high, the current data on the output read ports is the valid data. This signal gets enabled when you enable at least one PMA control port used in read transactions, for example `tx_vodctrl_out`.
`error`	output	It indicates that an unsupported operation is attempted. It is available for selection in the **Error checks/Data rate switch** screen. The dynamic reconfiguration controller de-asserts the `busy` signal and asserts the `error` signal for two `reconfig_clk` cycles when you attempt an unsupported operation. Refer to the "Error Indication in the ALTGX_RECONFIG MegaWizard Plug-In Manager" on page 5–56 for additional information.
`logical_channel_address [8:0]`		The `logical_channel_address` port is enabled by the ALTGX_RECONFIG MegaWizard Plug-In Manager, when you enable the **Use 'logical_channel_address' port** option in the **Analog controls** screen. The width of the `logical_channel_address` port depends on the value you set in the **What is the number of channels controlled by the reconfig controller?** in the **Reconfiguration settings** screen. The `logical_channel_address` port can be enabled only when the number of channels controlled by the dynamic reconfiguration controller is more than one.
`rx_tx_duplex_sel [1:0]`	Input	This is a 2-bit wide signal. It is available for selection in the **Error checks/Data rate switch** screen. The advantage of using this optional port is as follows: This enables the user to reconfigure only the transmitter portion of a channel, even if the channel configuration is duplex. For a setting of: ■ `rx_tx_duplex_sel` = 2'b00 => transmitter and receiver portion of the channel is reconfigured. ■ `rx_tx_duplex_sel` = 2'b01 => receiver portion of the channel is reconfigured. ■ `rx_tx_duplex_sel` = 2'b10 => transmitter portion of the channel is reconfigured.
Analog Settings Control/Status Signals		

Table 5–1. Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 3 of 5)

Port Name	Input/ Output	Description
`tx_vodctrl` *(1)*	Input	This is an optional transmit buffer VOD control signal. It is 3-bits per transmitter channel. The number of settings varies based on the transmit buffer supply setting and the termination resistor setting on the **TX Analog** screen of the ALTGX MegaWizard Plug-In Manager. The width of this signal is fixed to 3-bits if you enable either the **Use 'logical_channel_address' port** option or the **Use same control signal for all the channels** option in the **Analog controls** screen. Otherwise, the width of this signal is 3-bits per channel. Refer to "Dynamically Reconfiguring PMA Controls" on page 5–38 for additional details. The following shows the VOD values corresponding to the `tx_vodctrl` settings for 100 Ω termination. Refer to the *Programmable Output Differential Voltage* section of *Stratix IV Transceiver Architecture* chapter for additional details. tx_vodctrl VOD (mV) for 1.4 V V_{CCH} 000 200 001 400 010 600 011 700 100 800 101 900 110 1000 111 1200
`tx_preemp_0t` *(1)*	Input	This is an optional pre-emphasis control for pretap for the transmit buffer. Depending on what value you set at this input, the controller dynamically writes the value to the pre-emphasis control register of the transmit buffer. This signal controls both pre-emphasis positive and its inversion. The width of this signal is fixed to 5-bits if you enable either the **Use 'logical_channel_address' port** option or the **Use same control signal for all the channels** option in the **Analog controls** screen. Otherwise, the width of this signal is 5-bits per channel. Refer to "Dynamically Reconfiguring PMA Controls" on page 5–38 for additional details. The following values are the legal settings allowed for this signal: 0 represents 0 1-15 represents -15 to -1 16 represents 0 17-31 represents 1 to 15 Refer to the *Programmable Pre-Emphasis* section of the *Stratix IV Transceiver Architecture* chapter for additional details.

Table 5–1. Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 4 of 5)

Port Name	Input/ Output	Description
`tx_preemp_1t` *(1)*	Input	This is an optional pre-emphasis write control for first post tap for the transmit buffer. Depending on what value you set at this input, the controller dynamically writes the value to the first post tap control register of the transmit buffer. The width of this signal is fixed to 5-bits if you enable either the **Use 'logical_channel_address' port** option or the **Use same control signal for all the channels** option in the **Analog controls** screen. Otherwise, the width of this signal is 5-bits per channel. Refer to "Dynamically Reconfiguring PMA Controls" on page 5–38 for additional details, and the *Programmable Pre-Emphasis* section of the *Stratix IV Transceiver Architecture* chapter to understand the pre-emphasis feature.
`tx_preemp_2t` *(1)*	Input	This is an optional pre-emphasis write control for second post tap for the transmit buffer. This signal controls both pre-emphasis positive and its inversion. Depending on what value you set at this input, the controller dynamically writes the value to the pre-emphasis control register of the transmit buffer. This signal controls both pre-emphasis positive and its inversion. The width of this signal is fixed to 5-bits if you enable either the **Use 'logical_channel_address' port** option or the **Use same control signal for all the channels** option in the **Analog controls** screen. Otherwise, the width of this signal is 5-bits per channel. Refer to "Dynamically Reconfiguring PMA Controls" on page 5–38 for additional details. The following values are the legal settings allowed for this signal: 0 represents 0 1-15 represents -15 to -1 16 represents 0 17-31 represents 1 to 15 Refer to the *Programmable Pre-Emphasis* section of the *Stratix IV Transceiver Architecture* chapter to understand the pre-emphasis feature.
`rx_eqctrl` *(1)*	Input	This is an optional write control to write an equalization control value for the receive side of the PMA The width of this signal is fixed to 4-bits if you enable either the **Use 'logical_channel_address' port** option or the **Use same control signal for all the channels** option in the **Analog controls** screen. Otherwise, the width of this signal is 4-bits per channel. Refer to "Dynamically Reconfiguring PMA Controls" on page 5–38 and the *Programmable Equalization and DC Gain* section of the *Stratix IV Transceiver Architecture* chapter for additional details.

Table 5–1. Dynamic Reconfiguration Controller Port List (ALTGX_RECONFIG Instance) (Part 5 of 5)

Port Name	Input/ Output	Description
`rx_eqdcgain` *(1) (2)*	Input	This is an optional equalizer DC gain write control The width of this signal is fixed to 3-bits if you enable either the **Use 'logical_channel_address' port** option or the **Use same control signal for all the channels** option in the **Analog controls** screen. Otherwise, the width of this signal is 3-bits per channel. Refer to "Dynamically Reconfiguring PMA Controls" on page 5–38 for additional details. The following values are the legal settings allowed for this signal: 000 => 0 dB 001 => 3 dB 010 => 6 dB 011 => 9 dB 100 => 12 dB All other values => N/A Refer to the *Programmable Equalization and DC Gain* section of the *Stratix IV Transceiver Architecture* chapter for additional details
`tx_vodctrl_out`	Output	This is an optional transmit VOD read control signal. This signal reads out the value written into the VOD control register. The width of this output signal depends on the number of channels controlled by the dynamic reconfiguration controller.
`tx_preemp_0t_out`	Output	This is an optional pre-tap, pre-emphasis read control signal. This signal reads out the value written by its input control signal. The width of this output signal depends on the number of channels controlled by the dynamic reconfiguration controller.
`tx_preemp_1t_out`	Output	This is an optional first post-tap, pre-emphasis read control signal. This signal reads out the value written by its input control signal. The width of this output signal depends on the number of channels controlled by the dynamic reconfiguration controller.
`tx_preemp_2t_out`	Output	This is an optional second post-tap pre-emphasis read control signal. This signal reads out the value written by its input control signal. The width of this output signal depends on the number of channels controlled by the dynamic reconfiguration controller.
`rx_eqctrl_out`	Output	This is an optional read control signal to read the setting of equalization setting of the ALTGX instance. The width of this output signal depends on the number of channels controlled by the dynamic reconfiguration controller.
`rx_eqdcgain_out`	Output	This is an optional equalizer DC gain read control signal. This signal reads out the settings of the ALTGX instance DC gain. The width of this output signal depends on the number of channels controlled by the dynamic reconfiguration controller.

Notes to Table 5–1:

(1) Not all combinations of the input bits are legal values.

(2) In PCI Express (PIPE) mode, this input should be tied to 001 to be PCI E-compliant.

Clock Requirements for the ALTGX instance and ALTGX_RECONFIG instance

This section describes the dynamic reconfiguration clock requirements for both the ALTGX instance (transceiver instance) and the ALTGX_RECONFIG instance (dynamic reconfiguration controller instance).

Clock Requirements for ALTGX Instance

For all the functional mode configurations except PCI Express (PIPE) configurations, you must connect the `reconfig_clk` input port of the ALTGX instance to the same clock that is connected to the `reconfig_clk` input port of the ALTGX_RECONFIG instance.

For the PCI Express (PIPE) configurations of the ALTGX instance, the `fixedclk` must be used to clock the dynamic reconfiguration process instead of the `reconfig_clk`. Table 5–2 shows the range of frequency values for the `reconfig_clk` and `fixedclk` input ports.

Table 5–2. Dynamic Reconfiguration Clock Settings for ALTGX Instance *(Note 1)*

Clock Input	Frequency Range
`reconfig_clk`	37.5 MHz to 50 MHz
`fixedclk` (only for PCI Express [PIPE] configurations)	125 MHz

Note to Table 5–2:

(1) Altera recommends the `reconfig_clk` signal and `fixedclk` signal be driven on a global clock resource.

Clock Requirements for ALTGX_RECONFIG Instance

You must connect the `reconfig_clk` input port of the ALTGX_RECONFIG instance to the same clock that is connected to the `reconfig_clk` input port of the ALTGX instance.

Table 5–3 shows the range of frequency values of the `reconfig_clk` input port for the **Receiver only**, **Recevier and Transmitter**, and **Transmitter only** configuration modes of the ALTGX instance. Table 5–3 shows the clock requirements for `reconfig_clk` input port to the ALTGX_RECONFIG instance based on the ALTGX configurations.

Based on the ALTGX configurations (**Recevier only**, **Transmitter only**, **Recevier and Transmitter** configurations) controlled by the ALTGX_RECONFIG instance, select the fastest `reconfig_clk` frequency value. This satisfies both the offset cancellation control for receiver channels and the dynamic reconfiguration of the transmitter and receiver channels.

Table 5–3. Reconfig_clk Settings for ALTGX_RECONFIG Instance *(Note 1)*

ALTGX Instance Configuration	reconfig_clk Frequency Range
Receiver and Transmitter reconfiguration mode	37.5 MHz to 50 MHz
Receiver only reconfiguration mode	37.5 MHz to 50 MHz
Transmitter only reconfiguration mode	2.5 MHz to 50 MHz

Note to Table 5–3:

(1) Altera recommends the `reconfig_clk` signal be driven on a global clock resource.

Interfacing ALTGX_RECONFIG Instance and ALTGX Instances

This section describes the various dynamic reconfiguration settings available in the ALTGX_RECONFIG and ALTGX MegaWizard Plug-In Managers, and how to set them. It also provides information about the interface signals and connections between the ALTGX_RECONFIG and ALTGX instances.

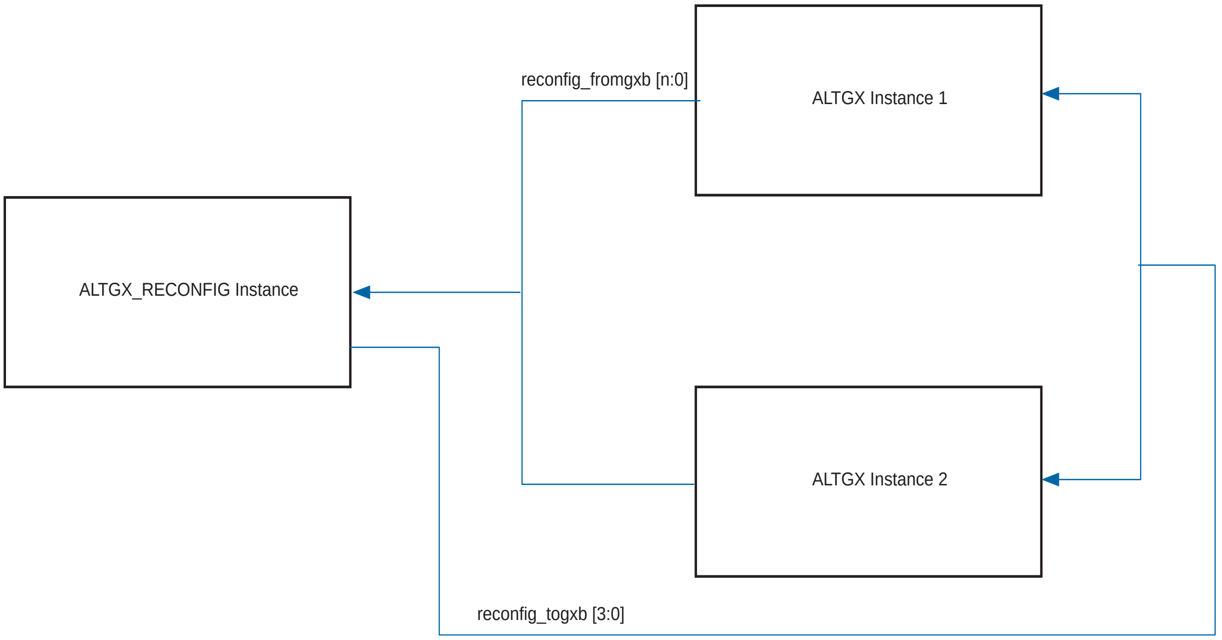
There are two ways to connect the ALTGX_RECONFIG instance to the ALTGX instance in your design:

- Single Dynamic Reconfiguration controller

  A single ALTGX_RECONFIG instance can be used to control all the ALTGX instances in your design.

  Figure 5–5 shows a block diagram of a single ALTGX_RECONFIG instance controlling multiple ALTGX instances.

Figure 5–5. Block Diagram of a Single Dynamic Reconfiguration Controller in a Design

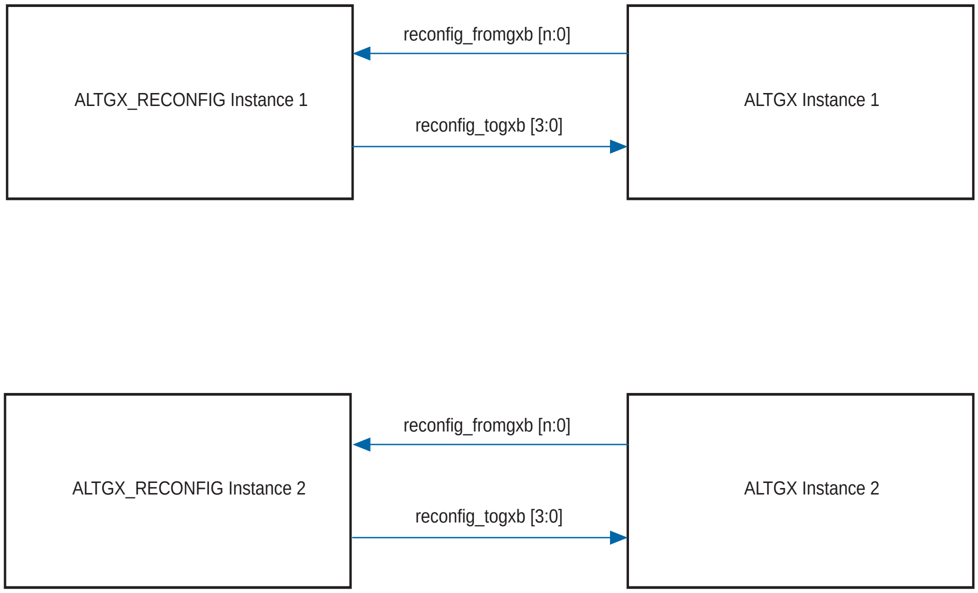


- Multiple Dynamic Reconfiguration controllers

  Your design can have multiple ALTGX_RECONFIG instances, where every ALTGX instance is controlled by it own ALTGX_RECONFIG instance.

  Figure 5–6 shows a block diagram of multiple ALTGX_RECONFIG instances controlling an ALTGX instance each.

Figure 5–6. Block Diagram of Multiple Dynamic Reconfiguration Controllers in a Design



To enable dynamic reconfiguration of a transceiver channel, it is important to understand the following:

- Logical Channel Addressing

  The dynamic reconfiguration controller identifies a transceiver channel by using the logical channel address. The **What is the starting channel number?** option in the **Reconfig screen** of the ALTGX MegaWizard Plug-In Manager enables the user to set the logical channel address of all the channels within the ALTGX instance. This concept is explained in detail in "Logical Channel Addressing".

- Total number of channels controlled by the ALTGX_RECONFIG instance

  Every dynamic reconfiguration controller in a design might be connected to either a single ALTGX instance or multiple ALTGX instances. Depending on the number of channels within each of these ALTGX instances, you must set the total number of channels controlled by the dynamic reconfiguration controller in the ALTGX_RECONFIG MegaWizard Plug-In Manager. This concept is explained in "Total Number of Channels Controlled by the ALTGX_RECONFIG Instance" on page 5–25 .

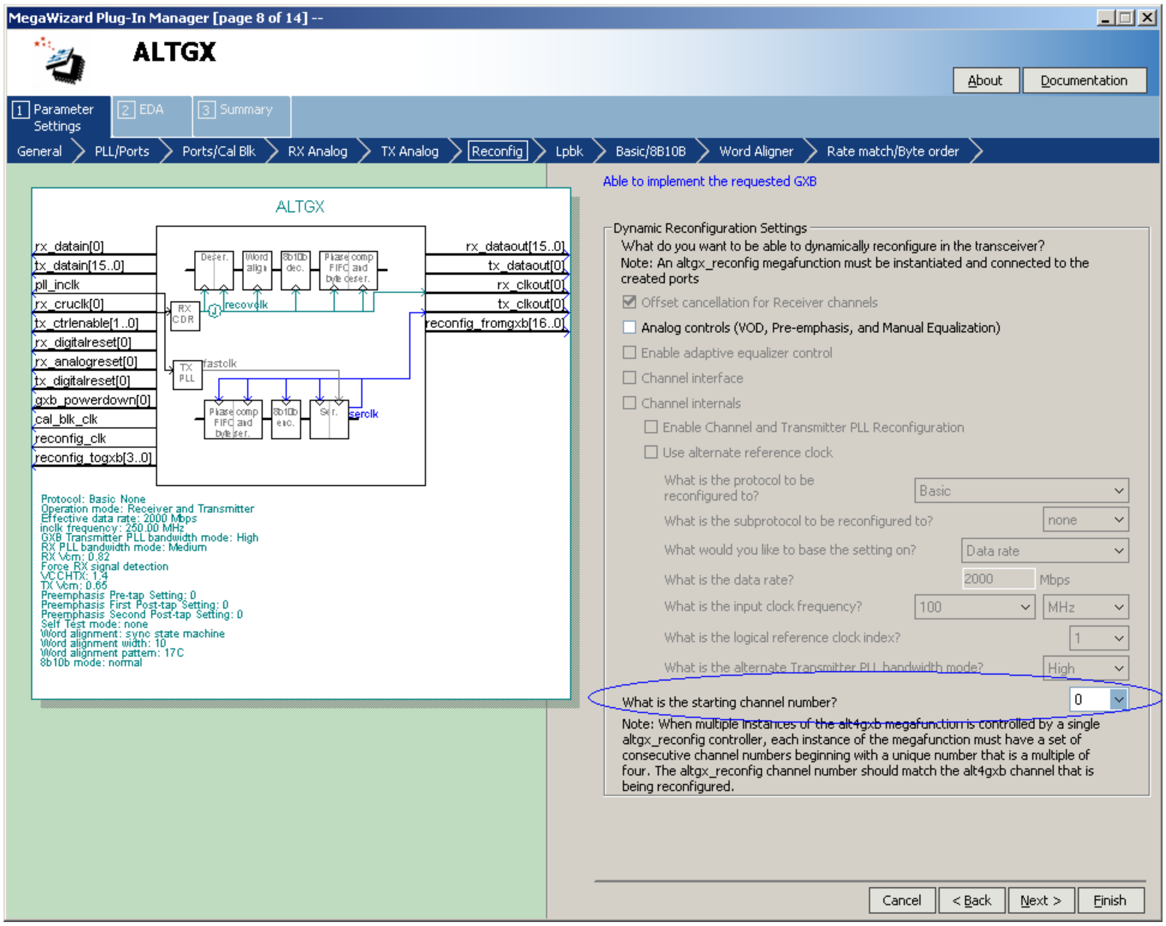
- Connecting `reconfig_fromgxb/reconfig_togxb` ports between the ALTGX and ALTGX_RECONFIG instances.

Logical Channel Addressing

This section describes how to set the **What is the starting channel number?** option using five different case scenarios.

Figure 5–7 shows the **What is the starting channel number?** option in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager.

Figure 5–7. The 'What is the starting channel number?' Option in the ALTGX MegaWizard Plug-In Manager



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Dynamic Reconfiguration Settings

What do you want to be able to dynamically reconfigure in the transceiver?
Note: An altgx_reconfig megafunction must be instantiated and connected to the created ports

☑ Offset cancellation for Receiver channels
☐ Analog controls (VOD, Pre-emphasis, and Manual Equalization)
☐ Enable adaptive equalizer control
☐ Channel interface
☐ Channel internals
☐ Enable Channel and Transmitter PLL Reconfiguration
☐ Use alternate reference clock

What is the protocol to be reconfigured to? Basic
What is the subprotocol to be reconfigured to? none
What would you like to base the setting on? Data rate
What is the data rate? 2000 Mbps
What is the input clock frequency? 100 MHz
What is the logical reference clock index? 1
What is the alternate Transmitter PLL bandwidth mode? High
What is the starting channel number? 0

Note: When multiple instances of the alt4gxb megafunction is controlled by a single altgx_reconfig controller, each instance of the megafunction must have a set of consecutive channel numbers beginning with a unique number that is a multiple of four. The altgx_reconfig channel number should match the alt4gxb channel that is being reconfigured.

This value determines the logical channel address of all the transceiver channels in the ALTGX instance. You must always set the starting channel number in an ALTGX instance as a multiple of 4.

Table 5–4 shows the example scenarios under which you must set the starting channel number differently.

Table 5–4. Example Scenarios for Logical Channel Addressing in ALTGX Instances (Part 1 of 2)

Example Scenario	Number of ALTGX Instances	Number of ALTGX_RECONFIG Instances
Case 1	Two ALTGX instances: ■ ALTGX_instance 1 ■ ALTGX_instance 2	One ALTGX_RECONFIG instance controlling both the ALTGX instances
Case 2	Two ALTGX instances: ■ ALTGX_instance 1 ■ ALTGX_instance 2	One ALTGX_RECONFIG instance controlling both the ALTGX instances

Table 5–4. Example Scenarios for Logical Channel Addressing in ALTGX Instances (Part 2 of 2)

Example Scenario	Number of ALTGX Instances	Number of ALTGX_RECONFIG Instances
Case 3	Two ALTGX instances: ■ ALTGX_instance 1 ■ ALTGX_instance 2	One ALTGX_RECONFIG instance controlling both the ALTGX instances
Case 4	Two ALTGX instances: ■ ALTGX_instance 1 ■ ALTGX_instance 2	Two ALTGX_RECONFIG instances: ALTGX_RECONFIG instance 1 and ALTGX_RECONFIG instance 2 ALTGX_RECONFIG instance 1 controlling the ALTGX instance 1 ALTGX_RECONFIG instance 2 controlling the ALTGX instance 2
Case 5	One ALTGX instance (ALTGX instance 1) stamped five times.	One ALTGX_RECONFIG instance controlling all the five stamped ALTGX.v or ALTGX.vhd instances.

Case 1

Consider that you have two ALTGX instances connected to one dynamic reconfiguration controller (ALTGX_RECONFIG Instance). The following are the configurations of the two ALTGX instances:

- ALTGX instance 1

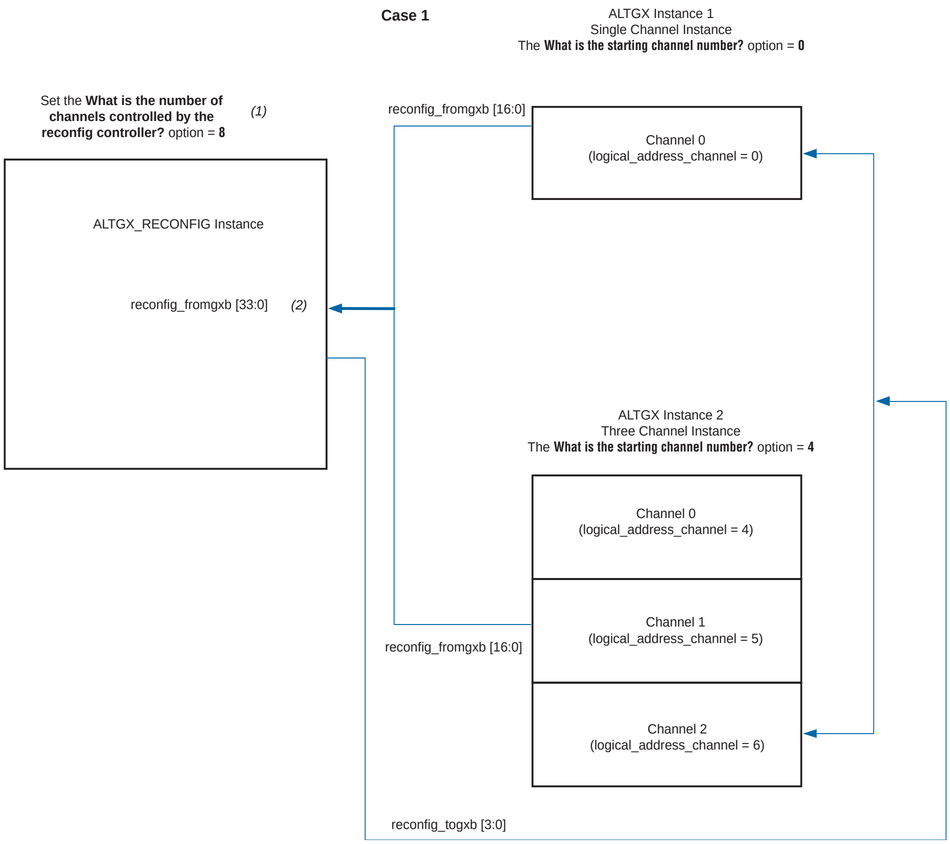
  The number of channels in this instance is 1

- ALTGX instance 2

  The number of channels in this instance is 3

Figure 5–8 shows a block diagram of the ALTGX instances and ALTGX_RECONFIG instance in the design.

Figure 5–8. Case 1 - Block Diagram of the ALTGX Instances and ALTGX_RECONFIG Instance


Case 1
ALTGX Instance 1
Single Channel Instance
The What is the starting channel number? option = 0
Set the What is the number of channels controlled by the reconfig controller? option = 8
(1)
reconfig_fromgxb [16:0]
Channel 0
(logical_address_channel = 0)
ALTGX_RECONFIG Instance
reconfig_fromgxb [33:0]
(2)
ALTGX Instance 2
Three Channel Instance
The What is the starting channel number? option = 4
Channel 0
(logical_address_channel = 4)
Channel 1
(logical_address_channel = 5)
reconfig_fromgxb [16:0]
Channel 2
(logical_address_channel = 6)
reconfig_togxb [3:0]

Note to Figure 5–8:

(1) This option is discussed more in detail in “Case 1” on page 5–26.

(2) `reconfig_fromgxb[33:0]` ={`reconfig_fromgxb[16:0]`, `reconfig_fromgxb[16:0]`}

ALTGX Instance 1 Parameter Settings

Set the **What is the starting channel number?** option as **0.** This implies that the ATLGX MegaWizard Plug-In Manager sets the logical channel address of the first channel of ALTGX instance 1 = **0**.

Figure 5–8 shows the logical channel addresses assigned to the channels in the ALTGX instance 1 by the ATLGX MegaWizard Plug-In Manager.

ALTGX Instance 2 Parameter Settings

Because the starting channel number increments in steps of 4, you must set the **What is the starting channel number?** option of the next ALTGX instance as a multiple of 4**.** Therefore, set the **What is the starting channel number?** option as **4** for ALTGX instance 2.

This implies that the ATLGX MegaWizard Plug-In Manager sets the logical channel address of the first channel of ALTGX instance 2 = 4. The ATLGX MegaWizard Plug-In Manager assigns consecutive numbers as the logical channel address for all the channels within an ALTGX instance. The logical channel address of the second channel of ALTGX instance 2 = **5**. The logical channel address of the third channel of ALTGX instance 2 = **6**.

You must set the next multiple of 4 for the **What is the starting channel number?** option of the following ALTGX instances, if any.

Figure 5–8 shows the logical channel addresses of all the channels in the ALTGX instance 2.

Case 2

Consider that you have two ALTGX instances and one ALTGX_RECONFIG instance in the design:

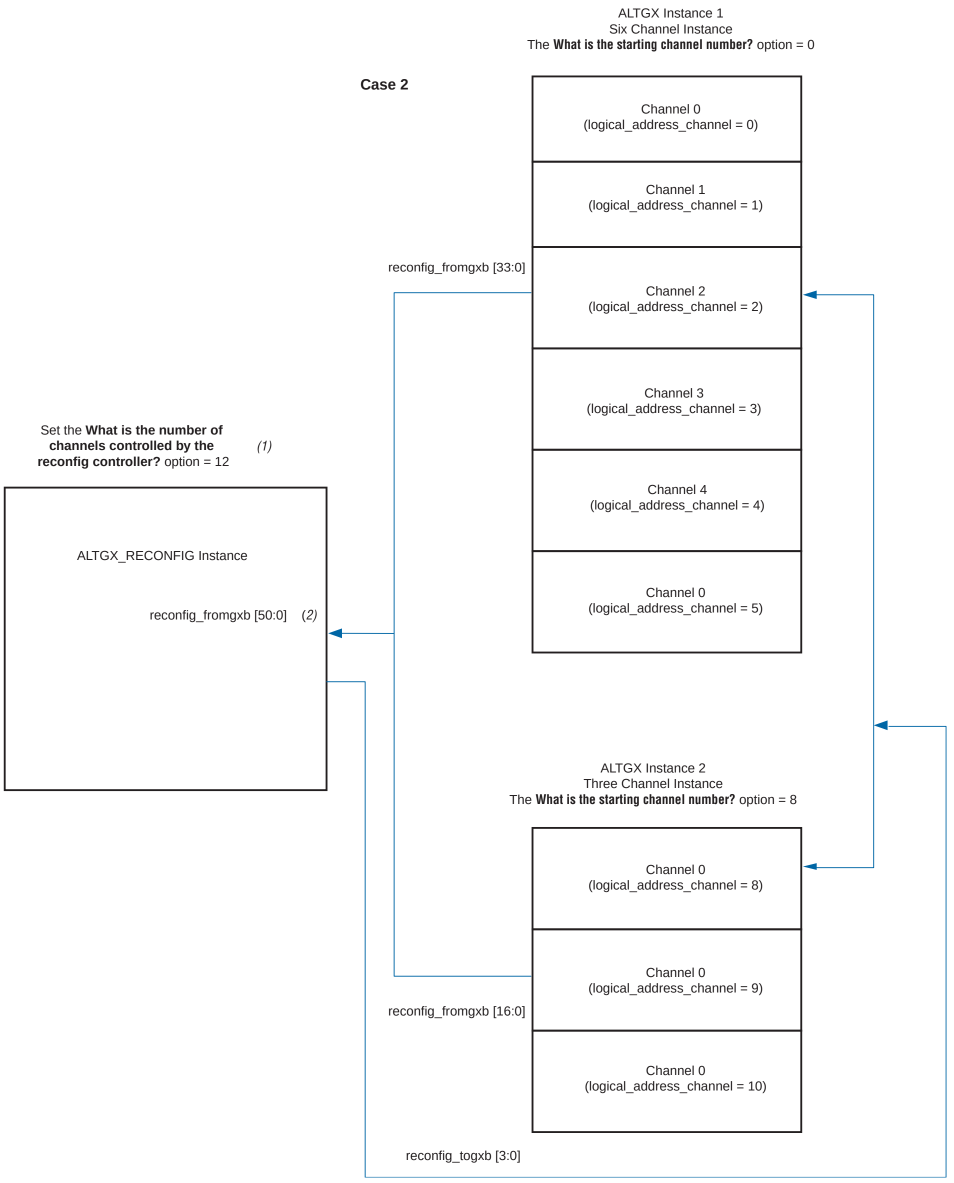
- ALTGX instance 1

  The number of channels in this instance is 6
- ALTGX instance 2

  The number of channels in this instance is 3

Figure 5–9 shows a block diagram of the ALTGX instances and ALTGX_RECONFIG instance in the design.

Figure 5–9. Case 2 - Block Diagram of the ALTGX Instances and ALTGX_RECONFIG Instance


ALTGX Instance 1
Six Channel Instance
The **What is the starting channel number?** option = 0
Case 2
Channel 0
(logical_address_channel = 0)
Channel 1
(logical_address_channel = 1)
reconfig_fromgxb [33:0]
Channel 2
(logical_address_channel = 2)
Channel 3
(logical_address_channel = 3)
Set the **What is the number of channels controlled by the reconfig controller?** option = 12
(1)
Channel 4
(logical_address_channel = 4)
ALTGX_RECONFIG Instance
Channel 0
(logical_address_channel = 5)
reconfig_fromgxb [50:0]
(2)
ALTGX Instance 2
Three Channel Instance
The **What is the starting channel number?** option = 8
Channel 0
(logical_address_channel = 8)
Channel 0
(logical_address_channel = 9)
reconfig_fromgxb [16:0]
Channel 0
(logical_address_channel = 10)
reconfig_togxb [3:0]

Note to Figure 5–9:

(1) This option is discussed in more detail in "Case 2" on page 5–26.

(2) `reconfig_fromgxb [50:0]` = {`reconfig_fromgxb [33:0]`, `reconfig_fromgxb [16:0]`}

ALTGX Instance 1 Parameter Settings

Set the **What is the starting channel number?** option as **0.** This implies that the ATLGX MegaWizard Plug-In Manager sets the logical channel address of the first channel of ALTGX instance 1 = **0.** The ATLGX MegaWizard Plug-In Manager assigns consecutive numbers as the logical channel address for all the channels within an ALTGX instance. The logical channel addresses of the second channel to sixth channel of the ALTGX instance 1 are **1** to **5**, respectively. of ALTGX instance 1 = **1**.

Figure 5–9 shows the logical channel addresses of all the channels in the ALTGX instance 1.

ALTGX Instance 2 Parameter Settings

Because the **What is the starting channel number?** option increments in steps of 4, you must set the **What is the starting channel number?** option of the next ALTGX instance as a multiple of 4**.** Therefore, set the **What is the starting channel number?** option as **8** for ALTGX instance 2.

This implies that the ATLGX MegaWizard Plug-In Manager sets the logical channel address of the first channel of ALTGX instance 2 = **8**. The logical channel addresses of the second channel to third channel are **9** to **10**, respectively. You must set the next multiple of 4 in the **What is the starting channel number?** option of the following ALTGX instances, if any.

Figure 5–9 shows the logical channel addresses of all the channels in the ALTGX instance 2.

☞ For Case 2, do not set the **What is the starting channel number?** option as **4** for ALTGX instance 2 as in Case 1 because the logical channel addresses 4 and 5 are used already for the fifth and sixth channels of ALTGX instance 1. Instead, use the next multiple of 4 to assign it as the starting channel number of ALTGX instance 2.

Case 3

Consider that you have two ALTGX instances and one ALTGX_RECONFIG instance in the design:

- ALTGX instance 1

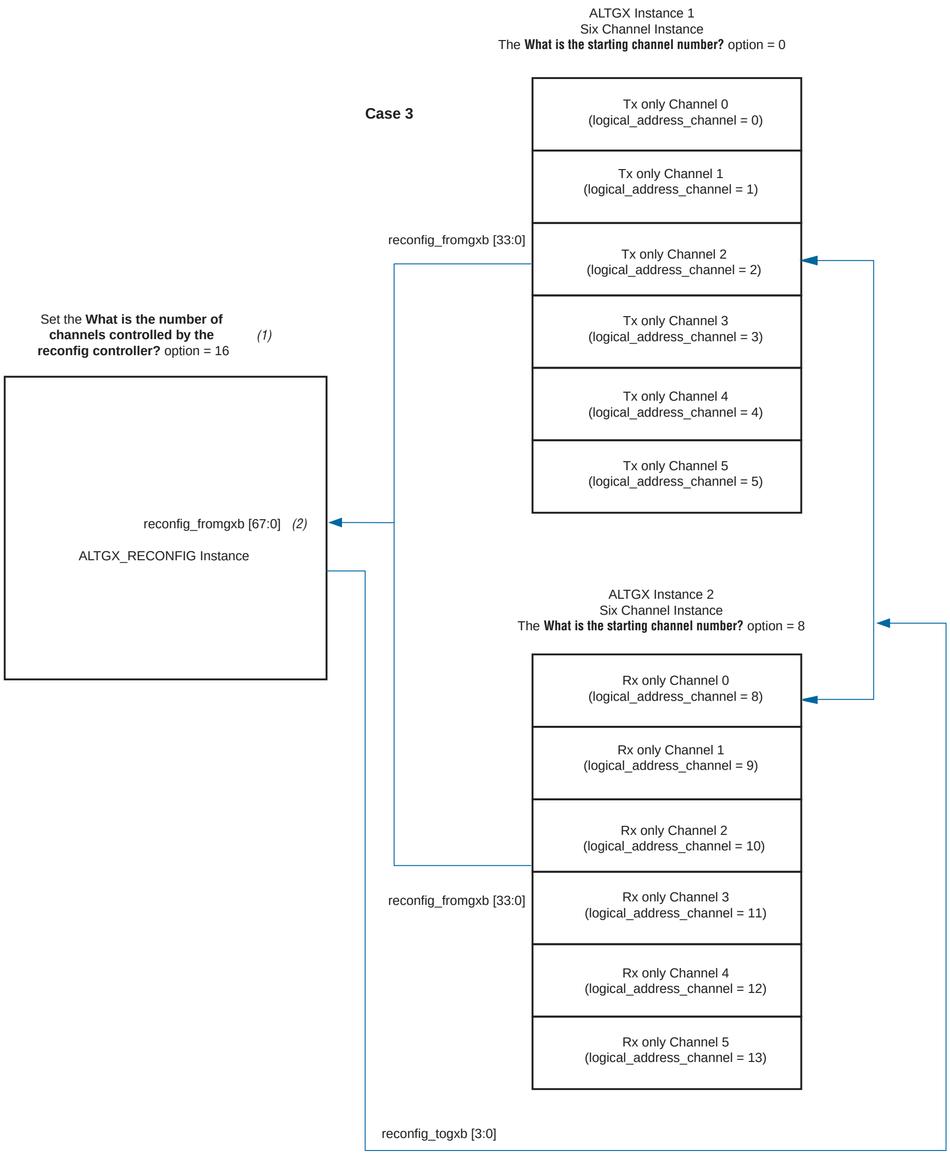
  This is a **Transmitter only** configuration and the number of channels is 6.

- ALTGX instance 2

  This is a **Receiver only** configuration and the number of channels is 6.

Figure 5–10 shows a block diagram of the ALTGX instances and ALTGX_RECONFIG instance in the design.

Figure 5–10. Case 3 - Block Diagram of the ALTGX Instances and ALTGX_RECONFIG Instance


ALTGX Instance 1
Six Channel Instance
The **What is the starting channel number?** option = 0
Case 3
Tx only Channel 0 (logical_address_channel = 0)
Tx only Channel 1 (logical_address_channel = 1)
reconfig_fromgxb [33:0]
Tx only Channel 2 (logical_address_channel = 2)
Tx only Channel 3 (logical_address_channel = 3)
Tx only Channel 4 (logical_address_channel = 4)
Tx only Channel 5 (logical_address_channel = 5)
Set the **What is the number of channels controlled by the reconfig controller?** option = 16 (1)
reconfig_fromgxb [67:0] (2)
ALTGX_RECONFIG Instance
ALTGX Instance 2
Six Channel Instance
The **What is the starting channel number?** option = 8
Rx only Channel 0 (logical_address_channel = 8)
Rx only Channel 1 (logical_address_channel = 9)
Rx only Channel 2 (logical_address_channel = 10)
reconfig_fromgxb [33:0]
Rx only Channel 3 (logical_address_channel = 11)
Rx only Channel 4 (logical_address_channel = 12)
Rx only Channel 5 (logical_address_channel = 13)
reconfig_togxb [3:0]

Note to Figure 5–10:

(1) This option is discussed in more detail in "Case 3" on page 5–27.

(2) `reconfig_fromgxb [67:0]` = {`reconfig_fromgxb [33:0]`, `reconfig_fromgxb [33:0]`}

ALTGX Instance 1 Parameter Settings

Set the **What is the starting channel number?** option as **0.** This implies that the ATLGX MegaWizard Plug-In Manager sets the logical channel address of the first channel of ALTGX instance 1 = **0.** The ATLGX MegaWizard Plug-In Manager assigns consecutive numbers as the logical channel address for all the channels within an ALTGX instance. The logical channel addresses of the second channel to sixth channel are **1** to **5**, respectively.

Figure 5–10 shows the logical channel addresses of all the channels in the ALTGX instance 1.

ALTGX Instance 2 Parameter Settings

Because the **What is the starting channel number?** option increments in steps of 4, you must set the **What is the starting channel number?** option of the next ALTGX instance as a multiple of 4**.** Therefore, set the starting channel number as **8** for ALTGX instance 2.

This implies that the ATLGX MegaWizard Plug-In Manager sets the logical channel address of the first channel of ALTGX instance 2 = **8.** The ATLGX MegaWizard Plug-In Manager assigns consecutive numbers as the logical channel address for all the channels within an ALTGX instance. The logical channel addresses of the second channel to sixth channel are **9** to **13**, respectively. You must set the next multiple of 4 in the **What is the starting channel number?** option of the following ALTGX instances, if any.

Figure 5–10 shows the logical channel addresses of all the channels in the ALTGX instance 2.

Consider the scenario where the design requires 32 **Transmitter only** configurations and 32 **Receiver only** configurations. The maximum possible ALTGX instances are 64 in this case (32 **Transmitter only** ALTGX instances and 32 **Receiver only** ALTGX instances).

The first **Transmitter only** instance has the **What is the starting channel number?** option set to **0**. The second **Transmitter only** ALTGX instance has the **What is the starting channel number?** option set to **4** and so on. The 32nd **Transmitter only** ALTGX instance has the **What is the starting channel number?** option set to **124**.

Extending the same logic to the 32 **Receiver only** ALTGX instances, the 32nd **Receiver only** ALTGX instance has the **What is the starting channel number?** option set to **252** (A total of 64 instances × 4). Therefore the maximum possible logical channel addresses that can be assigned to each of the channels in all the ALTGX instances is 256.

The Quartus II software automatically packs the logical channels into the physical placements. The physical placement includes combining channels into the same transceiver block. Refer to "Combining Transceiver Channels with Dynamic Reconfiguration Enabled" on page 5–56 for additional details.

Case 4

Similarly, consider the other scenario where there are multiple ALTGX_RECONFIG instances individually controlling the ALTGX instances in your design.

Case 4 explains the scenario under which you must set the **What is the starting channel number?** option differently. This scenario assumes that each ALTGX instance has its own ALTGX_RECONFIG instance, such as:

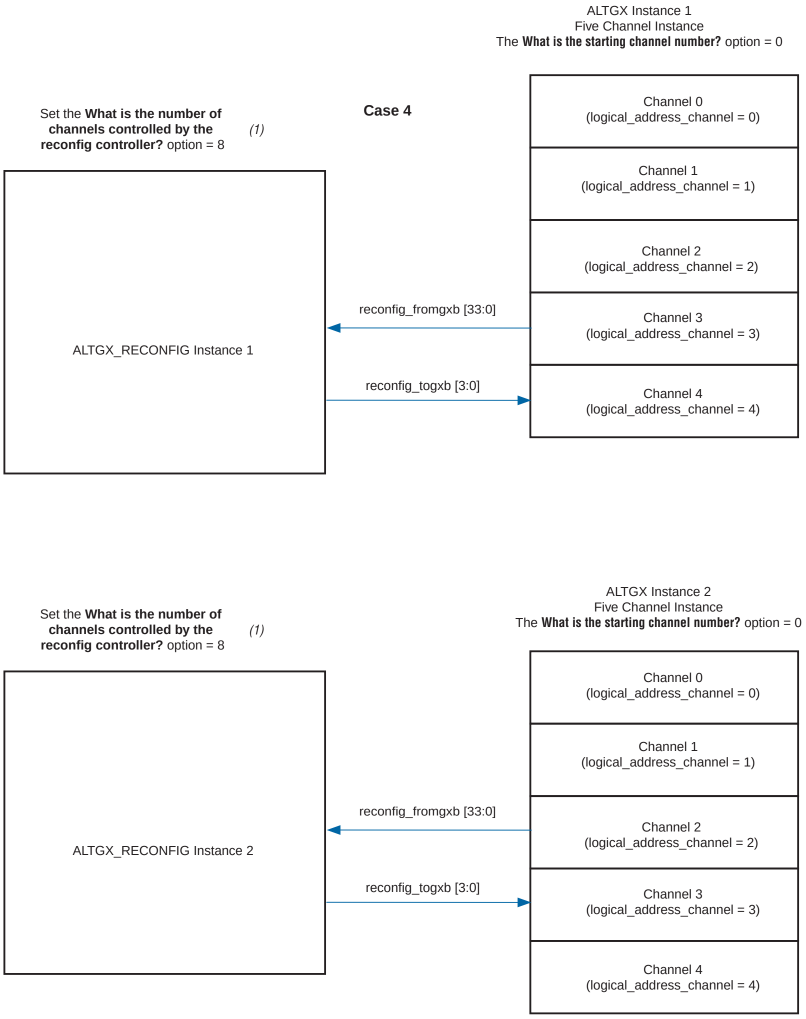
- If you have two ALTGX instances individually controlled by two ALTGX_RECONFIG instances in the design.
- ALTGX instance 1

  The number of channels is 5. This instance is controlled by ALTGX_RECONFIG instance 1.
- ALTGX instance 2

  The number of channels is 5. This instance is controlled by ALTGX_RECONFIG instance 2.

Figure 5–11 shows a block diagram of the ALTGX instances and ALTGX_RECONFIG instances in the design.

Figure 5–11. Case 4 - Block Diagram of the ALTGX Instances and ALTGX_RECONFIG Instances


ALTGX Instance 1
Five Channel Instance
The **What is the starting channel number?** option = 0
Case 4
Set the **What is the number of channels controlled by the reconfig controller?** option = 8
(1)
Channel 0 (logical_address_channel = 0)
Channel 1 (logical_address_channel = 1)
Channel 2 (logical_address_channel = 2)
Channel 3 (logical_address_channel = 3)
Channel 4 (logical_address_channel = 4)
reconfig_fromgxb [33:0]
reconfig_togxb [3:0]
ALTGX_RECONFIG Instance 1
ALTGX Instance 2
Five Channel Instance
The **What is the starting channel number?** option = 0
Set the **What is the number of channels controlled by the reconfig controller?** option = 8
(1)
Channel 0 (logical_address_channel = 0)
Channel 1 (logical_address_channel = 1)
Channel 2 (logical_address_channel = 2)
Channel 3 (logical_address_channel = 3)
Channel 4 (logical_address_channel = 4)
reconfig_fromgxb [33:0]
reconfig_togxb [3:0]
ALTGX_RECONFIG Instance 2

Note to Figure 5–11:

(1) This option is discussed in more detail in "Case 4" on page 5–27.

ALTGX Instance 1 Parameter Settings

Set the **What is the starting channel number?** option as **0.** This implies that the ATLGX MegaWizard Plug-In Manager sets the logical channel address of the first channel of ALTGX instance 1 = **0.** The ATLGX MegaWizard Plug-In Manager assigns consecutive numbers as the logical channel address for all the channels within an ALTGX instance.

The logical channel addresses of the second channel to fifth channel are **1** to **4**, respectively. You must set the next multiple of 4 as the starting channel number for the following ALTGX instances controlled by ALTGX_RECONFIG instance 1, if any.

Figure 5–11 shows the logical channel addresses of all the channels in the ALTGX instance 1.

ALTGX Instance 2 Parameter Settings

The ALTGX instance 2 is controlled by a second dynamic reconfiguration controller. Therefore, set the **What is the starting channel number?** option as **0** for ALTGX instance 2. This implies that the ATLGX MegaWizard Plug-In Manager sets the logical channel address of the first channel of ALTGX instance 2 = **0.** The ATLGX MegaWizard Plug-In Manager assigns consecutive numbers as the logical channel address for all the channels within an ALTGX instance. The logical channel addresses of the second channel to fifth channel are **1** to **4**, respectively. You must set the next multiple of 4 in the **What is the starting channel number?** option of the following ALTGX instances controlled by ALTGX_RECONFIG instance 1, if any.

Figure 5–11 shows the logical channel addresses of all the channels in the ALTGX instance 2.

Case 5

Consider that you have only one ALTGX instance (ALTGX instance 1) and one ALTGX_RECONFIG instance in the design:

- ALTGX instance 1

  The number of channels in this instance is 1.

- **ALTGX .v** or **ALTGX.vhd** is stamped five times in the design.

ALTGX Instance 1 Parameter Settings

Set the **What is the starting channel number?** option as **0.** This implies that the ATLGX MegaWizard Plug-In Manager sets the logical channel address of the single channel of ALTGX instance 1 = **0.**

When you stamp the above configured transceiver instance five times, the starting channel numbers of the other four instances (assume 'instance2', instance3', instance4, instance5, instance6) are **4**, **8**, **12**, and **16**, respectively.

Specify the starting channel number of the other stamped instances using the `defparam` parameter (for verilog) as shown:

```
defparam instance2. starting_channel_number = 4;
```

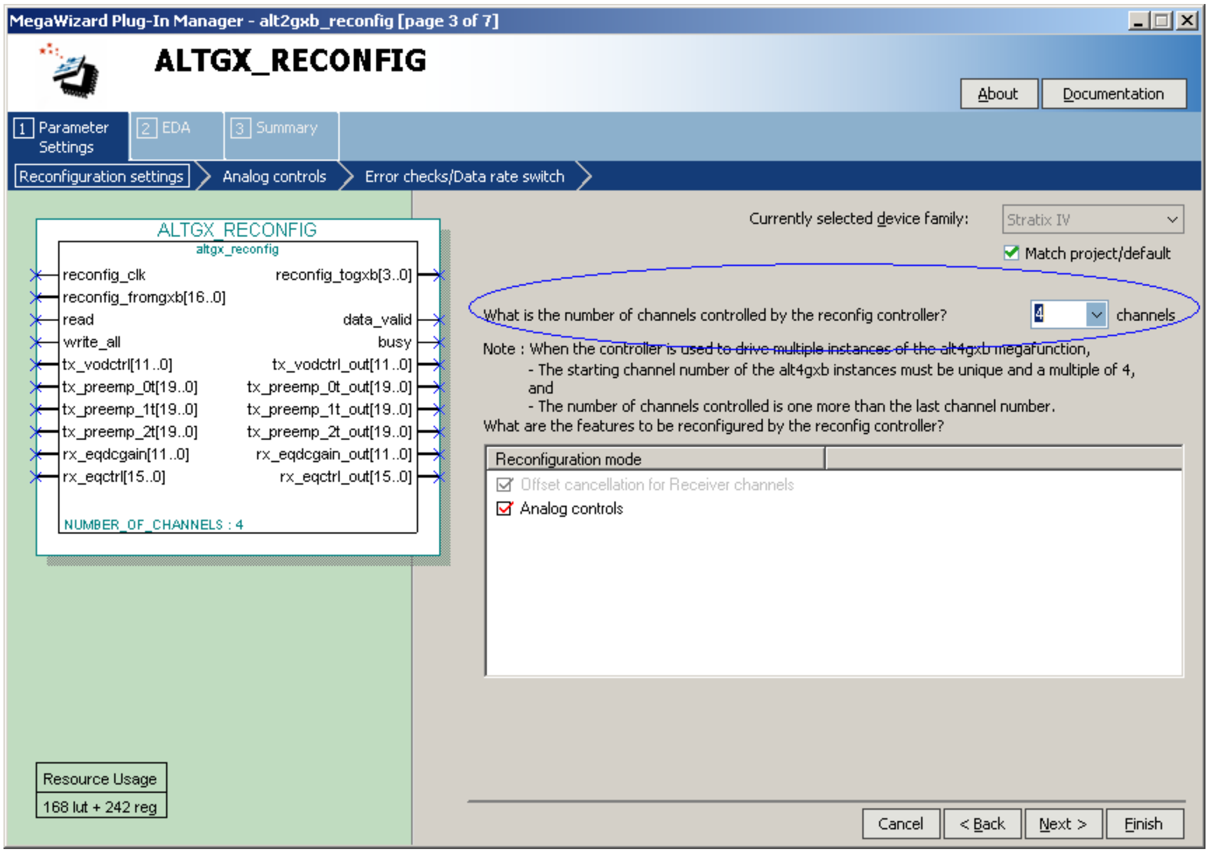
`defparam instance3. starting_channel_number = 8;` and so on for the remaining stamped instances.

Total Number of Channels Controlled by the ALTGX_RECONFIG Instance

The dynamic reconfiguration controller requires information about the total number of channels connected to it. Based on this information, the `reconfig_fromgxb`, `logical_channel_address` input ports vary in width. Therefore, provide this information in the **What is the number of channels controlled by the reconfig controller?** option in the **Reconfiguration settings** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager, as shown in Figure 5–12.

The following section describes how to set the total number of channels controlled by the dynamic reconfiguration controller (ALTGX_RECONFIG instance). The maximum number of channels that you can set in this option is 256.

Figure 5–12. 'What is the number of channels controlled by the reconfig controller?' Option in the ALTGX_RECONFIG MegaWizard Plug-In Manager



MegaWizard Plug-In Manager - alt2gxb_reconfig [page 3 of 7]

ALTGX_RECONFIG

Currently selected device family: Stratix IV

☑ Match project/default

What is the number of channels controlled by the reconfig controller? 4 channels

Note : When the controller is used to drive multiple instances of the alt4gxb megafunction,
- The starting channel number of the alt4gxb instances must be unique and a multiple of 4, and
- The number of channels controlled is one more than the last channel number.

What are the features to be reconfigured by the reconfig controller?

Reconfiguration mode

☑ Offset cancellation for Receiver channels

☑ Analog controls

Resource Usage
168 lut + 242 reg

Consider the scenario where one ALTGX_RECONFIG instance is controlling all the ALTGX instances in a design. Follow the rules listed below for setting the **What is the number of channels controlled by the controller?** option:

- Determine the highest logical channel address amongst all the transceiver instances connected to the same dynamic reconfiguration controller. Refer to "Logical Channel Addressing" on page 5–14 for information on determining the logical channel address using the starting channel number.
- Round the logical channel address value to the nearest multiple of 4.
- Use this value to set the **What is the number of channels controlled by the reconfig controller**? option.
- Consider the example scenarios discussed in Table 5–4 on page 5–15, and set the total number of channels in the ALTGX_RECONFIG MegaWizard Plug-In Manager, for the same example scenarios.

Case 1

Consider that you have two ALTGX instances connected to one dynamic reconfiguration controller (ALTGX_RECONFIG Instance): The following are the configurations of the two ALTGX instances:

- ALTGX instance 1

  The number of channels in this instance is 1
- ALTGX instance 2

  The number of channels in this instance is 3

Set the **What is the starting channel number?** option for both the ALTGX instances. This is discussed in detail in "Case 1" on page 5–16. Set the **What is the starting channel number?** option to **0** for ALTGX instance 1 and **4** for the ALTGX instance 2.

Determine the highest logical channel address amongst all the transceiver instances connected to the same dynamic reconfiguration controller. The highest logical channel address in this case is **6**. Round the logical channel address 6 to the nearest multiple of 4 = **8**. Even though the total number of channels in the design is 4, the total number of channels controlled is **8**. Therefore for this scenario, set the **What is the number of channels controlled by the controller**? option in the **Reconfiguration settings** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager to **8**.

Refer to Figure 5–8 on page 5–17 which shows a block diagram of the ALTGX instances and ALTGX_RECONFIG instance in this design.

Case 2

Consider that you have two ALTGX instances and one ALTGX_RECONFIG instance in the design.

- ALTGX instance 1

  The number of channels in this instance is 6
- ALTGX instance 2

  The number of channels in this instance is 3

Set the **What is the starting channel number?** option for both the ALTGX instances. This is discussed in detail in "Case 2" on page 5–18. Set the **What is the starting channel number?** option to **0** for ALTGX instance 1 and **8** for the ALTGX instance 2.

Determine the highest logical channel address amongst all the transceiver instances connected to the same dynamic reconfiguration controller. The highest logical channel address in this case is **10**. Round the logical channel address 10 to the nearest multiple of 4 = **12**. Even though the total number of channels in the design is 9, the total number of channels controlled is **12**. Therefore for this scenario, set the **What is the number of channels controlled by the controller**? option in the **Reconfiguration settings** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager to **12**.

Refer to Figure 5–9 which shows a block diagram of the ALTGX instances and ALTGX_RECONFIG instance in this design.

Case 3

Consider that you have two ALTGX instances and one ALTGX_RECONFIG instance in the design:

- ALTGX instance 1: This is a **Transmitter only** configuration and the number of channels is 6.
- ALTGX instance 2: This is a **Receiver only** configuration and the number of channels is 6.

Set the **What is the starting channel number?** option for both the ALTGX instances. This is discussed in detail in "Case 3" on page 5–20. Set the **What is the starting channel number?** option to **0** for ALTGX instance 1 and **8** for the ALTGX instance 2.

Determine the highest logical channel address amongst all the transceiver instances connected to the same dynamic reconfiguration controller. The highest logical channel address in this case is **13**. Round the logical channel address 13 to the nearest multiple of 4 = **16**. Even though the total number of channels in the design is 12, the total number of channels controlled is **16**. Therefore for this scenario, set the **What is the number of channels controlled by the controller**? option in the **Reconfiguration settings** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager to **16**.

Refer to Figure 5–10 which shows a block diagram of the ALTGX instances and ALTGX_RECONFIG instance in this design.

Case 4

Case 4 explains the scenario where you must set the total number of channels in the ALTGX_RECONFIG MegaWizard Plug-In Manager differently. This scenario assumes that each ALTGX instance has its own ALTGX_RECONFIG instance.

Consider that you have two ALTGX instances individually controlled by two ALTGX_RECONFIG instances in the design.

- ALTGX instance 1

  The number of channels is 5. This instance is controlled by ALTGX_RECONFIG instance 1.

- ALTGX instance 2

  The number of channels is 5. This instance is controlled by ALTGX_RECONFIG instance 2.

Set the **What is the starting channel number?** option for both the ALTGX instances. This is discussed in detail in "Case 4" on page 5–22. Set the **What is the starting channel number?** option to **0** for ALTGX instance 1 and **0** for the ALTGX instance 2.

Determine the highest logical channel address amongst all the transceiver instances connected to the same dynamic reconfiguration controller. The highest logical channel address from ALTGX instance 1 connected to ALTGX_RECONFIG instance 1 is **4**. Similarly, the highest logical channel address from ALTGX instance 2 connected to ALTGX_RECONFIG instance 2 is **4**. Round the highest logical channel address connected to ALTGX_RECONFIG instance 1, 4 to the nearest multiple of 4 = **8**. Similarly, round the highest logical channel address connected to ALTGX_RECONFIG instance 2, 4 to the nearest multiple of 4 = **8**. Even though the total number of channels in the design is 10, the total number of channels controlled by each of the ALTGX_RECONFIG instances is **8**. Therefore for this scenario, set the **What is the number of channels controlled by the controller**? option in the **Reconfiguration settings** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager to **8**, for both the dynamic reconfiguration controllers.

Refer to Figure 5–11 which shows a block diagram of the ALTGX instances and ALTGX_RECONFIG instances in this design.

Case 5

This example scenario has just one ALTGX instance stamped five times. One ALTGX_RECONFIG instance controls all the five stamped instances.

Set the **What is the starting channel number?** option for all the stamped ALTGX instances. This is discussed in detail in "Case 5" on page 5–24.

Determine the highest logical channel address amongst all the transceiver instances connected to the same dynamic reconfiguration controller. The highest logical channel address in this case is **16**. Round the logical channel address 16 to the nearest multiple of 4 = **20**. Even though the total number of channels in the design is 5, the total number of channels controlled is **20**. Therefore for this scenario, set the "**What is the number of channels controlled by the controller**?" option in the **Reconfiguration settings** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager to **20**.

Connecting reconfig_from_gxb/reconfig_to_gxb Ports

The dynamic reconfiguration interface has the `reconfig_fromgxb` and `reconfig_togxb` signals which must be connected between the ALTGX_RECONFIG instance and the ALTGX instance to successfully complete the dynamic reconfiguration process:

- `reconfig_togxb[3:0]`

  This is an input port of the ALTGX instance and an output port of the ALTGX_RECONFIG instance. You must connect the `reconfig_togxb[3:0]` input port of every ALTGX instance controlled by the dynamic reconfiguration controller, to the `reconfig_togxb[3:0]` output port of the ALTGX_RECONFIG instance. Refer to Figure 5–13 for additional information.

- `reconfig_fromgxb`

  This is an output port in the ALTGX instance and an input port in the ALTGX_RECONFIG instance. This is a transceiver block based signal. Therefore the width of this signal increases in steps of 17 bits per transceiver block.

In the ALTGX MegaWizard Plug-In Manager, the width of this signal depends on the number of channels you select in the **What is the number of channels?** option in the **General** screen.

For example, if you select the number of channels in the **ALTGX instance** as follows:

- 1 ≤Channels ≤4 then output port `reconfig_fromgxb` = 17 bits
- 5 ≤Channels ≤8 then output port `reconfig_fromgxb` = 34 bits
- 9 ≤Channels ≤12 then output port `reconfig_fromgxb` = 51 bits

In the ALTGX_RECONFIG MegaWizard Plug-In Manager, the width of this signal depends on the value you select in the **What is the number of channels controlled by the reconfig controller?** option in the **Reconfiguration settings** screen.

For example, if you select the total number of channels controlled by ALTGX_RECONFIG instance as follows:

- 1 ≤Channels ≤4 then input port `reconfig_fromgxb` = 17 bits
- 5 ≤Channels ≤ 8 then input port `reconfig_fromgxb` = 34 bits
- 9 ≤Channels ≤12 then input port `reconfig_fromgxb` = 51 bits

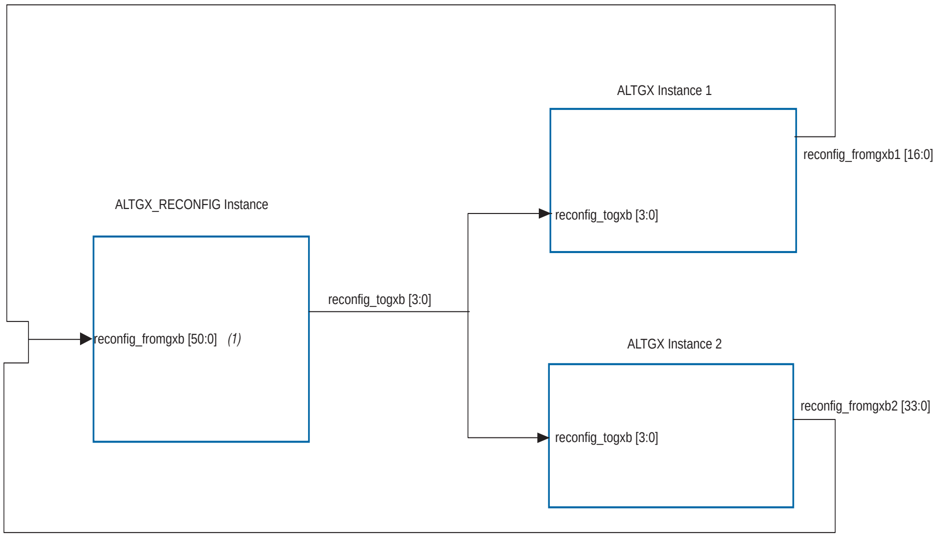
To connect the `reconfig_fromgxb` port between the ALTGX_RECONFIG instance and multiple ALTGX instances, follow these rules:

- Take the `reconfig_fromgxb[16:0]` of ALTGX instance 1 and connect it to the `reconfig_fromgxb[16:0]` of the ALTGX_RECONFIG instance. Connect the `reconfig_fromgxb[]` port of the next ALTGX instance to the next available bits of the ALTGX_RECONFIG instance, and so on.
- Similarly, connect the `reconfig_fromgxb` port of the ALTGX instance which has the highest **What is the starting channel number?** option to the MSB of the `reconfig_fromgxb` port of the ALTGX_RECONFIG instance.

The Quartus II Fitter produces an error if the dynamic reconfiguration option is enabled in the ALTGX instance, but the `reconfig_fromgxb` and `reconfig_togxb` ports are not connected to the ALTGX_RECONFIG instance.

Figure 5–13 illustrates how to connect the `reconfig_fromgxb` output port of the ALTGX instance to the `reconfig_fromgxb` input port of the ALTGX_RECONFIG instance.

Figure 5–13. 'reconfig_fromgxb' and 'reconfig_togxb' Connections between the ALTGX_RECONFIG Instance and ALTGX Instances



Note to Figure 5–13:

(1) `reconfig_fromgxb [50:0]` ={`reconfig_fromgxb[16:0]`, `reconfig_fromgxb[33:0]`}

Consider a design with two ALTGX instances:

- ALTGX instance 1 with one channel
- ALTGX instance 2 with five channels.

A single ALTGX_RECONFIG instance is controlling both these ALTGX instances.

The **What is the starting channel number?** option for ALTGX instance 1 is **0** (Refer to "Logical Channel Addressing" on page 5–14 for additional details.*)* This instance has a `reconfig_fromgxb1` output port. This port is 17 bits wide, because the number of channels selected in ALTGX instance 1 is one.

The **What is the starting channel number?** option for ALTGX instance 2 is **4** Refer to "Logical Channel Addressing" on page 5–14 for additional details.*)* This instance has a `reconfig_fromgxb2` output port. This port is 34 bits wide, because the number of channels selected in ALTGX instance 2 is five.

The **What is the number of channels controlled by the reconfig controller?** option in the **Reconfiguration settings** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager is **12** (Refer to "Total Number of Channels Controlled by the ALTGX_RECONFIG Instance" on page 5–25 for additional details).

The ALTGX_RECONFIG instance has a `reconfig_fromgxb` input port. This port is 51 bits wide.

You must connect `reconfig_fromgxb` input port of the ALTGX_RECONFIG instance to both the `reconfig_fromgxb1` output port of ALTGX instance 1 and `reconfig_fromgxb2` output port of ALTGX instance 2, as shown in the Figure 5–14. The lowest **What is the starting channel number?** option transceiver block is connected to the lowest significant bit and so on. The `reconfig_fromgxb1` of ALTGX instance 1 must be connected to the `reconfig_fromgxb[16:0]` of the ALTGX_RECONFIG instance. Similarly the `reconfig_fromgxb2` of ALTGX instance 2 must be connected to the `reconfig_fromgxb[50:17]` of the ALTGX_RECONFIG instance.

Offset Cancellation Control for Receiver Channels

As the silicon progresses towards smaller process nodes, the performance of circuits at these smaller nodes depends more on process variations. These process variations result in analog voltages to be offset from required ranges. The Stratix IV GX device provides an offset cancellation circuit per receiver channel to counter the offset variations due to process, voltage, and temperature. The offset cancellation logic corrects these offsets. The receiver buffer and receiver clock data recovery (CDR) require offset cancellation.

Offset cancellation is automatically executed once every time the device is powered on. The control logic for offset cancellation is integrated into the dynamic reconfiguration controller. You must connect the ALTGX_RECONFIG instance to the ALTGX instances with receiver channels in your design. You must connect the `reconfig_fromgxb`, `reconfig_togxb`, and necessary clock signals to both the ALTGX_RECONFIG and ALTGX (with receiver channels) instances.

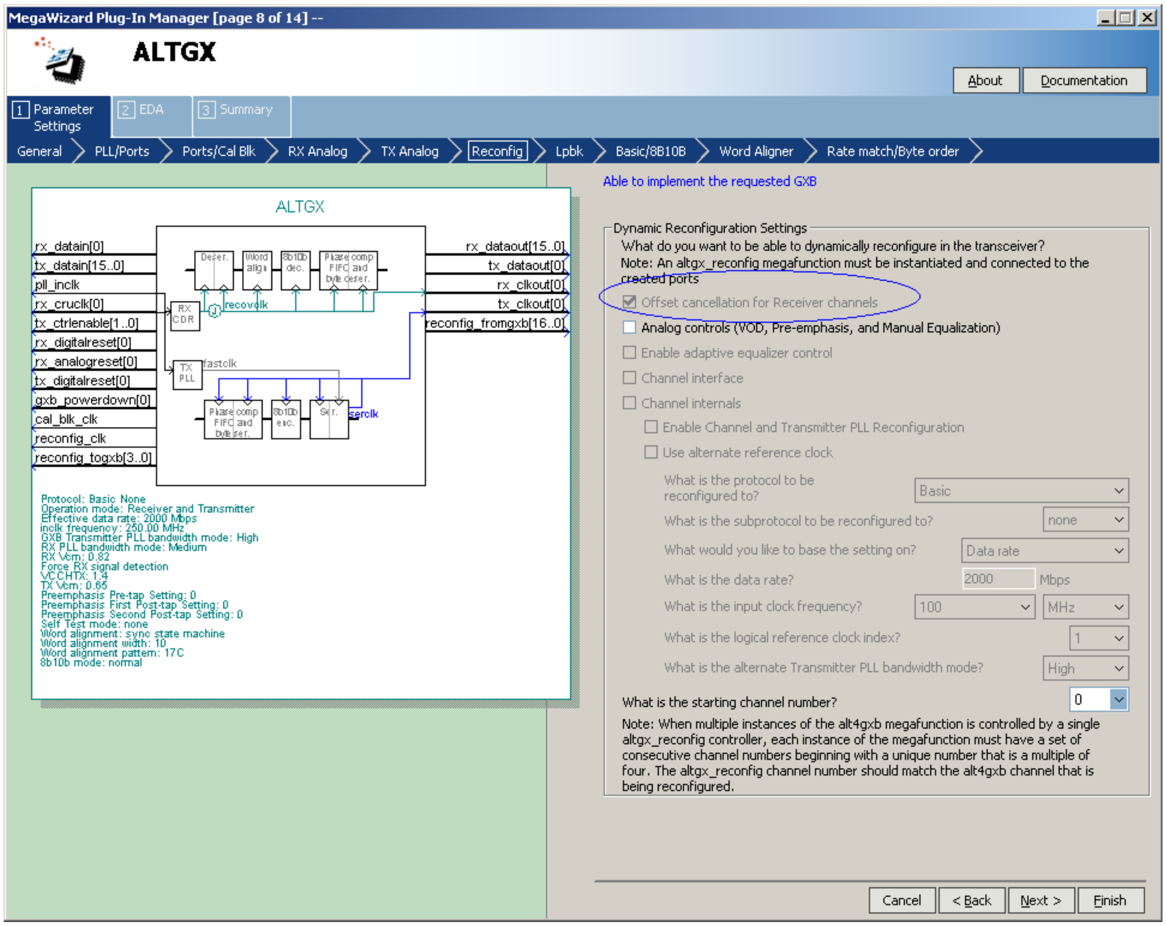
☞ For proper device operation, you must always connect the ALTGX_RECONFIG and ALTGX (with receiver channels) instances.

Operation

Every ALTGX instance for **Receiver and Transmitter** or **Receiver only** configuration requires that the **Offset cancellation for receiver channels** option is enabled in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager. This option is enabled by default for the above two configurations, as shown in Figure 5–14. It is disabled for **Transmitter only** configuration.

Figure 5–14 shows the **Offset Cancellation for Receiver Channels** option enabled by default in the ALTGX instance.

Figure 5–14. 'Offset Cancellation for Receiver Channels' Option in the ALTGX MegaWizard Plug-In Manager



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Dynamic Reconfiguration Settings

What do you want to be able to dynamically reconfigure in the transceiver?
Note: An altgx_reconfig megafunction must be instantiated and connected to the created ports

☑ Offset cancellation for Receiver channels
☐ Analog controls (VOD, Pre-emphasis, and Manual Equalization)
☐ Enable adaptive equalizer control
☐ Channel interface
☐ Channel internals
☐ Enable Channel and Transmitter PLL Reconfiguration
☐ Use alternate reference clock

What is the protocol to be reconfigured to? Basic
What is the subprotocol to be reconfigured to? none
What would you like to base the setting on? Data rate
What is the data rate? 2000 Mbps
What is the input clock frequency? 100 MHz
What is the logical reference clock index? 1
What is the alternate Transmitter PLL bandwidth mode? High

What is the starting channel number? 0

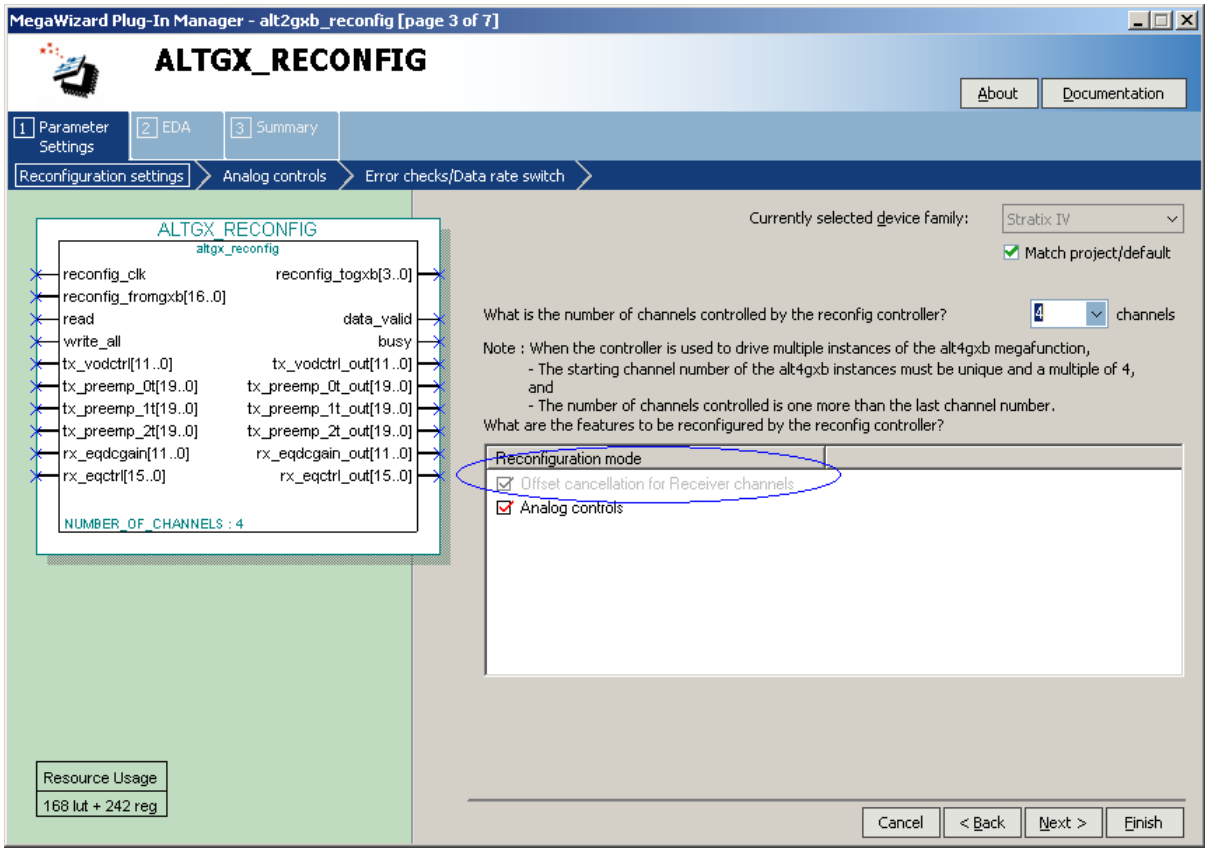
Note: When multiple instances of the alt4gxb megafunction is controlled by a single altgx_reconfig controller, each instance of the megafunction must have a set of consecutive channel numbers beginning with a unique number that is a multiple of four. The altgx_reconfig channel number should match the alt4gxb channel that is being reconfigured.

Because this option is enabled by default, the ALTGX instance must be connected to an ALTGX_RECONFIG instance (dynamic reconfiguration controller). The offset cancellation controls are also enabled by default in the **Reconfiguration settings** screen of the ALTGX_RECONFIG instance.

You must also set the starting channel number in the **What is the starting channel number?** option for every ALTGX instance connected to the ALTGX_RECONFIG instance. Refer to "Logical Channel Addressing" on page 5–14 for additional information about this option.

Figure 5–15 shows only the **Offset Cancellation for Receiver Channels** option enabled by default in the ALTGX_RECONFIG MegaWizard Plug-In Manager.

Figure 5–15. 'Offset Cancellation for Receiver Channels' Option in the ALTGX_RECONFIG MegaWizard Plug-In Manager



MegaWizard Plug-In Manager - alt2gxb_reconfig [page 3 of 7]

ALTGX_RECONFIG

Currently selected device family: Stratix IV

☑ Match project/default

What is the number of channels controlled by the reconfig controller? 4 channels

Note : When the controller is used to drive multiple instances of the alt4gxb megafunction,
- The starting channel number of the alt4gxb instances must be unique and a multiple of 4, and
- The number of channels controlled is one more than the last channel number.

What are the features to be reconfigured by the reconfig controller?

Reconfiguration mode
☑ Offset cancellation for Receiver channels
☑ Analog controls

Resource Usage
168 lut + 242 reg

When the device powers up, the dynamic reconfiguration controller initiates offset cancellation on the receiver channel by disconnecting the receiver input pins from the receiver data path. It also sets the receiver CDR into fixed set of dividers to guarantee a voltage controlled oscillator (VCO) clock rate within the range necessary to provide proper offset cancellation. Subsequently, the offset cancellation process goes through different states and culminates in the offset cancellation of the receiver buffer and the receiver CDR. After offset cancellation is completed, the user divider settings are restored.

The dynamic reconfiguration controller sends and receives data to the transceiver channel through the `reconfig_togxb` and `reconfig_fromgxb` signals. You must connect these signals between the ALTGX_RECONFIG instance and the ALTGX_instance. You must also set the **What is the number of channels controlled by the reconfig controller?** option in the **Reconfiguration settings** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager. Refer to "Total Number of Channels Controlled by the ALTGX_RECONFIG Instance" on page 5–25 for details on how to set this option.

The **Use 'logical_channel_address' port** option in the **Analog controls** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager is not applicable for receiver offset cancellation process.

☞ If the design does not require PMA controls reconfiguration and uses optimum LE resources, you can connect all the ALTGX instances in the design to a single dynamic reconfiguration controller (ALTGX_RECONFIG instance).

☞ The `gxb_powerdown` signal must not be asserted during the offset cancellation sequence.

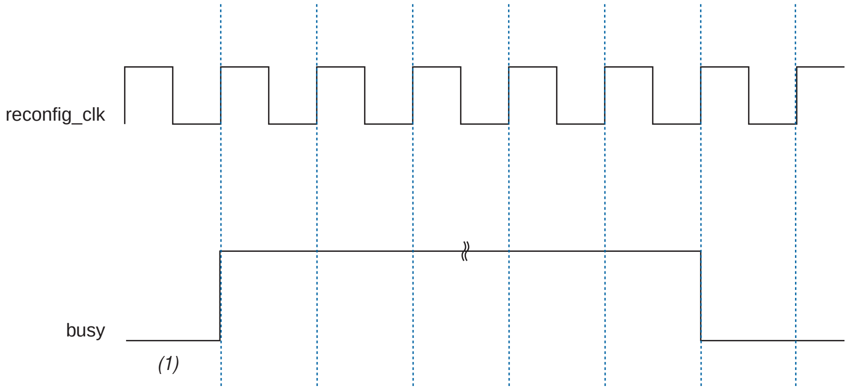
☞ Refer to "PMA Controls Reconfiguration Duration" on page 5–58 to understand the impact on system bring up when you control all the transceiver channels using a single dynamic reconfiguration controller.

Consider the scenario where the design has ALTGX instances with channels of both **Transmitter only** and **Receiver only** configurations. You must include the **Transmitter only** channels also while setting the **What is the starting channel number?** option in the ALTGX instance and while setting the **What is the number of channels controlled by the reconfig controller?** option in the the ALTGX_RECONFIG instance for the purpose of receiver offset cancellation.

- After the device powers up, the `busy` signal remains low for the very first `reconfig_clk` clock cycle.
- It then gets asserted for the second `reconfig_clk` clock cycle, when the dynamic reconfiguration controller initiates the offset cancellation process.
- The de-assertion of the `busy` signal indicates the successful completion of the offset cancellation process.

Figure 5–16 shows the dynamic reconfiguration signals transition during offset cancellation on receiver channels.

Figure 5–16. Dynamic Reconfiguration Signals Transition during Offset Cancellation on Receiver Channels



Note to Figure 5–16:

(1) After device power up, the busy signal remains low for the first `reconfig_clk` cycle.

☞ Due to the offset cancellation process, the transceiver reset sequence has changed. Refer to the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook* chapter for additional information.

Example for the Offset Cancellation Process of a Receiver Channel

The following example describes the offset cancellation process of a receiver channel.

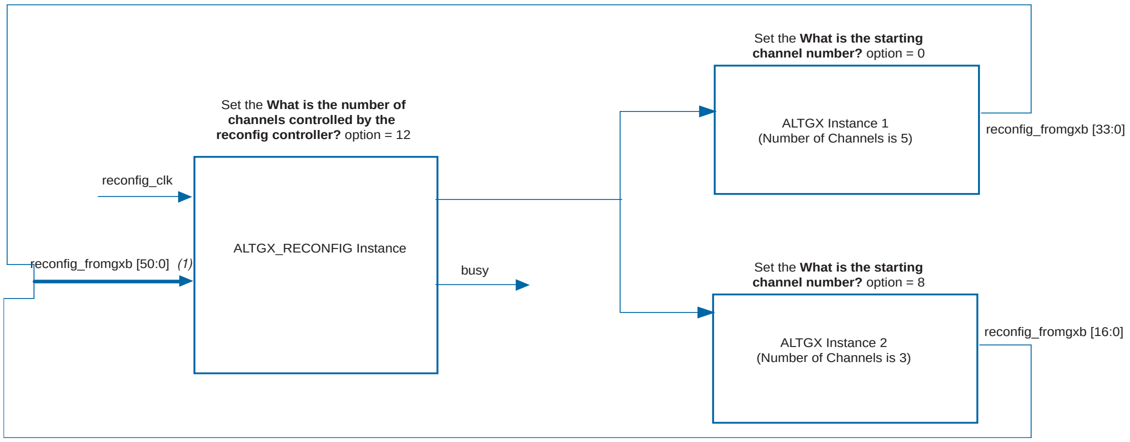
Consider a design with two ALTGX instances:

- ALTGX instance 1 with five transceiver channels
- ALTGX instance 2 with three transceiver channels

You must always connect the ALTGX_RECONFIG instance to the ALTGX instances. Even if you do not require PMA controls reconfiguration, you must set the **What is the starting channel number?** option in the **Reconfig** screen of both the ALTGX MegaWizard Plug-In Managers. You must also set the **What is the number of channels controlled by the reconfig controller?** option in the **Reconfiguration settings** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager, as shown in Figure 5–12 on page 5–25.

Figure 5–17 shows the ALTGX_RECONFIG instance connected to both the ALTGX_instance 1 and ALTGX_instance 2.

Figure 5–17. Example of the ALTGX_RECONFIG Instance and ALTGX Instances Set Up for the Offset Cancellation Process of a Receiver Channel


Set the **What is the starting channel number?** option = 0
Set the **What is the number of channels controlled by the reconfig controller?** option = 12
ALTGX Instance 1
(Number of Channels is 5)
reconfig_fromgxb [33:0]
reconfig_clk
ALTGX_RECONFIG Instance
reconfig_fromgxb [50:0] (1)
busy
Set the **What is the starting channel number?** option = 8
ALTGX Instance 2
(Number of Channels is 3)
reconfig_fromgxb [16:0]

Note to Figure 5–17:

(1) reconfig_fromgxb [50:0] = {reconfig_fromgxb[33:0], reconfig_fromgxb[16:0]}

The following are the typical steps that ensure proper device operation:

Five Channel Transceiver Instance (ALTGX Instance 1)

1. In the ALTGX MegaWizard Plug-In Manager, set the **What is the number of channels?** option in the **General** screen to **5**.
2. The output signal `reconfig_fromgxb` is transceiver block based, so the number of bits for this instance is 34. The input signal `reconfig_togxb` is a fixed bus (four bits wide).
3. Set the **What is the starting channel number?** option in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager to **0**. Refer to "Logical Channel Addressing" on page 5–14 for additional details.

Three Channel Transceiver Instance (ALTGX Instance 2)

1. In the ALTGX MegaWizard Plug-In Manager, set the **What is the number of channels?** option in the **General** screen to **3**.

2. The output signal `reconfig_fromgxb` is transceiver block based, so the number of bits for this instance is 17. The input signal `reconfig_togxb` is a fixed bus (four bits wide).
3. Set the **What is the starting channel number?** option in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager to **8**. This example has multiple ALTGX instances controlled by one dynamic reconfiguration controller. You must therefore set the **What is the starting channel number?** option as multiples of four. Refer to "Logical Channel Addressing" on page 5–14 for additional details.

Dynamic Reconfiguration Controller Instance (ALTGX_RECONFIG Instance)

1. Launch the ALTGX_RECONFIG MegaWizard Plug-In Manager.
2. Set the **What is the number of channels controlled by the reconfig controller?** option in the **Reconfiguration settings** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager to **12**. The setting for this option has a number that is more than the total number of channels required to be controlled (eight channels) by dynamic reconfiguration. Refer to "Total Number of Channels Controlled by the ALTGX_RECONFIG Instance" on page 5–25 for more information about this setting.
3. The input signal `reconfig_fromgxb` is transceiver block based, so the width of this signal is 51 (3×17) bits.

ALTGX Instances and ALTGX_RECONFIG Instances Connections

1. Connect the `reconfig_fromgxb [33:0]` output port from ALTGX instance 1 to the `reconfig_fromgxb [33:0]` input port of the ALTGX_RECONFIG instance.
2. Similarly connect the `reconfig_fromgxb [16:0]` output port from ALTGX instance 2 to the `reconfig_fromgxb [50:17]` input port of the ALTGX_RECONFIG instance.
3. Connect the `reconfig_togxb[3:0]` output port of the ALTGX_RECONFIG instance to the `reconfig_togxb[3:0]` input ports of both the ALTGX instance 1 and ALTGX instance 2.

Refer to "Connecting reconfig_from_gxb/reconfig_to_gxb Ports" on page 5–28 for additional details.

Dynamic Reconfiguration Controller-Offset Cancellation Control Sequence

1. The ALTGX_RECONFIG instance automatically performs offset cancellation on all the receiver channels of the ALTGX instances connected to it, on power up.
2. The `busy` signal is low for the first `reconfig_clk` clock cycle after power up.
3. The `busy` signal gets asserted for the second `reconfig_clk` clock cycle after power up.
4. The de-assertion of the `busy` signal indicates the successful completion of the offset cancellation process.

PMA Controls Reconfiguration

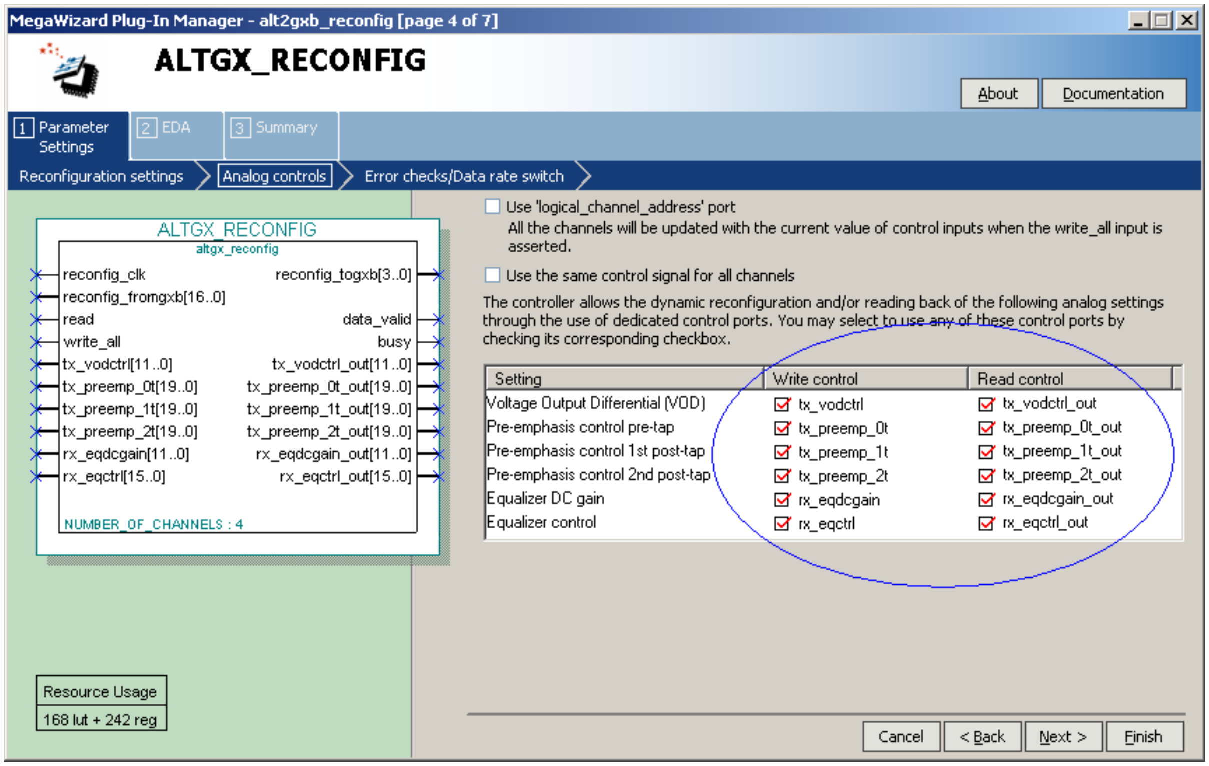
The various PMA controls that can be reconfigured are:

- Pre-emphasis settings
- Equalization settings
- DC gain settings
- Voltage output differential (VOD) settings

Dynamic Reconfiguration Controller Ports for PMA controls

The ALTGX_RECONFIG MegaWizard Plug-In Manager has the PMA control ports available in the **Analog controls** screen. Depending on which of the PMA controls you want to reconfigure, you can select the appropriate PMA control ports (example: `tx_vodctrl` to write new VOD settings, `tx_vodctrl_out` to read the existing VOD settings) as shown in Figure 5–18.

Figure 5–18. Dynamic Reconfiguration Controller Ports for PMA Controls in the ALTGX_RECONFIG MegaWizard Plug-In Manager



□ Use 'logical_channel_address' port
All the channels will be updated with the current value of control inputs when the write_all input is asserted.

□ Use the same control signal for all channels

The controller allows the dynamic reconfiguration and/or reading back of the following analog settings through the use of dedicated control ports. You may select to use any of these control ports by checking its corresponding checkbox.

Setting	Write control	Read control
Voltage Output Differential (VOD)	☑ tx_vodctrl	☑ tx_vodctrl_out
Pre-emphasis control pre-tap	☑ tx_preemp_0t	☑ tx_preemp_0t_out
Pre-emphasis control 1st post-tap	☑ tx_preemp_1t	☑ tx_preemp_1t_out
Pre-emphasis control 2nd post-tap	☑ tx_preemp_2t	☑ tx_preemp_2t_out
Equalizer DC gain	☑ rx_eqdcgain	☑ rx_eqdcgain_out
Equalizer control	☑ rx_eqctrl	☑ rx_eqctrl_out

Dynamically Reconfiguring PMA Controls

There are two methods by which the PMA controls of a transceiver channel can be dynamically reconfigured.

- Method 1

  The PMA controls of a specific transceiver channel can be reconfigured. This method is explained in detail in "Method 1" on page 5–39.

- Method 2

  Dynamically reconfiguring the PMA controls of the transceiver channels without using the `logical_channel_address` port. If you use this method, the PMA controls of all the transceiver channels connected to the dynamic reconfiguration controller are reconfigured. This method is explained in detail in "Method 2" on page 5–43.

For both these methods, you can additionally use the `rx_tx_duplex_sel` port. The width of this port is fixed to 2-bits.

You can enable this port by selecting the **Use 'rx_tx_duplex_sel' port to enable RX only, TX only or duplex reconfiguration** option in the **Error checks/Data rate switch** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager, as shown in Figure 5–19.

This option is available only when you select the **Analog controls** option in the **Reconfiguration settings** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager. Table 5–5 shows the allowed values for this port.

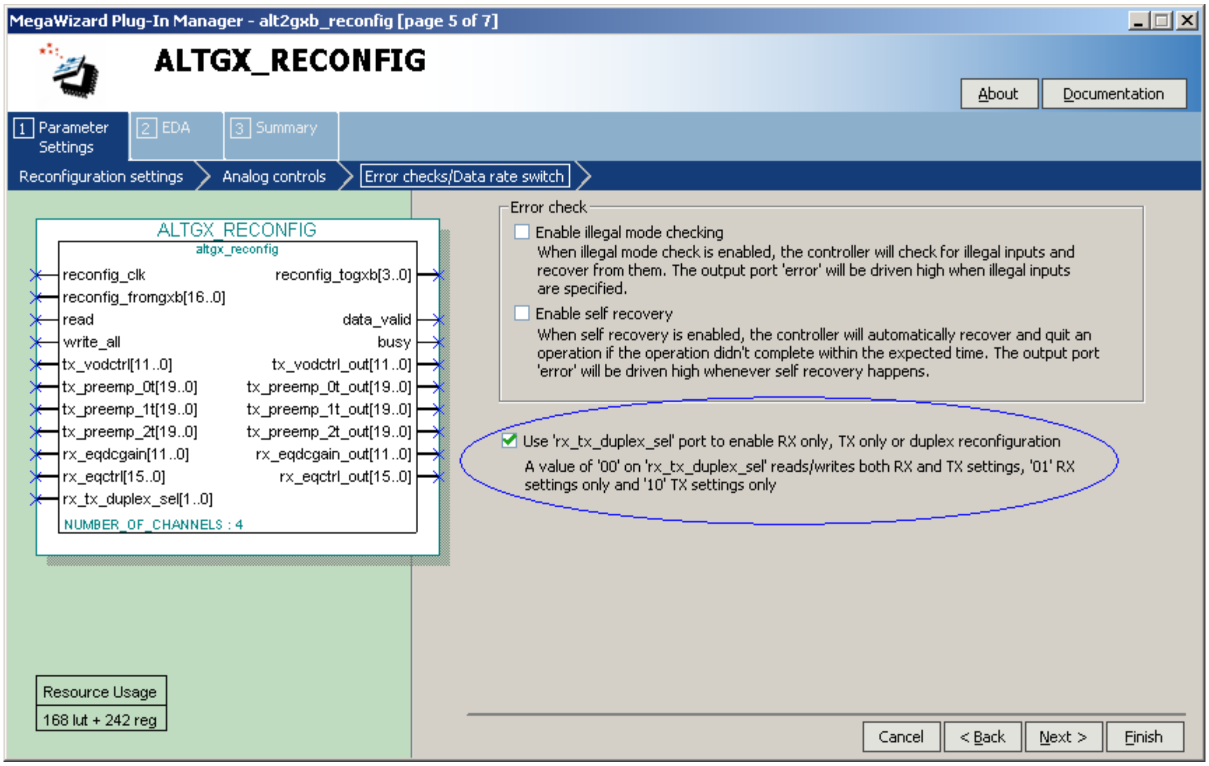
Table 5–5. Setting the rx_tx_duplex_sel Input Port of the ALTGX_RECONFIG Instance *(Note 1)*

rx_tx_duplex_sel	Reconfiguration Mode
00	Receiver and Transmitter
01	Receiver only
10	Transmitter only
11	Unsupported value

Note to Table 5–5:

(1) Refer to "Dynamic Reconfiguration Controller Port List" on page 5–7 for additional details on this port.

Figure 5–19. Use 'rx_tx_duplex_sel' port to enable RX only, TX only or duplex reconfiguration Option in the ALTGX_RECONFIG MegaWizard Plug-In Manager



MegaWizard Plug-In Manager - alt2gxb_reconfig [page 5 of 7]

ALTGX_RECONFIG

Error check

☐ Enable illegal mode checking
When illegal mode check is enabled, the controller will check for illegal inputs and recover from them. The output port 'error' will be driven high when illegal inputs are specified.

☐ Enable self recovery
When self recovery is enabled, the controller will automatically recover and quit an operation if the operation didn't complete within the expected time. The output port 'error' will be driven high whenever self recovery happens.

☑ Use 'rx_tx_duplex_sel' port to enable RX only, TX only or duplex reconfiguration
A value of '00' on 'rx_tx_duplex_sel' reads/writes both RX and TX settings, '01' RX settings only and '10' TX settings only

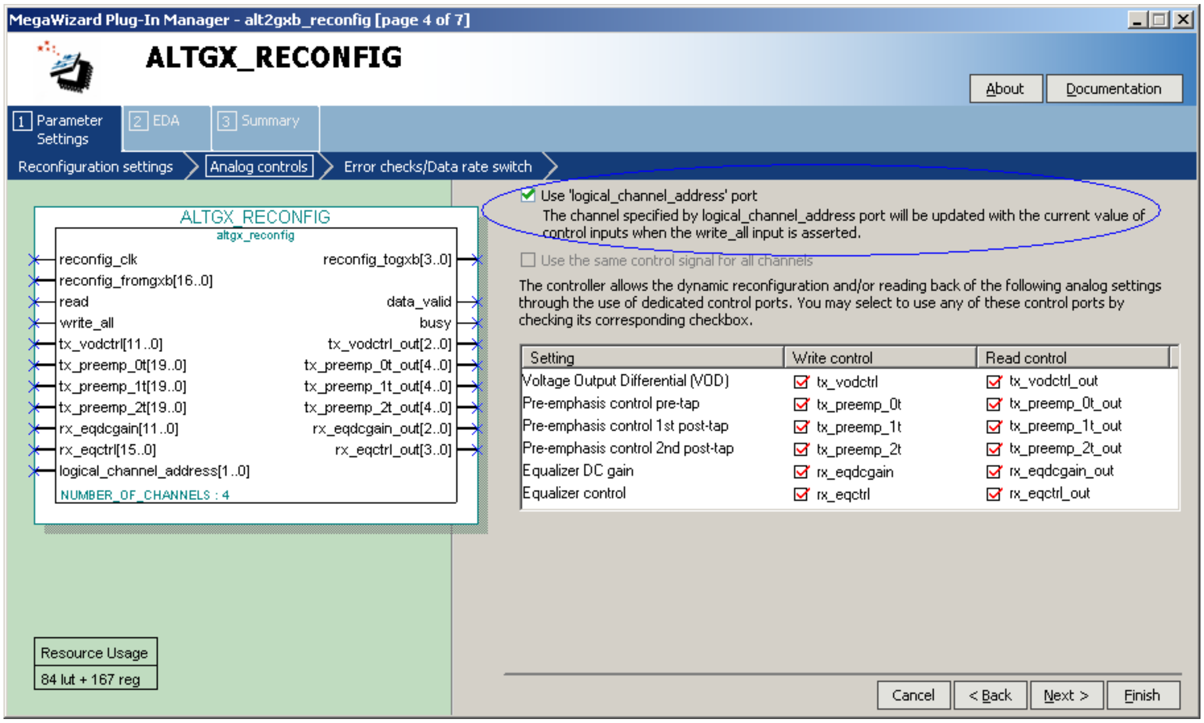
Resource Usage
168 lut + 242 reg

The two methods are discussed in detail in the following sections.

Method 1

Using this method, you can dynamically reconfigure the PMA controls of a transceiver channel by using the `logical_channel_address` port without affecting the remaining active channels. The `logical_channel_address_port` port can be enabled by selecting the **Use 'logical_channel_address' port** option in the **Analog controls** screen, as shown in Figure 5–20. This method is applicable only for a design where the dynamic reconfiguration controller controls more than one channel.

Figure 5–20. The "Use 'logical_channel_address' port" Option in the ALTGX_RECONFIG MegaWizard Plug-In Manager



Resource Usage
84 lut + 167 reg

☑ Use 'logical_channel_address' port

The channel specified by logical_channel_address port will be updated with the current value of control inputs when the write_all input is asserted.

□ Use the same control signal for all channels

The controller allows the dynamic reconfiguration and/or reading back of the following analog settings through the use of dedicated control ports. You may select to use any of these control ports by checking its corresponding checkbox.

Setting	Write control	Read control
Voltage Output Differential (VOD)	☑ tx_vodctrl	☑ tx_vodctrl_out
Pre-emphasis control pre-tap	☑ tx_preemp_0t	☑ tx_preemp_0t_out
Pre-emphasis control 1st post-tap	☑ tx_preemp_1t	☑ tx_preemp_1t_out
Pre-emphasis control 2nd post-tap	☑ tx_preemp_2t	☑ tx_preemp_2t_out
Equalizer DC gain	☑ rx_eqdcgain	☑ rx_eqdcgain_out
Equalizer control	☑ rx_eqctrl	☑ rx_eqctrl_out

You can additionally reconfigure either the receiver portion or transmitter portion or both the receiver and transmitter portions of the transceiver channel by setting the corresponding value on the `rx_tx_duplex_sel` input port.

The following section describes how to connect the PMA controls when using Method 1.

Connecting the PMA Control Ports

When using Method 1, the selected PMA control ports remain fixed in width, irrespective of the number of channels controlled by the ALTGX_RECONFIG instance:

- `tx_vodctrl` and `tx_vodctrl_out` are fixed to 3-bits
- `tx_preemp_0t`, `tx_preemp_1t`, `tx_preemp_2t`, `tx_preemp_0t_out`, `tx_preemp_1t_out` and `tx_preemp_2t_out` are fixed to 5-bits
- `rx_eqdcgain` and `rx_eqdcgain_out` are fixed to 3-bits
- `rx_eqctrl` and `rx_eqctrl_out` are fixed to 4-bits

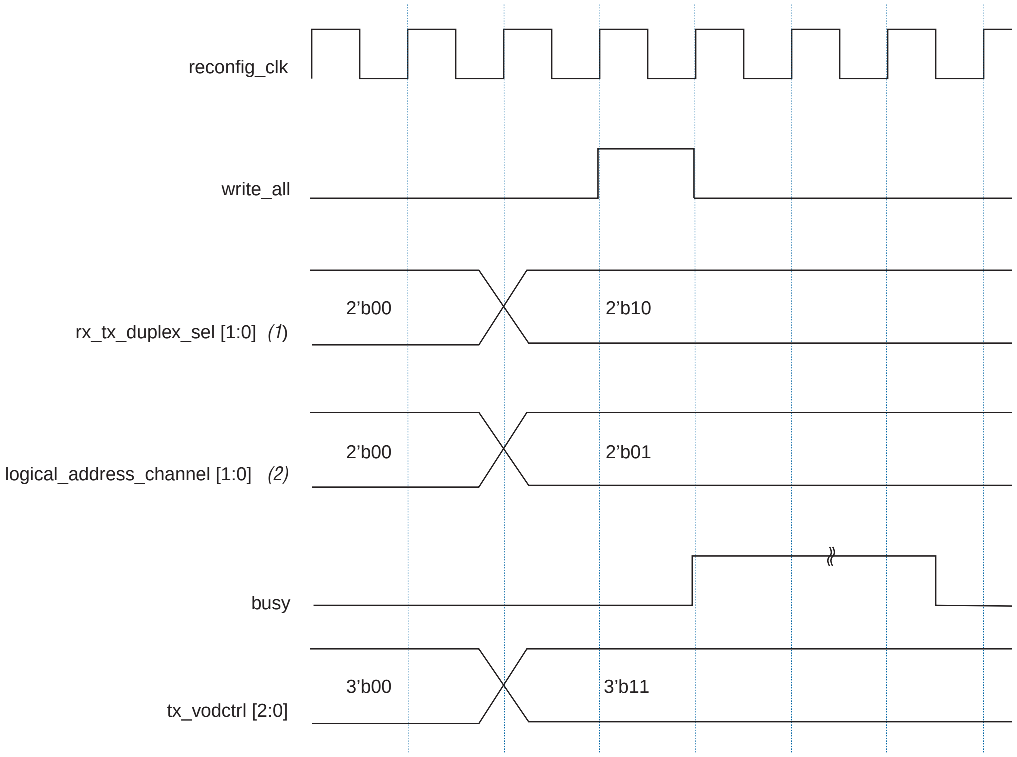
Write Transaction

Set the selected PMA control ports to the desired settings. (For example: `tx_vodctrl` = 3'b000). Set the input port `logical_channel_address` to the logical channel address of the transceiver channel whose PMA controls you want to reconfigure. Set the `rx_tx_duplex_sel` port to 2'b10 so that only the transmit PMA controls are written to the transceiver channel. Make sure that the `busy` signal is low, before you start a write transaction. Assert the `write_all` signal for one `reconfig_clk` clock cycle. This initiates the write transaction.

The `busy` output status signal is asserted high to indicate that the dynamic reconfiguration controller is busy writing the PMA control values. When the write transaction has completed, the `busy` signal goes low.

The write transaction waveform is depicted in Figure 5–21.

Figure 5–21. Method 1- Write Transaction Waveform



Notes to Figure 5–21:

(1) Consider that you want to write to only the transmitter portion of the channel.

(2) This waveform assumes that the number of channels connected to the dynamic reconfiguration controller is four. Hence, the `logical_channel_address` port is two bits wide.

Read Transaction

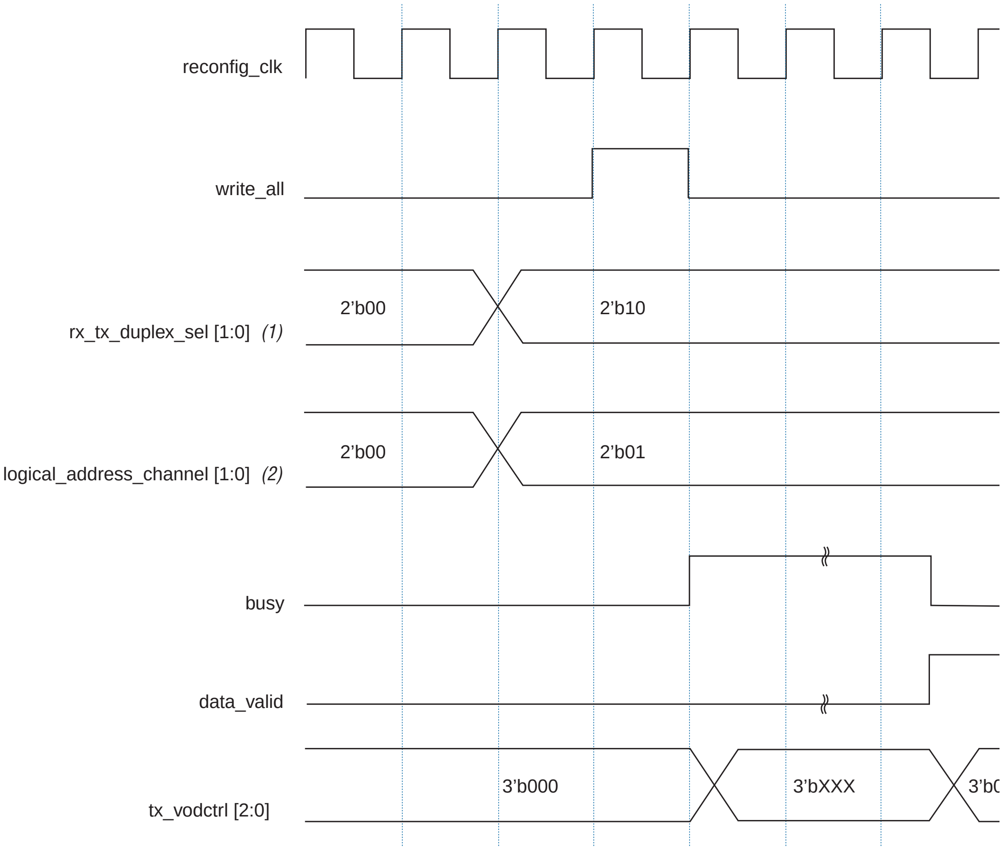
Consider the scenario where you want to read the existing VOD values from the transmit VOD control registers of the transmitter portion of a specific channel controlled by the ALTGX_RECONFIG instance. The read transaction in this scenario is explained in the following steps:

1. Set the input port `logical_channel_address` to the logical channel address of the transceiver channel whose PMA controls you want to read (for example: `tx_vodctrl_out`).
2. Set the `rx_tx_duplex_sel` port to 2'b10 so that only the transmit PMA controls are read from the transceiver channel.
3. Ensure that the `busy` signal is low, before you start a read transaction.
4. Assert the `read` signal for one `reconfig_clk` clock cycle. This initiates the read transaction.

The `busy` output status signal is asserted high to indicate that the dynamic reconfiguration controller is busy reading the PMA control values. Once the read transaction has completed, the `busy` signal goes low. The `data_valid` signal gets asserted indicating that the data available at the read control signal is valid.

The read transaction waveform is depicted in the following Figure 5–22.

Figure 5–22. Method 1- Read Transaction Waveform



Notes to Figure 5–22:

(1) Consider that you want to read from only the transmitter portion of the channel.
(2) This waveform assumes that the number of channels connected to the dynamic reconfiguration controller is four. Hence, the `logical_channel_address` port is two bits wide.

Simultaneous write and read transactions are not allowed.

Method 2

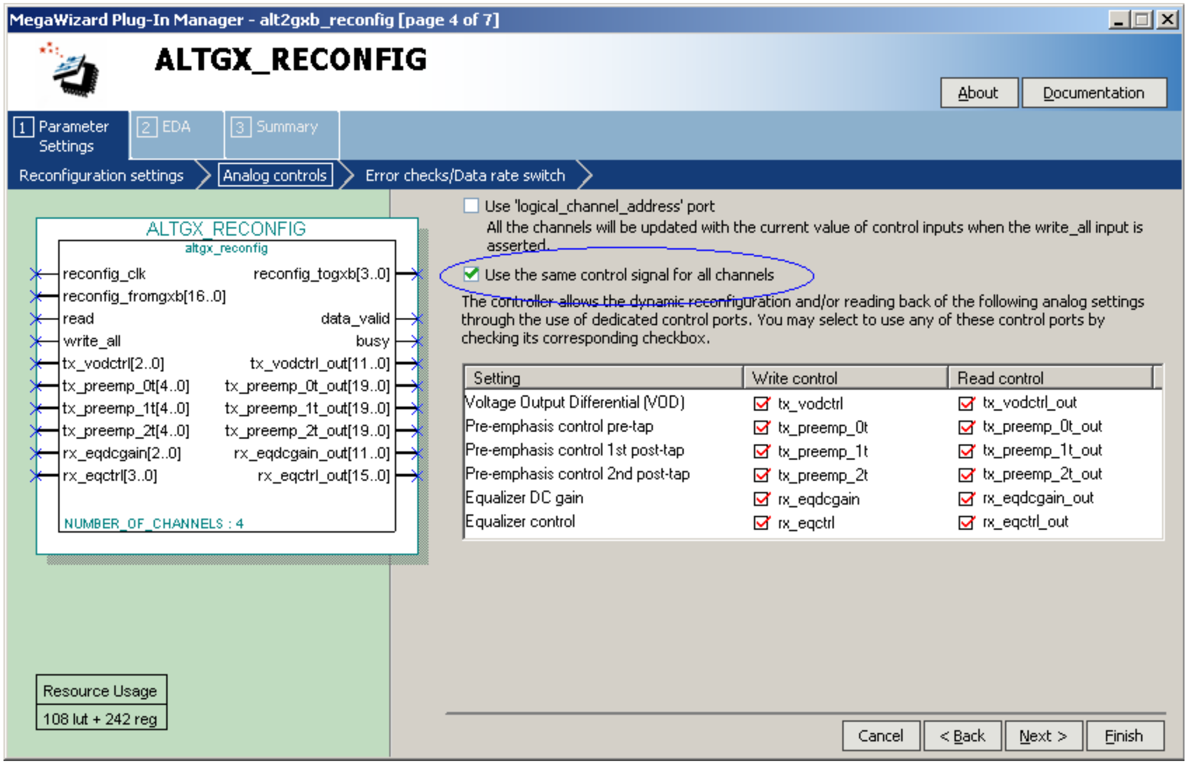
This method does not require the `logical_channel_address` port to dynamically reconfigure the PMA controls of the transceiver channels. With this method, the PMA controls of all the transceiver channels connected to the ALTGX_RECONFIG instance are reconfigured.

Method 2 can be further classified into:

- The **Use the same control signal for all the channels** option enabled
- The **Use the same control signal for all the channels** option disabled.

Figure 5–23 shows the **Use the same control signal for all the channels** option in the ALTGX_RECONFIG MegaWizard.

Figure 5–23. 'Use the same control signals for all the channels' Option in the ALTGX_RECONFIG MegaWizard Plug-In Manager



☐ Use 'logical_channel_address' port

All the channels will be updated with the current value of control inputs when the write_all input is asserted.

☑ Use the same control signal for all channels

The controller allows the dynamic reconfiguration and/or reading back of the following analog settings through the use of dedicated control ports. You may select to use any of these control ports by checking its corresponding checkbox.

Setting	Write control	Read control
Voltage Output Differential (VOD)	☑ tx_vodctrl	☑ tx_vodctrl_out
Pre-emphasis control pre-tap	☑ tx_preemp_0t	☑ tx_preemp_0t_out
Pre-emphasis control 1st post-tap	☑ tx_preemp_1t	☑ tx_preemp_1t_out
Pre-emphasis control 2nd post-tap	☑ tx_preemp_2t	☑ tx_preemp_2t_out
Equalizer DC gain	☑ rx_eqdcgain	☑ rx_eqdcgain_out
Equalizer control	☑ rx_eqctrl	☑ rx_eqctrl_out

The following sections discuss the two classifications in detail:

The 'Use the same control signal for all the channels' Option Enabled

The **Use the same control signal for all the channels** option is available in the **Analog controls** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager. If enabled, the width of the PMA control ports are fixed as shown below:

PMA Control Ports Used in a Write Transaction

- `tx_vodctrl` is fixed to 3-bits.
- `tx_preemp_0t`, `tx_preemp_1t`, `tx_preemp_2t` are fixed to 5-bits.
- `rx_eqdcgain` is fixed to 3-bits.
- `rx_eqctrl` is fixed to 4-bits.

PMA Control Ports Used in a Read Transaction

- `tx_vodctrl_out` is 3-bits per channel.
- `tx_preemp_0t_out`, `tx_preemp_1t_out`, and `tx_preemp_2t_out` are 5-bits per channel.
- `rx_eqdcgain_out` is 3-bits per channel.
- `rx_eqctrl_out` is to 4-bits per channel.

For example, if the number of channels controlled by the dynamic reconfiguration controller is two, `tx_vodctrl_out` will be 6 bits wide.

Write Transaction

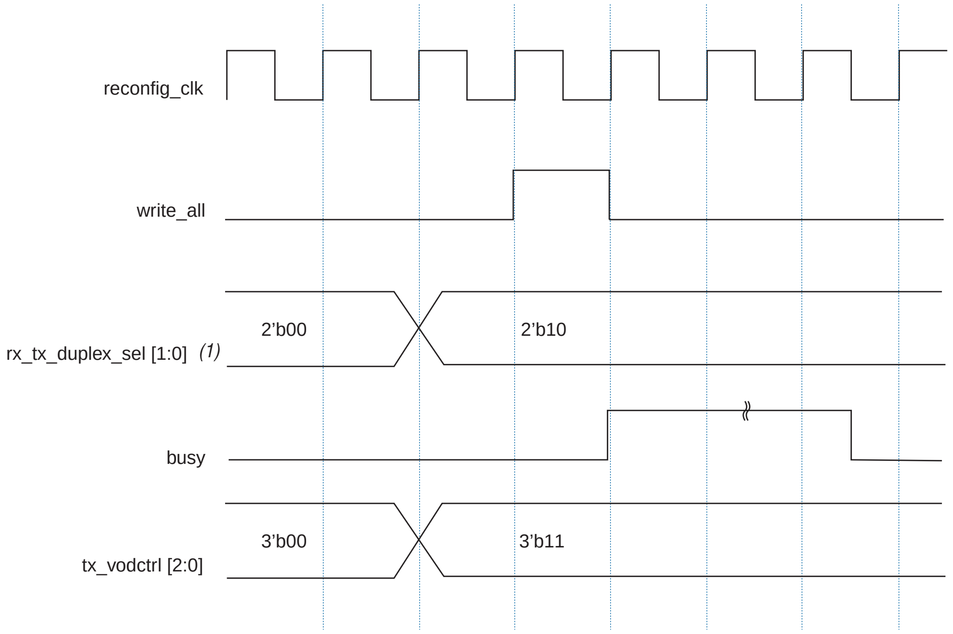
The value you set at the selected PMA control ports gets written to all the transceiver channels connected to the ALTGX_RECONFIG instance.

Consider that you have enabled the `tx_vodctrl` in the ALTGX_RECONFIG MegaWizard Plug-In Manager to reconfigure the VOD of the transceiver channels.

The following are the steps involved in the write transaction to reconfigure the VOD, as shown in Figure 5–24:

1. Before you initiate a write transaction, set the selected PMA control ports to the desired settings. (For example: `tx_vodctrl` = 3'b000).
2. Set the `rx_tx_duplex_sel` port to 2'b10 so that only the transmit PMA controls are written to the transceiver channel.
3. Make sure that the `busy` signal is low before you start a write transaction.
4. Assert the `write_all` signal for one `reconfig_clk` clock cycle. This initiates the write transaction.
5. The `busy` output status signal is asserted high to indicate that the dynamic reconfiguration controller is busy writing the PMA control values. Once the write transaction has completed, the `busy` signal goes low.

Figure 5–24. Method 2 - Write Transaction Waveform - 'Use the same control signal for all the channels' Option Enabled



Note to Figure 5–24:

(1) Consider that you want to write to only the transmitter portion of the channel.

Read Transaction

If you want to read the existing values from a specific channel connected to the ALTGX_RECONFIG instance, observe the corresponding byte positions of the PMA control output port after the read transaction is completed.

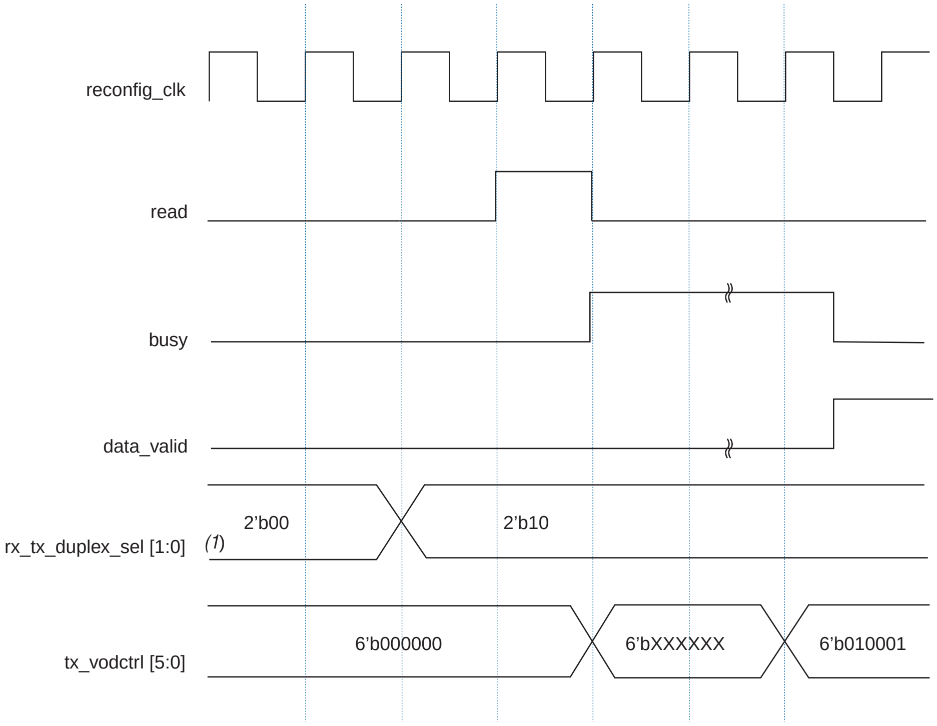
For example, if the number of channels controlled by the ALTGX_RECONFIG is 2, the `tx_vodctrl_out` is 6-bits wide. The `tx_vodctrl_out[2:0]` corresponds to channel 1, and similarly `tx_vodctrl_out[5:3]` corresponds to channel 2.

The following list shows the steps to read the VOD values of the second channel:

1. Before you initiate a read transaction, set the `rx_tx_duplex_sel` port to 2'b10 so that only the transmit PMA controls are read from the transceiver channel.
2. Make sure that the `busy` signal is low, before you start a read transaction.
3. Assert the `read` signal for one `reconfig_clk` clock cycle. This initiates the read transaction.
4. The `busy` output status signal is asserted high to indicate that the dynamic reconfiguration controller is busy reading the PMA control settings.
5. Once the read transaction has completed, the `busy` signal goes low. The `data_valid` signal gets asserted indicating that the data available at the read control signal is valid. To read the current VOD values in channel 2, observe the values in `tx_vodctrl_out[5:3]`.

The waveform shown in Figure 5–25 assumes that the transmit VOD settings written in channels 1 and 2 prior to the read transaction are 3'b001 and 3'b010, respectively.

Figure 5–25. Method 2- Read Transaction Waveform



Note to Figure 5–25:

(1) Consider that you want to read from only the transmitter portion of all the channels.

☞ Simultaneous write and read transactions are not allowed.

The 'Use the same control signal for all the channels' Option Disabled

When this option is disabled, the PMA control ports for write transaction are separate for each channel.

PMA Control Ports Used in a Write Transaction

- `tx_vodctrl` is 3-bits per channel.
- `tx_preemp_0t`, `tx_preemp_1t`, and `tx_preemp_2t` are 5-bits per channel.
- `rx_eqdcgain` is 3-bits per channel.
- `rx_eqctrl` is to 4-bits per channel.

For example, if you have two channels, the `tx_vodctrl` is 6-bits wide (`tx_vodctrl [2:0]` corresponds to channel 1 and `tx_vodctrl [5:3]` corresponds to channel 2).

PMA Control Ports Used in a Read Transaction

The width of the PMA control ports for read transaction are always separate for each channel (same as the PMA control ports explained in "The 'Use the same control signal for all the channels' Option Enabled" on page 5–43).

Write Transaction

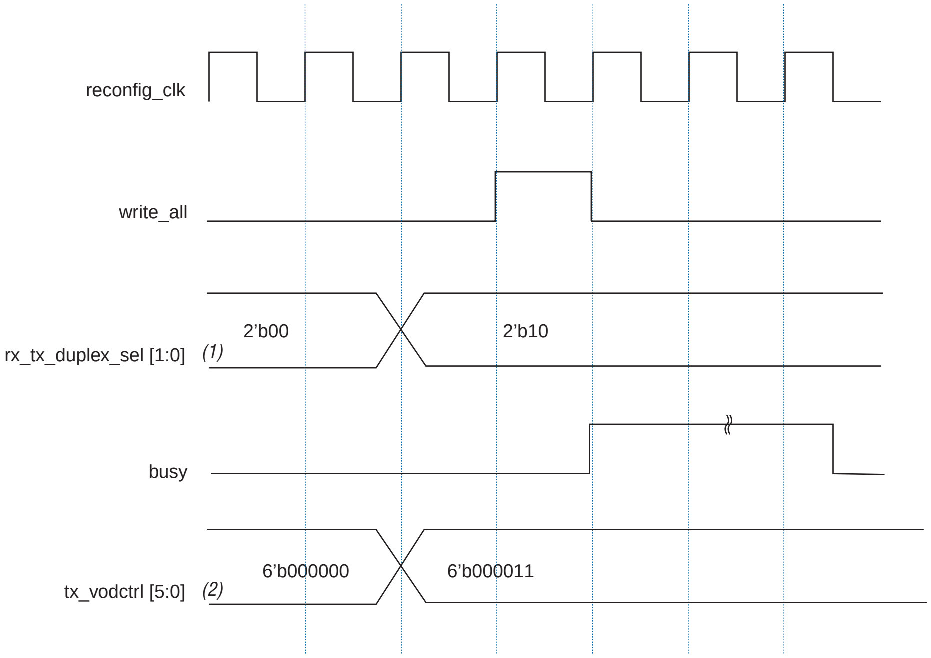
Since the PMA controls of all channels are written, if you want to reconfigure a specific channel connected to the ALTGX_RECONFIG instance, set the new value at the corresponding PMA control port of the channel under consideration and retain the previously stored values in the other active channels using a read transaction prior to this write transaction.

For example, assume that the number of channels controlled by the ALTGX_RECONFIG is two, the `tx_vodctrl` in this case is 6-bits wide. The `tx_vodctrl [2:0]` corresponds to channel 1 and similarly `tx_vodctrl [5:3]` corresponds to channel 2.

1. If you want to dynamically reconfigure the PMA controls of only the channel 2 with a new value, first perform a read transaction to retrieve the existing PMA control values from the `tx_vodctrl_out[5:0]`. Take the `tx_vodctrl_out [2:0]` and provide this value in `tx_vodctrl [2:0]` to write in channel 1. By doing so, channel 1 gets overwritten with the same value.
2. Perform a write transaction. This ensures that the new values are written only to channel 2, while channel 1 remains unchanged.

Figure 5–26 illustrates a write transaction waveform with the **Use the same control signal for all the channels** option disabled.

Figure 5–26. Method 2 - Write Transaction Waveform with the Use the same control signal for all the channels Option Disabled



Notes to Figure 5–26:

(1) Consider that you want to write to only the transmitter portion of the channel.

(2) The waveform assumes that the number of channels controlled by the dynamic reconfiguration controller (ALTGX_RECONFIG instance) is two and that `tx_vodctrl` control port is enabled.

☞ Simultaneous write and read transactions are not allowed.

Read Transaction

The read transaction is explained in "Read Transaction" on page 5–45.

Examples: Dynamic Reconfiguration Controller (ALTGX_RECONFIG)

The following design examples illustrate the various possible topologies of the dynamic reconfiguration controller with ALTGX instances. The first two design examples specifically discuss a single controller controlling multiple instances of an ALTGX megafunction and a single controller controlling one instance of an ALTGX megafunction. Design example three discusses the HDL construct requirements if you are stamping the ALTGX instances. Each ALTGX instance in turn can have more than one transceiver channel. The dynamic reconfiguration of PMA controls is enabled for all the design examples.

Example 1: One Reconfiguration Controller Connected to Multiple ALTGX Instances

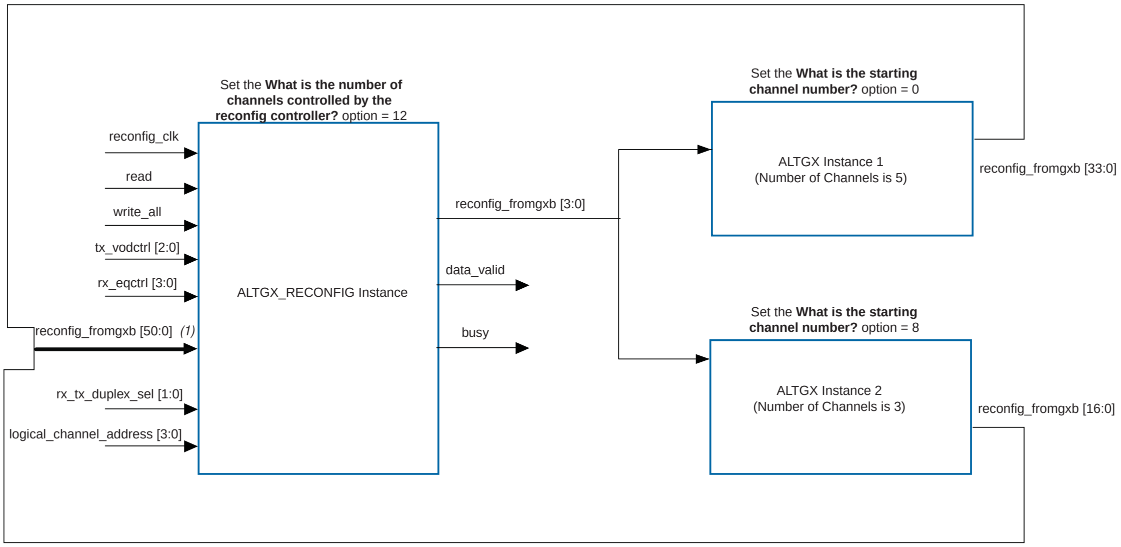
Consider a design with two ALTGX instances: ALTGX instance 1 with five transceiver channels and ALTGX instance 2 with three transceiver channels.

Assume the following for this example:

- ALTGX instance 1 and ALTGX instance 2 cannot be packed into the same transceiver block physically.
- One dynamic reconfiguration controller controls both the ALTGX instances.
- You want to dynamically reconfigure the transmit VOD PMA control (`tx_vodctrl`) of the first channel of ALTGX instance 1 and receiver equalization PMA control (`rx_eqctrl`) of the second channel of the ALTGX instance 2.

Figure 5–27 shows the ALTGX_RECONFIG instance connected to both the ALTGX_instance 1 and ALTGX_instance 2.

Figure 5–27. Example 1 for PMA Controls Reconfiguration


Set the **What is the number of channels controlled by the reconfig controller?** option = 12
reconfig_clk
read
write_all
tx_vodctrl [2:0]
rx_eqctrl [3:0]
reconfig_fromgxb [50:0] (1)
rx_tx_duplex_sel [1:0]
logical_channel_address [3:0]
ALTGX_RECONFIG Instance
reconfig_fromgxb [3:0]
data_valid
busy
Set the **What is the starting channel number?** option = 0
ALTGX Instance 1
(Number of Channels is 5)
reconfig_fromgxb [33:0]
Set the **What is the starting channel number?** option = 8
ALTGX Instance 2
(Number of Channels is 3)
reconfig_fromgxb [16:0]

Note to Figure 5–27:

(1) `reconfig_fromgxb [50:0]` = {`reconfig_fromgxb[33:0]`, `reconfig_fromgxb[16:0]`}

The following are the typical steps that help setup the dynamic reconfiguration process.

Five Channel Transceiver Instance (ALTGX Instance 1)

1. In the ALTGX MegaWizard Plug-In Manager, set the **What is the number of channels?** option in the **General** screen to **5**.
2. Enable the **Analog controls (VOD, Pre-emphasis, and Manual Equalization)** option in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager.
3. The `reconfig_fromgxb` output signal is transceiver block based, so the number of bits for this instance is 34. The `reconfig_togxb` input signal is a fixed bus (four bits wide).
4. Set the **What is the starting channel number?** option in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager to **0**. Refer to "Logical Channel Addressing" on page 5–14 for additional details.

Three Channel Transceiver Instance (ALTGX Instance 2)

1. In the ALTGX MegaWizard Plug-In Manager, set the **What is the number of channels?** option in the **General** screen to **3**.
2. Enable the **Analog controls (VOD, Pre-emphasis, and Manual Equalization)** option in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager.
3. The `reconfig_fromgxb` output signal is transceiver block based, so the number of bits for this instance is 17. The `reconfig_togxb` input signal is a fixed bus (four bits wide).
4. Set the **What is the starting channel number?** option in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager to **8**. Refer to "Logical Channel Addressing" on page 5–14 for additional details.

Dynamic Reconfiguration Controller Instance (ALTGX_RECONFIG Instance)

1. Launch the ALTGX_RECONFIG MegaWizard Plug-In Manager.
2. Set the **What is the number of channels controlled by the reconfig controller?** option in the **Reconfiguration settings** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager to **12**. Refer to "Total Number of Channels Controlled by the ALTGX_RECONFIG Instance" on page 5–25 for more information about this setting.
3. The `reconfig_fromgxb` input signal is transceiver block based, so the width of this signal is 51 (3×17) bits.
4. Select the **Use 'logical_channel_address' port** option in the **Analog controls** screen. The ALTGX_RECONFIG MegaWizard Plug-In Manager enables the `logical_channel_address [3:0]` input port.
5. Select the `rx_tx_duplex_sel [1:0]` port in the **Error checks/Data rate switch** screen. The ALTGX_RECONFIG MegaWizard Plug-In Manager enables the `rx_tx_duplex_sel [1:0]` input port.
6. Select the `tx_vodctrl` and `rx_eqctrl` controls. The `tx_vodctrl` is 3-bits wide and the `rx_eqctrl` is 4-bits wide.

ALTGX Instances and ALTGX_RECONFIG Instances Connections

1. Connect the `reconfig_fromgxb [33:0]` output port from ALTGX instance 1 to the `reconfig_fromgxb [33:0]` input port of the ALTGX_RECONFIG instance.
2. Similarly connect the `reconfig_fromgxb [16:0]` output port from ALTGX instance 2 to the `reconfig_fromgxb [50:34]` input port of the ALTGX_RECONFIG instance.
3. Connect the `reconfig_togxb [3:0]` output port of the ALTGX_RECONFIG instance to the `reconfig_togxb [3:0]` input ports of both the ALTGX instance 1 and ALTGX instance 2.
4. Refer to "Connecting reconfig_from_gxb/reconfig_to_gxb Ports" on page 5–28 for additional details.

Dynamically Reconfiguring the tx_vodctrl PMA Controls Using Method 1

1. Set the control port: `tx_vodctrl` to the desired setting. (For example: `tx_vodctrl` = 3'b000).
2. Set the `logical_channel_address [3:0]` to 4'b0000 (logical channel address of the first channel of the ALTGX instance 1).
3. Set the `rx_tx_duplex_sel [1:0]` = 2'b10 (to write to the transmitter portion of the first channel of the ALTGX instance 1).
4. Make sure that the `busy` signal is low, before you start a write transaction.
5. Assert the `write_all` signal for one `reconfig_clk` clock cycle. This initiates the write transaction.
6. The `busy` output status signal is asserted to show that the controller is busy writing the new values.
7. Once the write transaction has completed, the `busy` signal goes low.

   Refer to "Method 1" on page 5–39 for additional details.

Dynamically Reconfiguring the rx_eqctrl PMA Control Using Method 1

1. Set the control port: `rx_eqctrl` to the desired settings. (For example: `rx_eqctrl` = 4'b0000).
2. Set the `logical_channel_address [3:0]` to 4'b1001 (logical channel address of the second channel of the ALTGX instance 2).
3. Set the `rx_tx_duplex_sel [1:0]` = 2'b01 (to write to the receiver portion of the second channel of the ALTGX instance 2).
4. Make sure that the `busy` signal is low, before you start a write transaction.
5. Assert the `write_all` signal for one `reconfig_clk` clock cycle. This initiates the write transaction.
6. The `busy` output status signal is asserted to show that the controller is busy writing the new values.
7. Once the write transaction has completed, the `busy` signal goes low.

   Refer to "Method 1" on page 5–39 for additional details.

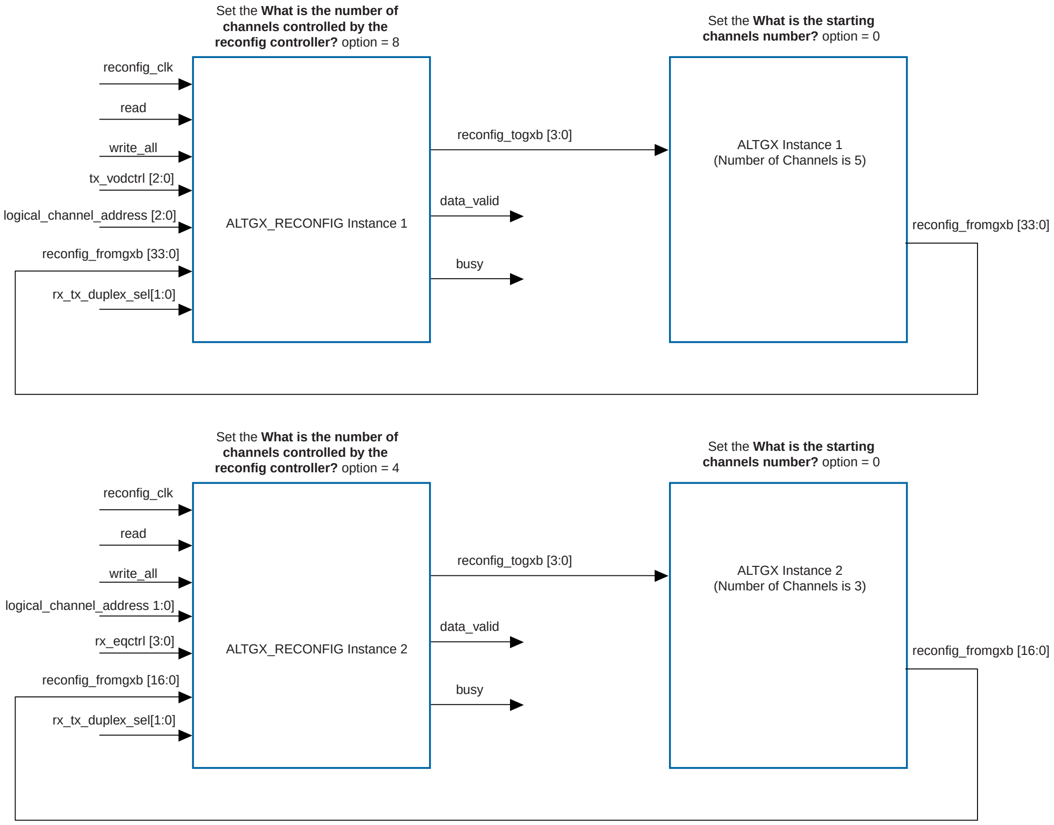
Example 2: Two ALTGX_RECONFIG Instances Connected to Two ALTGX Instances

This design example has two instances of distinct configurations: ALTGX instance 1 with five transceiver channels and ALTGX instance 2 with three channels.

This configuration requires separate dynamic reconfiguration controllers for the two instances. This scenario covers the case of multiple dynamic reconfiguration controllers controlling multiple instances of the ALTGX.

Figure 5–28 shows the ALTGX_RECONFIG instance 1 connected to the ALTGX instance 1; ALTGX_RECONFIG instance 2 connected to the ALTGX instance 2.

Figure 5–28. Example 2 for PMA Controls Reconfiguration


Set the **What is the number of channels controlled by the reconfig controller?** option = 8
Set the **What is the starting channels number?** option = 0
reconfig_clk
read
write_all
tx_vodctrl [2:0]
logical_channel_address [2:0]
reconfig_fromgxb [33:0]
rx_tx_duplex_sel[1:0]
ALTGX_RECONFIG Instance 1
reconfig_togxb [3:0]
data_valid
busy
ALTGX Instance 1
(Number of Channels is 5)
reconfig_fromgxb [33:0]
Set the **What is the number of channels controlled by the reconfig controller?** option = 4
Set the **What is the starting channels number?** option = 0
reconfig_clk
read
write_all
logical_channel_address 1:0]
rx_eqctrl [3:0]
reconfig_fromgxb [16:0]
rx_tx_duplex_sel[1:0]
ALTGX_RECONFIG Instance 2
reconfig_togxb [3:0]
data_valid
busy
ALTGX Instance 2
(Number of Channels is 3)
reconfig_fromgxb [16:0]

Assume that you want to reconfigure the transmit VOD PMA control of the second channel of the ALTGX instance 1 and the receive equalization PMA control of the third channel of ALTGX instance 2. The following are the typical steps to setup the configuration:

Five Channel Transceiver Instance 1 (ALTGX Instance 1)

1. In the ALTGX MegaWizard Plug-In Manager, set the **What is the number of channels?** option in the **General** screen to **5**.
2. Enable the **Analog controls (VOD, Pre-emphasis, and Manual Equalization)** option in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager.

3. The `reconfig_fromgxb` output signal is transceiver block based, so the number of bits for this instance is 34. The `reconfig_togxb` input signal is a fixed bus (four bits wide).
4. Set the **What is the starting channel number?** option in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager to **0**. Refer to "Logical Channel Addressing" on page 5–14 for additional details.

Dynamic Reconfiguration Controller Instance 1 (ALTGX_RECONFIG Instance 1)

1. Launch the ALTGX_RECONFIG MegaWizard Plug-In Manager.
2. Set the **What is the number of channels controlled by the reconfig controller?** option in the **Reconfiguration settings** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager to **8**. Refer to "Total Number of Channels Controlled by the ALTGX_RECONFIG Instance" on page 5–25 for more information about this setting. The `reconfig_fromgxb` input signal is transceiver block based, so the width of this signal is 34 (2× 17) bits. Select the **Use 'logical_channel_address' port** option in the **Analog controls** screen. The ALTGX_RECONFIG MegaWizard Plug-In Manager enables the `logical_channel_address [2:0]` input port. Select the `rx_tx_duplex_sel [1:0]` port in the **Error checks/Data rate switch** screen. The ALTGX_RECONFIG MegaWizard Plug-In Manager enables the `rx_tx_duplex_sel [1:0]` input port.
3. Select the `tx_vodctrl` control. The width of this signal is fixed to 3-bits wide.

Three Channel Transceiver Instance 2 (ALTGX instance 2)

1. In the ALTGX MegaWizard Plug-In Manager, set the **What is the number of channels?** option in the **General** screen to **3**.
2. Enable the **Analog controls (VOD, Pre-emphasis, and Manual Equalization)** option in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager.
3. The `reconfig_fromgxb` output signal is transceiver block based, so the number of bits for this instance is 17. The `reconfig_togxb` input signal is a fixed bus (four bits wide).
4. Set the **What is the starting channel number?** option in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager to **0**. You do not require the next multiple of 4 as the **What is the starting channel number?** option for this ALTGX instance 2 because these two ALTGX instances are controlled by different dynamic reconfiguration controllers. Refer to "Logical Channel Addressing" on page 5–14 for additional details.

Dynamic Reconfiguration Controller Instance 2 (ALTGX_RECONFIG Instance 2)

1. Launch the ALTGX_RECONFIG MegaWizard Plug-In Manager.
2. Set the **What is the number of channels controlled by the reconfig controller?** option in the **Reconfiguration settings** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager to **4**. Refer to "Total Number of Channels Controlled by the ALTGX_RECONFIG Instance" on page 5–25 for more information about this setting. The `reconfig_fromgxb` input signal is transceiver block based, so the width of this signal is 17 (1× 17) bits.

3. Select the **Use 'logical_channel_address' port** option in the **Analog controls** screen. The ALTGX_RECONFIG MegaWizard Plug-In Manager enables the `logical_channel_address [1:0]` input port.
4. Select the `rx_tx_duplex_sel [1:0]` port in the **Error checks/Data rate switch** screen. The ALTGX_RECONFIG MegaWizard Plug-In Manager enables the `rx_tx_duplex_sel [1:0]` input port.
5. Select the `rx_eqctrl` control port. It is 4-bits wide.

ALTGX Instances and ALTGX_RECONFIG Instances Connections

1. Connect the `reconfig_fromgxb` signal from each ALTGX instance to the same signal of the corresponding ALTGX_RECONFIG instance. Refer to Figure 5–28 for additional information.
2. Connect the `reconfig_togxb` signal from each ALTGX_RECONFIG instance to the same signal of the corresponding ALTGX instance.

Dynamically Reconfiguring the tx_vodctrl PMA Control of ALTGX Instance 1 from ALTGX_RECONFIG Instance 1 Using Method 1:

1. Set the PMA control port: `tx_vodctrl` to the desired setting. (For example: `tx_vodctrl` = 3'b000).
2. Set the `logical_channel_address[2:0]` = 3'b001 (logical channel address of the second channel of the ALTGX instance 1).
3. Set the `rx_tx_duplex_sel[1:0]` = 2'b10 (to write to the transmitter portion of the second channel of the ALTGX instance 1).
4. Make sure that the `busy` signal is low, before you start a write transaction.
5. Assert the `write_all` signal for one `reconfig_clk` clock cycle. This initiates the write transaction.
6. The `busy` output status signal is asserted to show that the controller is busy writing the new values.
7. When the write transaction is completed, the `busy` signal goes low.
8. Refer to "Method 1" on page 5–39 for additional details.

Dynamically Reconfiguring the rx_eqctrl PMA Control of ALTGX Instance 2 from ALTGX_RECONFIG Instance 2 Using Method 1:

1. Set the PMA control port: `rx_eqctrl` to the desired settings. (For example: `rx_eqctrl` = 4'b0000).
2. Set the `logical_channel_address[1:0]` = 2'b10 (logical channel address of the third channel of the ALTGX instance 2).
3. Set the `rx_tx_duplex_sel[1:0]` = 2'b01 (to write to the receiver portion of the third channel of the ALTGX instance 2).
4. Make sure that the `busy` signal is low, before you start a write transaction.
5. Assert the `write_all` signal for one `reconfig_clk` clock cycle. This initiates the write transaction.
6. The `busy` output status signal is asserted to show that the controller is busy writing the new values.

7. When the write transaction is completed, the `busy` signal goes low.
8. Refer to "Method 1" on page 5–39 for additional details.

Example 3: One ALTGX_RECONFIG Instance Connected to an ALTGX Instance Stamped Five Times

This design example consists of five channels of transceivers. This configuration has one dynamic reconfiguration controller to control five channels. This scenario covers the case stamping five instantiations of one channel ALTGX instance configuration.

ALTGX Instance with One Transceiver Channel

1. Set the **What is the number of channels?** option in the **General** screen of the ALTGX MegaWizard Plug-In Manager to **1**.
2. Enable the **Analog controls (VOD, Pre-emphasis, and Manual Equalization)** option in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager.
3. The `reconfig_fromgxb` output signal is transceiver block based so the number of bits for this instance is 17. This is because the number of channels is one and it can logically fit into a single transceiver block. The `reconfig_togxb` input signal is a fixed bus (four bits).
4. Set the option **What is the starting channel number**? in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager to **0**. Refer to "Logical Channel Addressing" on page 5–14 for additional information.
5. Click **Finish**.
6. Assume that the instantiation name is **instance1**.

Instantiating Five Transceiver Channels Using the Same ALTGX Instance

When you stamp instance 1 five times, the **What is the starting channel number?** options of the other four stamped instances (assume instance2, instance3, instance4, instance5, instance6) are **4**, **8**, **12**, and **16** respectively. Refer to "Logical Channel Addressing" on page 5–14 for additional information.

Dynamic Reconfiguration Controller Instance (ALTGX_RECONFIG Instance)

1. Launch the ALTGX_RECONFIG MegaWizard Plug-In Manager.
2. Set the **What is the number of channels controlled by the reconfig controller?** option in the **Reconfiguration settings** screen of the ALTGX_RECONFIG MegaWizard Plug-In Manager to **20** . This enables five sets of MegaWizard Plug-In Manager signals. (`reconfig_fromgxb[84:0]`). Connect each of the stamped ALTGX instance to one set of MegaWizard Plug-In Manager signals.
3. Select the necessary write and read controls to write in and read out from the VOD, pre-emphasis, equalization, and DC gain options. For example, if you select the VOD setting: the `tx_vodctrl` signal is 60-bits wide (3-bits per channel). The `tx_vodctrl [2:0]` corresponds to the single channel of the first stamped instance. The bits `tx_vodctrl [11:3]` are not used because they correspond to the unused channels in the first stamped instance with logical channel addresses 1 to 3. Similarly, `tx_vodctrl [14:12]` corresponds to the single channel of the second stamped instance, and so on.

ALTGX Instances and ALTGX_RECONFIG Instance Connections

1. Connect the `reconfig_fromgxb` signal from each ALTGX instance to the same signal in the ALTGX_RECONFIG instance. You must connect it such that a way that the `reconfig_fromgxb` output port of the first ALTGX instance (ALTGX instance with the **What is the starting channel number?** option of 0) is connected to the LSB of the `reconfig_fromgxb` input port of the ALTGX_RECONFIG instance, and so on.
2. Connect the `reconfig_togxb` signal from the ALTGX_RECONFIG instance to the same signal in each of the ALTGX instances.

Dynamically Reconfiguring the tx_vodctrl of Instance 1 Using Method 2

1. Set the `tx_vodctrl` port to the desired setting. For example, if you want to write a VOD value of 2, set the `tx_vodctrl [2:0]` port to 3'b010.
2. Refer to "Method 1" on page 5–39 for additional details.

☞ When you perform a write transaction using Method 2, the values on the PMA control ports are written on all transceiver channels connected to the dynamic reconfiguration controller. Therefore, ensure that you also have the desired values on the `tx_vodctrl [59:3]`. If you want to make sure that the VOD settings of the remaining channels are not affected, you can optionally perform a read transaction, and obtain the existing values and write back the same values.

Error Indication in the ALTGX_RECONFIG MegaWizard Plug-In Manager

The ALTGX_RECONFIG MegaWizard Plug-In Manager provides an `error` status signal when you select the **Enable illegal mode checking** option or the **Enable self recovery** option in the **Error checks/data rate switch** screen. The conditions under which the `error` signal is asserted are:

- **Enable illegal mode checking** option

  When you select this option, the dynamic reconfiguration controller checks whether an attempted operation falls under one of the conditions listed below. The dynamic reconfiguration controller detects these conditions within two `reconfig_clk` cycles, de-asserts the `busy` signal, and asserts the `error` signal for two `reconfig_clk` cycles.

  - PMA controls - read operation:
    - None of the output ports (`rx_eqctrl_out`, `rx_eqdcgain_out`, `tx_vodctrl_out`, `tx_preemp_0t_out`, `tx_preemp_1t_out`, and `tx_preemp_2t_out`) are selected in the ALTGX_RECONFIG instance.

      and
    - `read` signal is asserted
  - PMA controls - write operation:
    - None of the input ports (`rx_eqctrl`, `rx_eqdcgain`, `tx_vodctrl`, `tx_preemp_0t`, `tx_preemp_1t`, and `tx_preemp_2t`) are selected in the ALTGX_RECONFIG instance.

      and
    - `write_all` signal is asserted.
- **Enable self recovery** option

  When you select this option, the controller automatically recovers if the operation did not complete within the expected time. The `error` signal is driven high whenever the controller performs a self recovery.

Combining Transceiver Channels with Dynamic Reconfiguration Enabled

Packing the transceiver channels into the same physical transceiver block is called “combining”. You can combine the transceiver channels in a design into the same physical transceiver block by assigning the `tx_dataout` and `rx_datain` pins of the channels to the same transceiver block. By default, the software automatically packs the transceiver channels into the same physical transceiver block based on certain requirements which are discussed below.

The Quartus II software also allows you to combine multiple channels into the same physical transceiver block based on the same requirements discussed below.

Requirements

When dynamic reconfiguration is enabled, the Quartus II software has certain requirements for combining multiple transceiver channels in the same physical transceiver block:

- All the channels that you want to combine into the same transceiver block should have the same options enabled in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager. When you enable **Analog controls (VOD, Pre-emphasis, and Manual Equalization)** option in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager for a channel, you should enable the same option for all the other channels to be combined.
- All the channels must be controlled by the same ALTGX_RECONFIG (dynamic reconfiguration controller) instance. The transceiver channels connected to multiple ALTGX_RECONFIG instances cannot be combined into the same physical transceiver block, even if they are configured to the same functional mode and data rate.

Combining a **Transmitter only** instance and **Receiver only** instance.

Consider that you want to combine one **Receiver only** instance and another **Transmitter only** instance in the same transceiver block:

- The **Receiver only** instance must be controlled by an ALTGX_RECONFIG instance for offset cancellation control.
- Because you want to combine the **Recevier only** instance with another **Transmitter only** instance into the same transceiver block, you must control the **Transmitter only** instance using the same ALTGX_RECONFIG instance.
- Therefore you must enable the **Analog controls** option in the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager for both the **Transmitter only** and **Receiver only** instances.

☞ There are constraints with the independent **Transmitter only** and independent **Receiver only** configurations. Both transmitter and receiver have to go through a reset sequence, even if the transmitter or receiver is reconfigured.

Dynamic Reconfiguration Duration and FPGA-Fabric Resource Utilization

This section describes the time taken for dynamic reconfiguration transactions and FPGA-fabric resources used by the dynamic reconfiguration controller when used in different modes of reconfiguration.

Dynamic Reconfiguration Duration

Dynamic reconfiguration duration is the number of cycles for which the `busy` signal is asserted when the dynamic reconfiguration controller performs write transactions, read transactions or offset cancellation of receiver channels.

PMA Controls Reconfiguration Duration

The following section gives an estimate of the number of `reconfig_clk` clock cycles for which the `busy` signal is asserted during the PMA controls reconfiguration using Method 1 and Method 2. Refer to "Dynamically Reconfiguring PMA Controls" on page 5–38 for additional details about these two methods.

PMA Controls Reconfiguration Duration When Using Method 1

The `logical_channel_address` port is used in this method. The write transaction and read transaction duration is as follows:

Write Transaction Duration

For writing values to the following PMA controls, the `busy` signal is asserted for 260 `reconfig_clk` clock cycles for each of these controls.

- `tx_preemp_1t` (pre-emphasis control first post-tap)
- `tx_vodctrl` (voltage output differential)
- `rx_eqctrl` (equalizer control)
- `rx_eqdcgain` (equalizer DC gain)

For writing values to the following PMA controls, the `busy` signal is asserted for 520 `reconfig_clk` clock cycles for each of these controls.

- `tx_preemp_0t` (pre-emphasis control pre-tap)
- `tx_preemp_2t` (pre-emphasis control second post-tap)

Read Transaction Duration

For reading the existing values of the following PMA controls, the `busy` signal is asserted for 130 `reconfig_clk` clock cycles for each of these controls. The `data_valid` signal is then asserted once `busy` signal goes low.

- `tx_preemp_1t_out` (pre-emphasis control first post-tap)
- `tx_vodctrl_out` (voltage output differential)
- `rx_eqctrl_out` (equalizer control)
- `rx_eqdcgain_out` (equalizer DC gain)

For reading the existing values of the following PMA controls, the `busy` signal is asserted for 260 `reconfig_clk` clock cycles for each of these controls. The `data_valid` signal is then asserted once `busy` signal goes low.

- `tx_preemp_0t_out` (pre-emphasis control pre-tap)
- `tx_preemp_2t_out` (pre-emphasis control second post-tap)

PMA Controls Reconfiguration Duration When Using Method 2

The `logical_channel_address` port is NOT used in this method. The write transaction duration and read transaction duration is as follows:

Write Transaction Duration

For writing values to the following PMA controls, the `busy` signal is asserted for 260 `reconfig_clk` clock cycles **per channel for each of these controls.**

- `tx_preemp_1t` (pre-emphasis control first post-tap)
- `tx_vodctrl` (voltage output differential)
- `rx_eqctrl` (equalizer control)
- `rx_eqdcgain` (equalizer DC gain)

For writing values to the following PMA controls, the `busy` signal is asserted for 520 `reconfig_clk` clock cycles **per channel for each of these controls**.

- `tx_preemp_0t` (pre-emphasis control pre-tap)
- `tx_preemp_2t` (pre-emphasis control second post-tap)

Read Transaction Duration

For reading the existing values of the following PMA controls, the `busy` signal is asserted for 130 `reconfig_clk` clock cycles **per channel for each of these controls**. The `data_valid` signal is then asserted once `busy` signal goes low.

- `tx_preemp_1t_out` (pre-emphasis control first post-tap)
- `tx_vodctrl_out` (voltage output differential)
- `rx_eqctrl_out` (equalizer control)
- `rx_eqdcgain_out` (equalizer DC gain)

For reading the existing values of the following PMA controls, the `busy` signal is asserted for 260 `reconfig_clk` clock cycles **per channel for each of these controls**. The `data_valid` signal is then asserted once `busy` signal goes low.

- `tx_preemp_0t_out` (pre-emphasis control pre-tap)
- `tx_preemp_2t_out` (pre-emphasis control second post-tap)

Offset Cancellation Duration

When the device powers up, the `busy` signal remains low for the first `reconfig_clk` clock cycle. After the device powers up, it takes 70 `reconfig_clk` clock cycles for the dynamic reconfiguration controller to identify the receiver channels.

When it identifies the receiver channels, the dynamic reconfiguration controller takes another 2600 `reconfig_clk` clock cycles to perform the offset cancellation process. In other words the `busy` signal goes low after 2670 (70 + 2600) `reconfig_clk` clock cycles per receiver channel.

☞ If the design does not require PMA controls reconfiguration, each ALTGX instance in the design can have its own dynamic reconfiguration controller (ALTGX_RECONFIG instance). This minimizes the offset cancellation duration.

Dynamic Reconfiguration (ALTGX_RECONFIG Instance) Resource Utilization

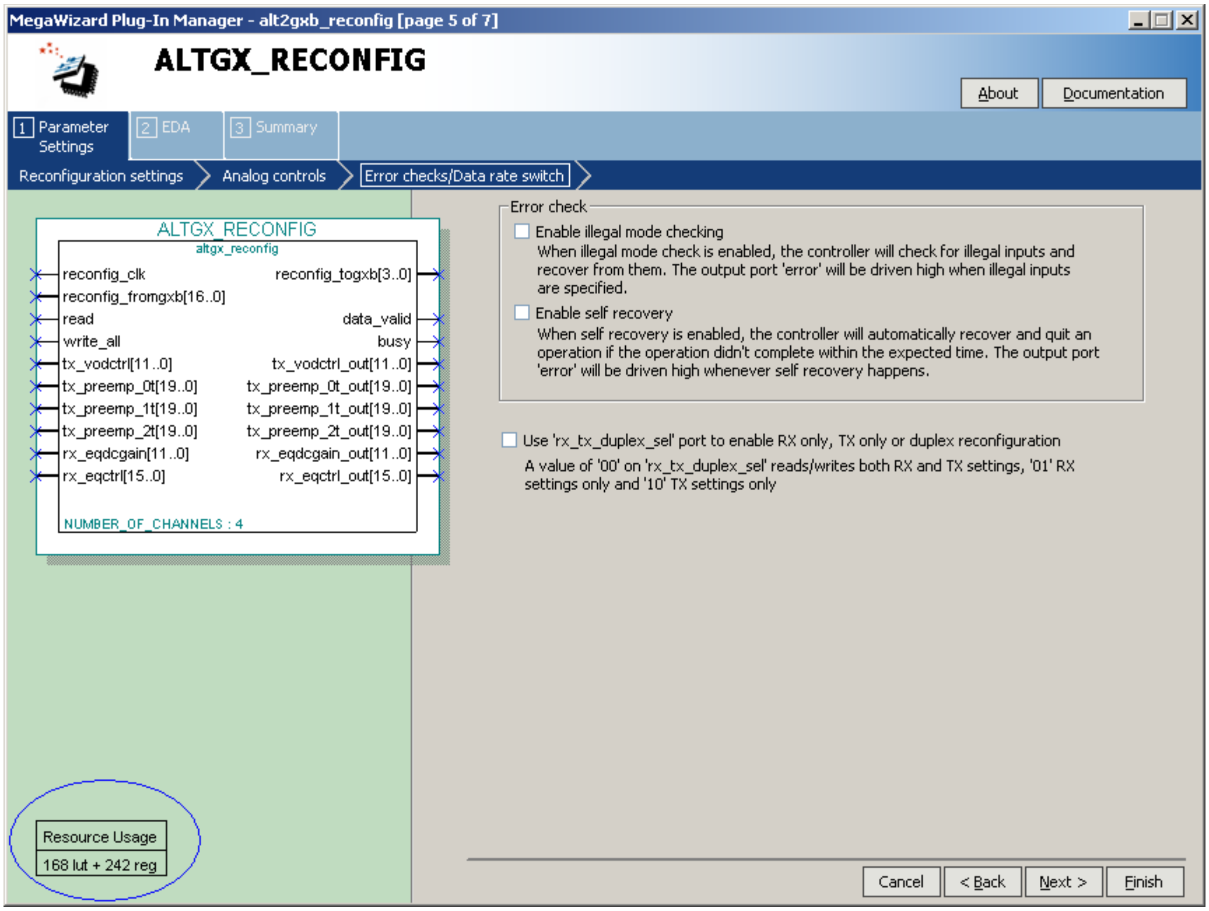
You can observe the resources utilized during dynamic reconfiguration, in the ALTGX_RECONFIG MegaWizard Plug-In Manager itself. The following sections give an estimate of the logic elements (LE) resources utilized during dynamic reconfiguration.

You can obtain the resource utilization for all other PMA controls from the ALTGX_RECONFIG MegaWizard Plug-In Manager.

For example, the number of LEs used by one dynamic reconfiguration controller is 43 with only `tx_vodctrl` selected. Similarly, the number of registers is 130.

Figure 5–29 shows the resource utilization in the ALTGX_RECONFIG MegaWizard Plug-In Manager.

Figure 5–29. Resource Utilization in the ALTGX_RECONFIG MegaWizard Plug-In Manager



Error check

□ Enable illegal mode checking

When illegal mode check is enabled, the controller will check for illegal inputs and recover from them. The output port 'error' will be driven high when illegal inputs are specified.

□ Enable self recovery

When self recovery is enabled, the controller will automatically recover and quit an operation if the operation didn't complete within the expected time. The output port 'error' will be driven high whenever self recovery happens.

□ Use 'rx_tx_duplex_sel' port to enable RX only, TX only or duplex reconfiguration

A value of '00' on 'rx_tx_duplex_sel' reads/writes both RX and TX settings, '01' RX settings only and '10' TX settings only

Resource Usage
168 lut + 242 reg

Functional Simulation of the Offset Cancellation Process

This section lists the points to be considered during the functional simulation of the dynamic reconfiguration process.

You must connect the ALTGX_RECONFIG instance to the ALTGX_instance/ALTGX instances in your design for functional simulation. The functional simulation uses a reduced timing model of the dynamic reconfiguration controller. Therefore, the duration of the dynamic reconfiguration process is 16 `reconfig_clk` clock cycles for functional simulation only. The `gxb_powerdown` signal must not be asserted during the offset cancellation sequence (for functional simulation and silicon).

Document Revision History

Table 5–6 shows the revision history for this document.

Table 5–6. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v1.0	Initial release	—

Stratix IV Device Handbook, Volume 3

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

SIV5V3-2.0

Contents

Chapter Revision Dates vii

Additional Information

About this Handbook . . . . Info-ix
How to Contact Altera . . . . Info-ix
Typographic Conventions . . . . Info-ix

Section I. Transceiver Configuration Guide

Revision History I-1

Chapter 1. ALTGX Transceiver Setup Guide

Introduction 1-1
Basic Mode 1-2
General Screen for Basic Mode 1-3
PLL/Ports Screen for Basic Mode 1-6
Ports/Cal Blk Screen for Basic Mode 1-9
RX Analog Screen for Basic Mode 1-11
TX Analog Screen for Basic Mode 1-13
Reconfig Screen for Basic Mode 1-15
Lpbk Screen for Basic Mode 1-17
Basic/8B10B Screen 1-19
Word Aligner Screen for Basic Mode 1-22
Rate Match/Byte Order Screen for Basic Mode 1-27
EDA Screen for Basic Mode 1-30
Summary for Basic Mode 1-31
Physical Interface for PCI-Express (PIPE) Mode 1-31
General Screen for PCI Express Mode 1-32
PLL/Ports Screen for PCI Express Mode 1-34
Ports/Cal Blk Screen for PCI Express Mode 1-37
RX Analog Screen for PCI Express Mode 1-39
TX Analog Screen for PCI Express Mode 1-42
Reconfig Screen for PCI Express Mode 1-43
Lpbk Screen for PCI Express Mode 1-44
PCI Express (PIPE) 1 Screen 1-45
PCI Express (PIPE) 2 Screen 1-49
EDA Screen for PCI Express Mode 1-52
Summary for PCI Express Mode 1-52

Serial RapidIO Mode . . . 1-52
- General Screen for Serial RapidIO Mode . . . 1-52
- PLL/Ports for Serial RapidIO Mode . . . 1-55
- Ports/Cal Blk Screen for Serial RapidIO Mode . . . 1-58
- RX Analog Screen for Serial RapidIO Mode . . . 1-59
- TX Analog Screen for Serial RapidIO Mode . . . 1-60
- Reconfig Screen for Serial RapidIO Mode . . . 1-60
- Lpbk Screen for Serial RapidIO Mode . . . 1-60
- Serial RapidIO/8B10B Screen for Serial RapidIO Mode . . . 1-61
- Word Aligner Screen for Serial RapidIO Mode . . . 1-65
- Rate Match/Byte Order Screen for Serial RapidIO Mode . . . 1-68
- EDA Screen for Serial RapidIO Mode . . . 1-71
- Summary for Serial RapidIO Mode . . . 1-71

(OIF) CEI PHY Interface Mode . . . 1-71
- General Screen for (OIF) CEI PHY Interface Mode . . . 1-71
- PLL/Ports Screen for (OIF) CEI PHY Interface Mode . . . 1-75
- Ports/Cal Blk Screen for (OIF) CEI PHY Interface Mode . . . 1-78
- RX Analog Screen for (OIF) CEI PHY Interface Mode . . . 1-79
- TX Analog Screen for (OIF) CEI PHY Interface Mode . . . 1-80
- Reconfig Screen for (OIF) CEI PHY Interface Mode . . . 1-80
- Lpbk Screen for (OIF) CEI PHY Interface Protocol Selection . . . 1-80
- CEI Screen . . . 1-82
- EDA Screen for (OIF) CEI PHY Interface Mode . . . 1-83
- Summary for (OIF) CEI PHY Interface Mode . . . 1-83

SDI Mode . . . 1-83
- General Screen for SDI Mode . . . 1-83
- PLL/Ports Screen for SDI Mode . . . 1-86
- Ports/Cal Blk Screen for SDI Mode . . . 1-89
- RX Analog Screen for SDI Mode . . . 1-91
- TX Analog Screen for SDI Mode . . . 1-91
- Reconfig Screen for SDI Mode . . . 1-91
- Lpbk Screen for SDI Mode . . . 1-91
- SDI/8B10B Screen for SDI Mode . . . 1-93
- Word Aligner Screen for SDI Mode . . . 1-96
- EDA Screen for SDI Mode . . . 1-98
- Summary for SDI Mode . . . 1-98

XAUI Mode . . . 1-98
- General Screen for XAUI Mode . . . 1-99
- PLL/Ports Screen for XAUI Mode . . . 1-101
- Ports/Cal Blk Screen for XAUI Mode . . . 1-104
- RX Analog Screen for XAUI Mode . . . 1-105
- TX Analog Screen for XAUI Mode . . . 1-106
- Reconfig Screen for XAUI Mode . . . 1-106
- Lpbk Screen for XAUI Mode . . . 1-106
- XAUI Screen . . . 1-107
- EDA Screen for XAUI Mode . . . 1-111
- Summary for XAUI Mode . . . 1-111

GIGE Mode 1-111
General Screen for GIGE Mode 1-112
PLL/Ports Screen for GIGE Mode 1-114
Ports/Cal Blk Screen for GIGE Mode 1-117
RX Analog Screen for GIGE Mode 1-119
TX Analog Screen for GIGE Mode 1-119
Reconfig Screen for GIGE Mode 1-119
Lpbk Screen for GIGE Mode 1-119
GIGE Screen 1-122
EDA Screen for GIGE Mode 1-125
Summary for GIGE Mode 1-125
SONET/SDH Mode 1-125
General Screen for SONET/SDH Mode 1-125
PLL/Ports Screen for SONET/SDH Mode 1-128
Ports/Cal Blk Screen for SONET/SDH Mode 1-131
RX Analog Screen for SONET/SDH Mode 1-133
TX Analog Screen for SONET/SDH Mode 1-133
Reconfig Screen for SONET/SDH Mode 1-133
Lpbk Screen for SONET/SDH Mode 1-133
SONET/SDH Screen for SONET/SDH Mode 1-135
EDA Screen for SONET/SDH Mode 1-138
Summary for SONET/SDH Mode 1-138
Referenced Documents 1-139
Document Revision History 1-139

Chapter 2. Transceiver Design Flow Guide

Overview 2-1
Architecture 2-3
Device Specification 2-3
Transceiver Configuration 2-3
Dynamic Reconfiguration 2-4
Clocking 2-5
Implementation and Integration 2-6
Create Transceiver Instances 2-6
Create Dynamic Reconfiguration Controller Instances 2-7
Create Reset Control Logic 2-7
Create Data Processing and Other User Logic 2-8
PPM Detector When Receiver CDR Is Used in Manual Lock Mode 2-8
Sync State Machine in Manual Word Alignment Mode 2-8
Compilation 2-8
Report Files 2-9
Fitter Summary 2-9
Pin-Out File 2-9
Resource Section 2-10
Verification 2-10
Simulation 2-10
Functional Simulation 2-10
Example 1 – Fibre Channel Protocol Application 2-12
Architecture 2-13
Implementation 2-16
Compilation 2-32
Simulating the Design 2-32
Documents Referenced 2-32
Document Revision History 2-32

Chapter 3. Porting a Stratix II GX Transceiver Design to a Stratix IV GX Device

Introduction 3-1
Quartus II Software Migration Guide 3-1
GXB Behavior Differences Tab 3-2
GXB Assignment Changes Needed Tab 3-2
GXB Parameter Changes Needed Tab 3-3
GXB Port Changes Needed Tab 3-4
Targeting the Stratix II GX Design for a Stratix IV GX Device 3-5
Opening ALT2GXB Instances in the ALTGX MegaWizard Plug-In Manager 3-6
Document Revision History 3-7

Chapter 4. Stratix IV ALTGX_RECONFIG Megafunction User Guide

Introduction 4-1
Dynamic Reconfiguration 4-1
Referenced Documents 4-10
Document Revision History 4-10

Chapter Revision Dates

The chapters in this book, *Stratix IV Device Handbook, Volume 3*, were revised on the following dates. Where chapters or groups of chapters are available separately, part numbers are listed.

Chapter 1 ALTGX Transceiver Setup Guide
Revised: *November 2008*
Part Number: *SIV53001-2.0*

Chapter 2 Transceiver Design Flow Guide
Revised: *November 2008*
Part Number: *SIV53002-2.0*

Chapter 3 Porting a Stratix II GX Transceiver Design to a Stratix IV GX Device
Revised: *November 2008*
Part Number: *SIV53003-1.0*

Chapter 4 Stratix IV ALTGX_RECONFIG Megafunction User Guide
Revised: *November 2008*
Part Number: *SIV53004-2.0*

Additional Information

About this Handbook

This handbook provides comprehensive information about the Altera® Stratix® IV family of devices.

How to Contact Altera

For the most up-to-date information about Altera products, see the following table.

Contact *(Note 1)*	Contact Method	Address
Technical support	Website	www.altera.com/support
Technical training	Website	www.altera.com/training
	Email	custrain@altera.com
Altera literature services	Email	literature@altera.com
Non-technical support (General)	Email	nacomp@altera.com
(Software Licensing)	Email	authorization@altera.com

Note:

(1) You can also contact your local Altera sales office or sales representative.

Typographic Conventions

The following table shows the typographic conventions that this document uses.

Visual Cue	Meaning
Bold Type with Initial Capital Letters	Indicates command names, dialog box titles, dialog box options, and other GUI labels. For example, **Save As** dialog box. For GUI elements, capitalization matches the GUI.
bold type	Indicates directory names, project names, disk drive names, file names, file name extensions, dialog box options, software utility names, and other GUI labels. For example, **\qdesigns** directory, **d:** drive, and **chiptrip.gdf** file.
Italic Type with Initial Capital Letters	Indicates document titles. For example, *AN 519: Stratix IV Design Guidelines*.
Italic type	Indicates variables. For example, *n* + 1. Variable names are enclosed in angle brackets (< >). For example, <*file name*> and <*project name*>**.pof** file.
Initial Capital Letters	Indicates keyboard keys and menu names. For example, Delete key and the Options menu.
“Subheading Title”	Quotation marks indicate references to sections within a document and titles of Quartus II Help topics. For example, “Typographic Conventions.”

Visual Cue	Meaning
`Courier type`	Indicates signal, port, register, bit, block, and primitive names. For example, `data1`, `tdi`, and `input`. Active-low signals are denoted by suffix `n`. For example, `resetn`. Indicates command line commands and anything that must be typed exactly as it appears. For example, `c:\qdesigns\tutorial\chiptrip.gdf`. Also indicates sections of an actual file, such as a Report File, references to parts of files (for example, the AHDL keyword `SUBDESIGN`), and logic function names (for example, `TRI`).
1., 2., 3., and a., b., c., and so on.	Numbered steps indicate a list of items when the sequence of the items is important, such as the steps listed in a procedure.
■ ■	Bullets indicate a list of items when the sequence of the items is not important.
☞	The hand points to information that requires special attention.
⚠ CAUTION	A caution calls attention to a condition or possible situation that can damage or destroy the product or your work.
⚠ WARNING	A warning calls attention to a condition or possible situation that can cause you injury.
↵	The angled arrow instructs you to press **Enter**.
👣	The feet direct you to more information about a particular topic.

Section I. Transceiver Configuration Guide

This section includes the following chapters:

- Chapter 1, ALTGX Transceiver Setup Guide
- Chapter 2, Transceiver Design Flow Guide
- Chapter 3, Porting a Stratix II GX Transceiver Design to a Stratix IV GX Device
- Chapter 4, Stratix IV ALTGX_RECONFIG Megafunction User Guide

Revision History

Refer to each chapter for its own specific revision history. For information on when each chapter was updated, refer to the Chapter Revision Dates section, which appears in the full handbook.

1. ALTGX Transceiver Setup Guide

SIV53001-2.0

Introduction

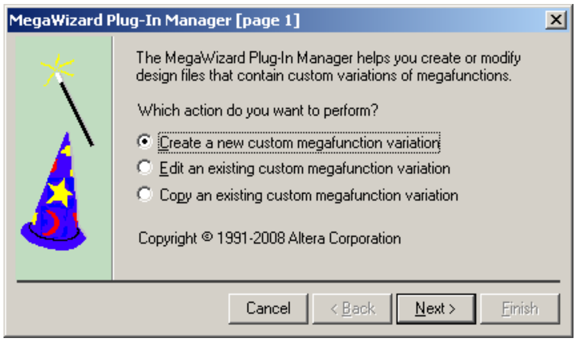
The MegaWizard® Plug-In Manager in the Quartus® II software creates or modifies design files that contain custom megafunction variations that can then be instantiated in a design file. The MegaWizard Plug-In Manager provides a MegaWizard that allows you to specify options for the ALTGX megafunction. You can use the MegaWizard Plug-In Manager to set the ALTGX megafunction features in the design. The ALTGX megafunction allows you to configure one or more transceiver channels. You can select the PCS and PMA functional blocks depending on your transceiver configuration.

Start the MegaWizard Plug-In Manager using one of the following methods:

- Select the **MegaWizard Plug-In Manager** from the Tools Menu.
- When working in the Block Editor, click **MegaWizard Plug-In Manager** in the **Symbol** dialog box (Edit menu).
- Start the stand-alone version of the MegaWizard Plug-In Manager by typing the following command at the command prompt: `qmegawiz`.

Figure 1–1 shows the first page of the MegaWizard Plug-In Manager. To generate an ALTGX custom megafunction variation, select **Create a new custom megafunction variation**.

Figure 1–1. *MegaWizard Plug-In Manager (Page 1)*



The MegaWizard Plug-In Manager helps you create or modify design files that contain custom variations of megafunctions.

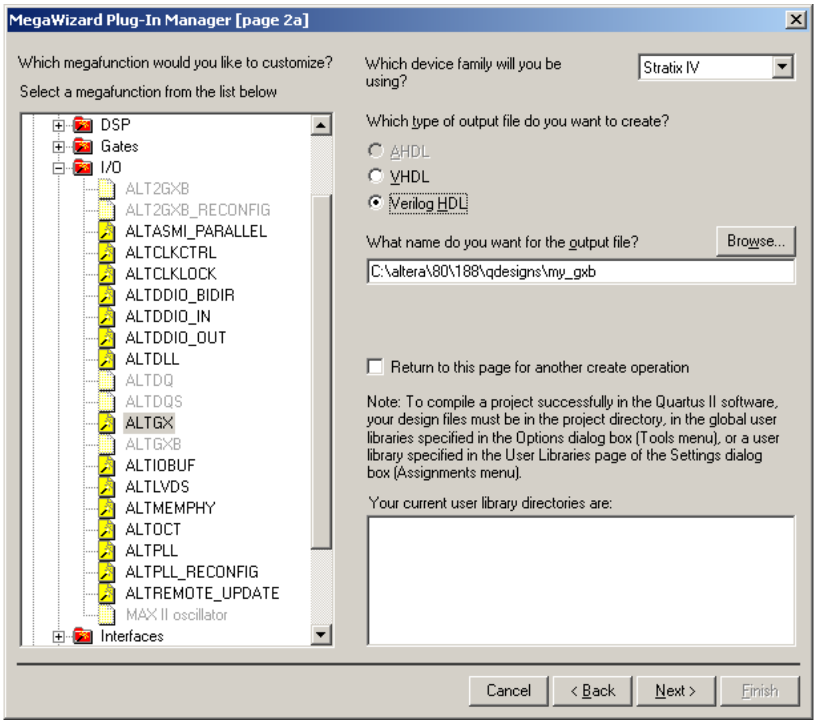
Which action do you want to perform?

● Create a new custom megafunction variation
○ Edit an existing custom megafunction variation
○ Copy an existing custom megafunction variation

Copyright © 1991-2008 Altera Corporation

Figure 1–2 shows the second page of the MegaWizard Plug-In Manager. Select the **Stratix IV** device as the device family. Select either **VHDL** or **Verilog HDL** depending on the type of output files you want to create. Select the **ALTGX** megafunction under the I/O section of the available megafunctions. After naming the output file, **Browse** to the folder you want to save your file in and click **Next**.

Figure 1–2. ***MegaWizard Plug-In Manager (Page 2)***



MegaWizard Plug-In Manager [page 2a]

Which megafunction would you like to customize?

Select a megafunction from the list below

Which device family will you be using? Stratix IV

Which type of output file do you want to create?

○ AHDL
○ VHDL
● Verilog HDL

What name do you want for the output file?

C:\altera\80\188\qdesigns\my_gxb

□ Return to this page for another create operation

Note: To compile a project successfully in the Quartus II software, your design files must be in the project directory, in the global user libraries specified in the Options dialog box (Tools menu), or a user library specified in the User Libraries page of the Settings dialog box (Assignments menu).

Your current user library directories are:

All reset and control signals are active high unless mentioned otherwise.

All output ports are synchronous to the data path unless specified otherwise.

The following section describes the steps involved in configuring the ALTGX megafunction in the Basic mode.

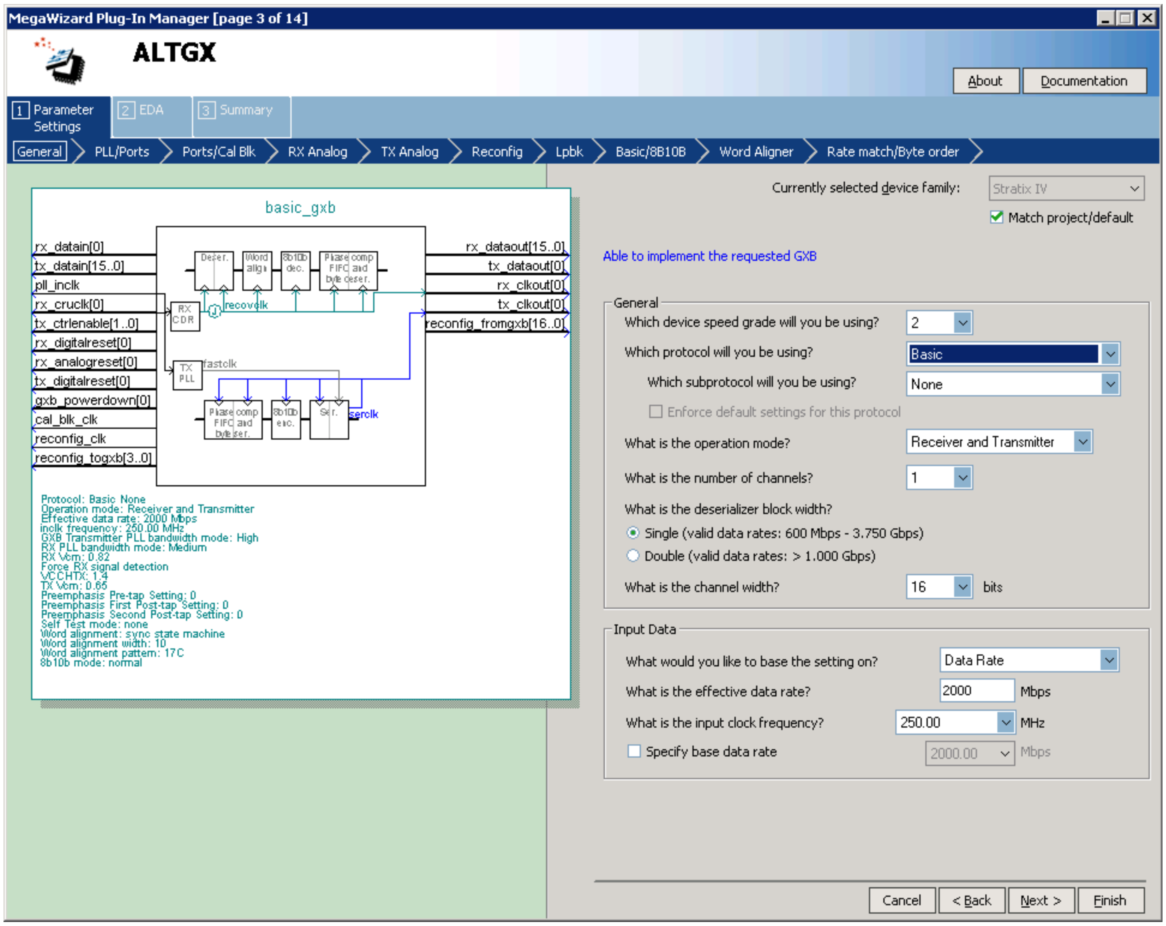
Basic Mode

This section provides descriptions of the options available on the individual pages of the ALTGX MegaWizard Plug-In Manager for Basic mode. The MegaWizard Plug-In Manager provides a warning if any of the settings you choose are illegal.

General Screen for Basic Mode

Figure 1–3 shows the **General** screen of the ALTGX MegaWizard Plug-In Manager for the Basic mode.

Figure 1–3. MegaWizard Plug-In Manager - ALTGX (General Screen for Basic Mode)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Currently selected device family: Stratix IV
☑ Match project/default

Able to implement the requested GXB

General
Which device speed grade will you be using? 2
Which protocol will you be using? Basic
Which subprotocol will you be using? None
☐ Enforce default settings for this protocol
What is the operation mode? Receiver and Transmitter
What is the number of channels? 1
What is the deserializer block width?
● Single (valid data rates: 600 Mbps - 3.750 Gbps)
○ Double (valid data rates: > 1.000 Gbps)
What is the channel width? 16 bits

Input Data
What would you like to base the setting on? Data Rate
What is the effective data rate? 2000 Mbps
What is the input clock frequency? 250.00 MHz
☐ Specify base data rate 2000.00 Mbps

Table 1–1 describes the available options on the **General** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–1. MegaWizard Plug-In Manager Options (General Screen for Basic Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Which device speed grade will you be using?	Select the speed grade of the device you will be using. The available speed grades are 2, 2x, 3, and 4. Based on the speed grade you select, the corresponding Stratix IV device can operate at the following maximum speeds: ■ -2 => 8.5 Gbps ■ -2x, 3 : 6.5 Gbps ■ -4 => 5 Gbps	Table 1-18 in the *DC and Switching Characteristics of the Stratix IV Device Family* in volume 5 of the *Stratix IV Device Handbook*
Which protocol will you be using?	Determines the specific protocol under which the transceiver operates. For the Basic mode, you must select the **Basic** protocol.	*Basic Functional Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Which subprotocol will you be using?	In Basic mode, the subprotocols are the diagnostic modes. The available options are as follows: ■ None – This is the normal operation of the transceiver. ■ X4 – This mode can be used to implement SFI-5 interface. In this mode, all four channels within the transceiver block are clocked from its central clock divider block to minimize the transmitter channel-to-channel skew. ■ X8 mode – In this mode, all eight channels in two transceiver blocks are clocked from the central clock divider of the master transceiver block to minimize the transmitter channel-to-channel skew.	*Basic Functional Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enforce default settings for this protocol.	This selection is not available in the Basic mode.	—
What is the operation mode?	The available operation modes are **Receiver only**, **Transmitter only**, and **Receiver and Transmitter**.	—
What is the number of channels?	The number of channels required with the same configuration. This option determines how many identical channels this ALTGX instance contains.	—
What is the deserializer block width?	This option sets the transceiver data path width. ■ **Single-width** This mode operates from 600 Mbps to 5 Gbps. ■ **Double-width** This mode operates from 1 Gbps to 8.5 Gbps.	*Basic Single-Width Mode Configurations* and *Basic Double-Width Mode Configurations* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the channel width?	This option determines the FPGA fabric - transceiver interface width. ■ In single-width mode, selecting **8** or **10** bits bypasses the byte serializer/deserializer. If you select **16** or **20** bits, the byte serializer/deserializer is used. ■ In double-width mode, selecting **16** or **20** bits bypasses the byte serializer/deserializer. If you select **32** or **40** bits, the byte serializer/deserializer is used.	*Byte Serializer* and *Byte Deserializer* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

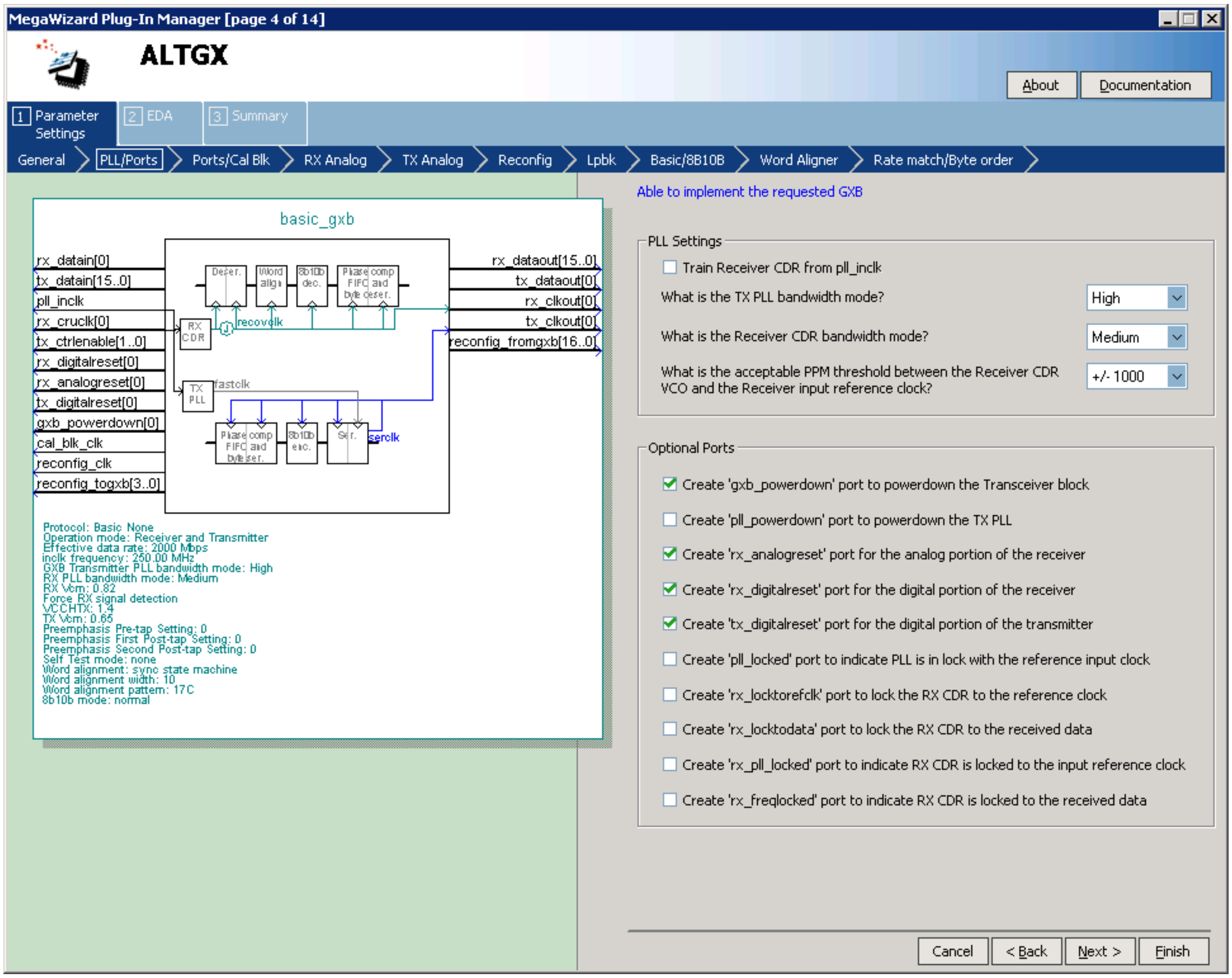
Table 1–1. MegaWizard Plug-In Manager Options (General Screen for Basic Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
What would you like to base the setting on?	You can select one of the following options: ■ Input clock frequency — Selecting this option allows you to enter your input clock frequency. Based on the value you enter, the Quartus II ALTGX MegaWizard Plug-In Manager populates the data rate options in the **What is the effective data rate?** field. The Quartus II ALTGX MegaWizard Plug-In Manager determines these data rate options depending on the available multipler settings. ■ Data rate — Selecting this option allows you to enter the transceiver channel serial data rate. Based on the value you enter, the Quartus II ALTGX MegaWizard Plug-In Manager populates the input reference clock frequency options in the **What is the input clock frequency?** field. The Quartus II ALTGX MegaWizard Plug-In Manager determines these input reference clock frequencies depending on the available multiplier settings.	—
What is the effective data rate?	■ If you select the **data rate** option in the **What would you like to base the setting on?** field, the ALTGX MegaWizard Plug-In Manager allows you to specify the effective serial data rate value in this field. ■ If you select the **input clock frequency** option in the **What would you like to base the setting on?** field, the ALTGX MegaWizard Plug-In Manager displays the list of effective serial data rates in this field.	—
What is the input clock frequency?	■ If you select the **input clock frequency** option in the **What would you like to base the setting on?** field, the ALTGX MegaWizard Plug-In Manager allows you to specify the input reference clock frequency in this field. ■ If you select the **data rate** option in the **What would you like to base the setting on?** field, the ALTGX MegaWizard Plug-In Manager displays the list of input reference clock frequencies in this field.	*CMU PLL and Receiver CDR Input Reference Clock* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*.
Specify base data rate.	The ALTGX MegaWizard Plug-In Manager provides you the base data rate options for the VCO in the CMU PLL and receiver CDR. If you select a value in this field that is greater than the value in the **What is the effective data rate?**, the ALTGX MegaWizard Plug-In Manager enables appropriate local clock divider values. The local divider is present in the transmitter and receiver channels.	—

PLL/Ports Screen for Basic Mode

Figure 1–4 shows the **PLL/Ports** screen of the ALTGX MegaWizard Plug-In Manager for Basic mode.

Figure 1–4. MegaWizard Plug-In Manager - ALTGX (PLL/Ports Screen)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

PLL Settings

□ Train Receiver CDR from pll_inclk

What is the TX PLL bandwidth mode? High

What is the Receiver CDR bandwidth mode? Medium

What is the acceptable PPM threshold between the Receiver CDR VCO and the Receiver input reference clock? +/- 1000

Optional Ports

☑ Create 'gxb_powerdown' port to powerdown the Transceiver block

□ Create 'pll_powerdown' port to powerdown the TX PLL

☑ Create 'rx_analogreset' port for the analog portion of the receiver

☑ Create 'rx_digitalreset' port for the digital portion of the receiver

☑ Create 'tx_digitalreset' port for the digital portion of the transmitter

□ Create 'pll_locked' port to indicate PLL is in lock with the reference input clock

□ Create 'rx_locktorefclk' port to lock the RX CDR to the reference clock

□ Create 'rx_locktodata' port to lock the RX CDR to the received data

□ Create 'rx_pll_locked' port to indicate RX CDR is locked to the input reference clock

□ Create 'rx_freqlocked' port to indicate RX CDR is locked to the received data

Table 1–2 describes the available options on the **PLL/ports** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–2. MegaWizard Plug-In Manager Options (PLL/Ports Screen for Basic Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Train Receiver CDR from `pll_inclk`.	If you select this option, the input reference clock to the CMU PLL trains the receiver CDR.	*Table 1-2* in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
What is the TX PLL bandwidth mode?	The available options are low, medium, and high. Select the appropriate option based on your system requirements.	*PLL Bandwidth Setting* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
What is the Receiver CDR bandwidth mode?	The available options are low, medium, and high. Select the appropriate option based on your system requirements.	—
What is the acceptable PPM threshold between the Receiver CDR VCO and the Receiver input reference clock?	In automatic lock mode, the CDR remains in LTD mode as long as the PPM difference between the CDR VCO output clock and the input reference clock is less than the PPM value that you set in this option. If the PPM difference is greater than the PPM value that you set in this option, the CDR switches to LTR mode. The range of values available in this option is ±62.5 ppm to ±1000 ppm. *(1)*	*Automatic Lock Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create a `gxb_powerdown` port to power down the Transceiver block.	This is an optional signal. When asserted, it powers down the entire transceiver block. If none of the channels are instantiated in a transceiver block, the Quartus II software automatically powers down the entire transceiver block.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create a `pll_powerdown` port to power down the TX PLL.	Each transceiver block has two CMU PLLs. Each CMU PLL has a dedicated power down signal called `pll_powerdown`. This signal powers down the CMU PLL.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create a `rx_analogreset` port for the analog portion of the receiver.	Receiver analog reset port available in **Receiver only** and **Receiver and Transmitter** operation modes. Resets part of the analog portion of the receiver CDR in the receiver channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create a `rx_digitalreset` port for the digital portion of the receiver.	Receiver digital reset port available in **Receiver only** and **Receiver and Transmitter** operation modes. Resets the PCS portion of the receiver channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook*.

Table 1–2. MegaWizard Plug-In Manager Options (PLL/Ports Screen for Basic Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
Create a `tx_digitalreset` port for the digital portion of the transmitter.	Transmitter digital reset port available in **Transmitter only** and **Receiver and Transmitter** operation modes. Resets the PCS portion of the transmitter channel. Altera® recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `pll_locked` port to indicate PLL is in lock with the input reference clock.	Each CMU PLL has a dedicated `pll_locked` signal that is fed to the FPGA fabric to indicate when the PLL is locked to the input reference clock.	*Transceiver Reset Sequences* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_locktorefclk` port to lock the RX CDR to the input reference clock.	This is an optional port. When this signal is asserted high, the LTR/LTD controller forces the receiver CDR to lock to the phase and frequency of the input reference clock. *(1), (2)*	*LTR/LTD Controller* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_locktodata` port to lock the RX CDR to the received data.	This is an optional port. When this signal is asserted high, the LTR/LTD controller forces the receiver CDR to lock to the received data. *(1), (2)*	*LTR/LTD Controller* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_pll_locked` port to indicate RX CDR is locked to the input reference clock.	In LTR mode, this signal is asserted high to indicate that the receiver CDR has locked to the phase and frequency of the input reference clock. In LTD mode, this signal has no significance. *(1)*	*Lock-to-Reference Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_freqlocked` port to indicate RX CDR is locked to the received data.	This signal gets asserted high to indicate that the receiver CDR has switched from LTR to LTD mode. This signal has relevance only in automatic lock mode and may be required to control the transceiver resets, as discussed in the *User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook. (1)*	*LTR/LTD Controller* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Notes to Table 1–2:

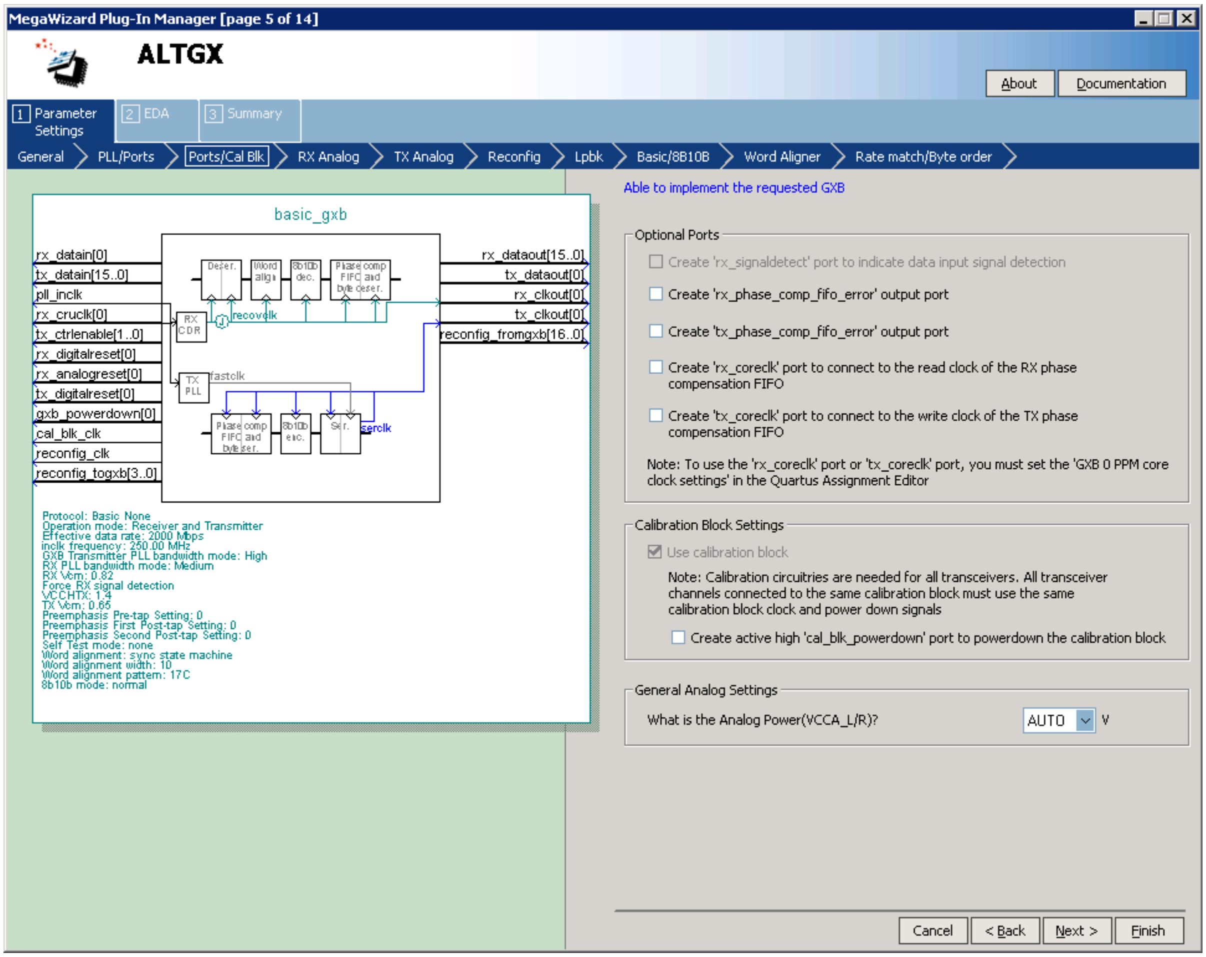
(1) LTR mode is lock-to-reference mode and LTD mode is lock-to-data mode.

(2) When `rx_locktorefclk` and `rx_locktodata` are both asserted high, `rx_locktodata` takes precedence over `rx_locktorefclk`, forcing the CDR to lock to the received data. When both these signals are de-asserted, the LTR/LTD controller is configured in automatic lock mode.

Ports/Cal Blk Screen for Basic Mode

Figure 1–5 shows the **Ports/Cal Blk** screen of the ALTGX MegaWizard Plug-In Manager for Basic mode.

Figure 1–5. MegaWizard Plug-In Manager - ALTGX (Ports/Cal Blk Screen for Basic Mode)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Optional Ports

□ Create 'rx_signaldetect' port to indicate data input signal detection

□ Create 'rx_phase_comp_fifo_error' output port

□ Create 'tx_phase_comp_fifo_error' output port

□ Create 'rx_coreclk' port to connect to the read clock of the RX phase compensation FIFO

□ Create 'tx_coreclk' port to connect to the write clock of the TX phase compensation FIFO

Note: To use the 'rx_coreclk' port or 'tx_coreclk' port, you must set the 'GXB 0 PPM core clock settings' in the Quartus Assignment Editor

Calibration Block Settings

☑ Use calibration block

Note: Calibration circuitries are needed for all transceivers. All transceiver channels connected to the same calibration block must use the same calibration block clock and power down signals

□ Create active high 'cal_blk_powerdown' port to powerdown the calibration block

General Analog Settings

What is the Analog Power(VCCA_L/R)? AUTO V

Table 1–3 describes the available options on the **Ports/Cal Blk** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–3. MegaWizard Plug-In Manager Options (Ports/Cal Blk Screen for Basic Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Create a `rx_signaldetect` port to indicate data input signal detection.	This port is not available in Basic mode.	*Signal Threshold Detection Circuitry* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create a `rx_phase_comp_fifo_error` output port.	This optional output port indicates Receiver Phase Compensation FIFO overflow or an underrun condition.	*Receiver Phase Compensation FIFO Error Flag* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

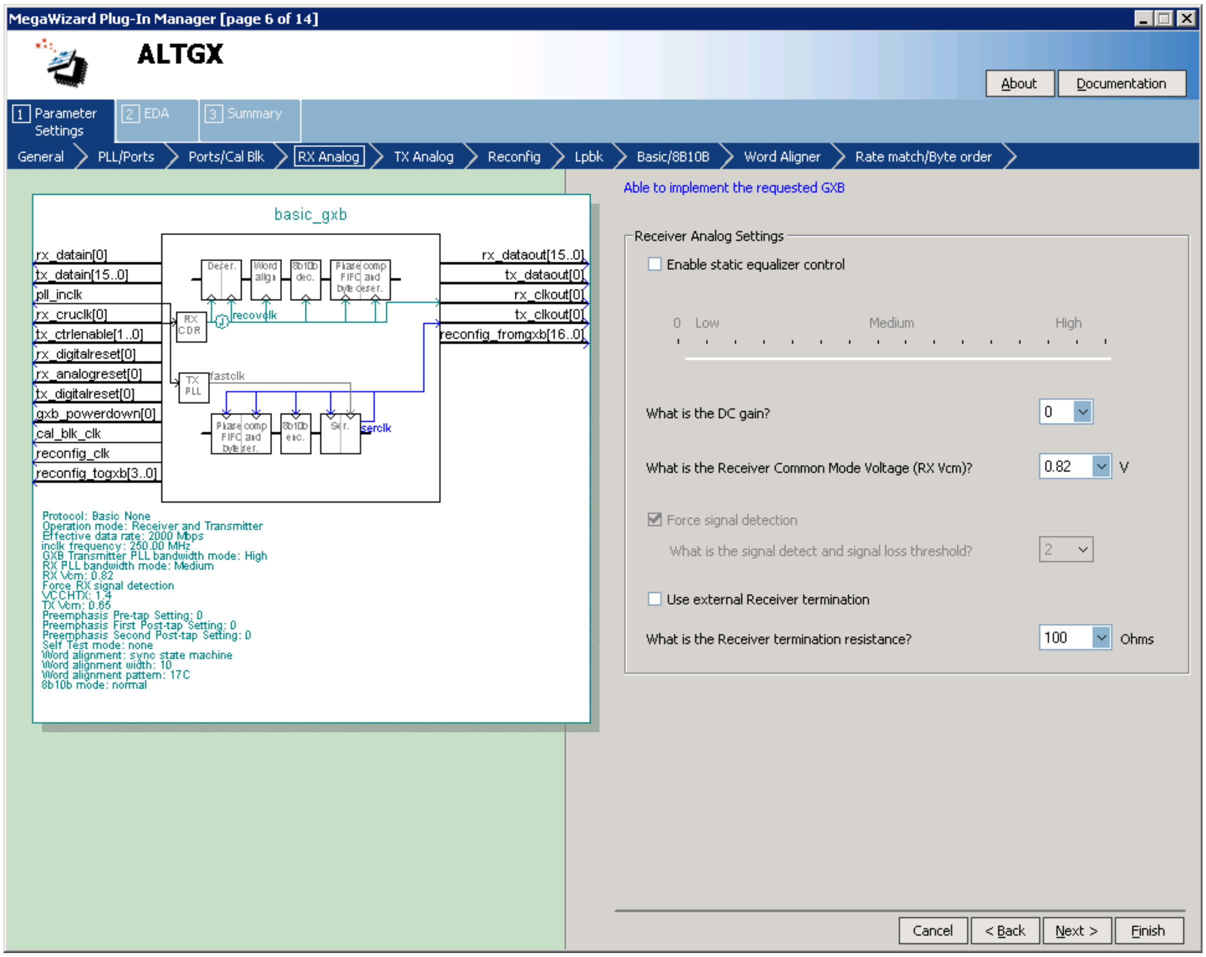
Table 1–3. MegaWizard Plug-In Manager Options (Ports/Cal Blk Screen for Basic Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
Create a `tx_phase_comp_fifo_error` output port.	This optional output port indicates Transmitter Phase Compensation FIFO overflow or an underrun condition.	*TX Phase Compensation FIFO Status Signal* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `rx_coreclk` port to connect to the read clock of the RX phase compensation FIFO.	The parallel output data from the receiver can be clocked using this optional input port. It allows you to clock the read side of the Receiver Phase Compensation FIFO with a user-provided clock (FPGA fabric clock, FPGA fabric-Transceiver interface clock, or input reference clock).	*FPGA Fabric-Transceiver Interface Clocking* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_coreclk` port to connect to the write clock of the TX phase compensation FIFO.	The parallel transmitter data generated in the FPGA fabric can be clocked using this optional input port. It allows you to clock the write side of the Transmitter Phase Compensation FIFO with a user-provided clock (FPGA fabric clock, FPGA fabric-Transceiver interface clock, or input reference clock).	*FPGA Fabric-Transceiver Interface Clocking* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Use calibration block.	Calibration block is always enabled.	*Calibration Blocks* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an active high `cal_blk_powerdown` to power down the calibration block.	Asserting this signal high powers down the calibration block. A high-to-low transition on this signal restarts calibration.	*Input Signals to the Calibration Block* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the Analog Power ($V_{CCA\_L/R}$)?	The options available for selection are 2.5V, 3.0V, and AUTO. ■ 3.0 V : up to 8.5 Gbps ■ 2.5 V : up to 4.25 Gbps ■ AUTO : The ALTGX MegaWizard Plug-In Manager automatically sets V_{CCA} to 2.5 V and V_{CCH} (transmitter buffer voltage) to 1.5 V for the VCO base data rates less than 4.25 Gbps. OR V_{CCA} to 3.0 V and V_{CCH} (transmitter buffer voltage) to 1.4 V for the VCO base data rates greater than 4.25 Gbps.	*General Requirements to Combine Channels* section in the *Configuring Multiple Protocols and Data Rates* chapter in volume 2 of the *Stratix IV Device Handbook.*

RX Analog Screen for Basic Mode

Figure 1–6 shows the **RX Analog** screen of the ALTGX MegaWizard Plug-In Manager for the Basic mode.

Figure 1–6. MegaWizard Plug-In Manager - ALTGX (RX Analog Screen)



```
Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal
```

Able to implement the requested GXB

Receiver Analog Settings

□ Enable static equalizer control

What is the DC gain? 0

What is the Receiver Common Mode Voltage (RX Vcm)? 0.82 V

☑ Force signal detection

What is the signal detect and signal loss threshold? 2

□ Use external Receiver termination

What is the Receiver termination resistance? 100 Ohms

Table 1–4 describes the available options on the **RX Analog** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

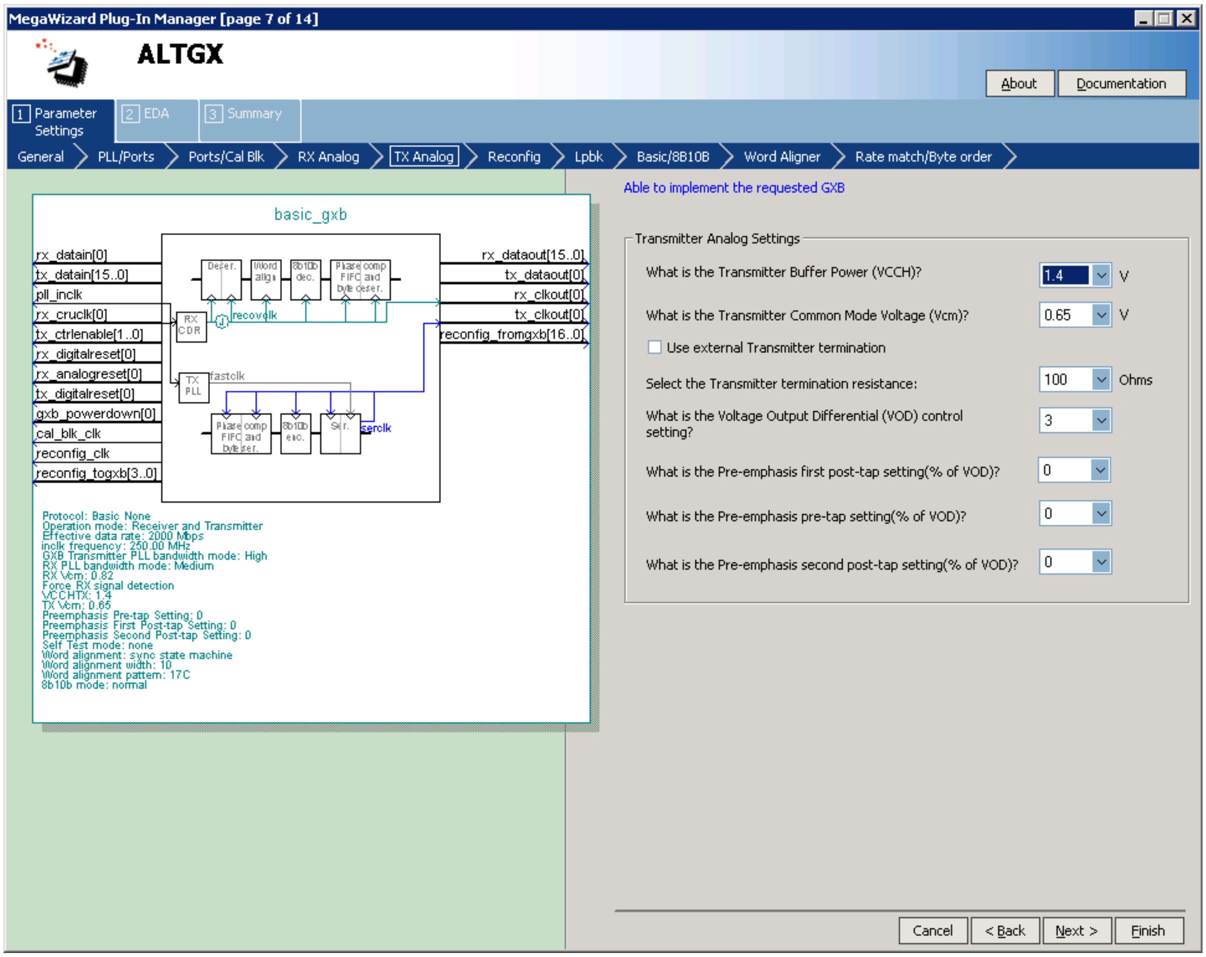
Table 1–4. MegaWizard Plug-In Manager Options (RX Analog Screen for Basic Mode)

ALTGX Setting	Description	Reference
Enable static equalizer control.	This option enables the static equalizer settings.	*Programmable Equalization and DC Gain* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
What is the DC gain?	This DC gain option has five settings: 0 – 0 dB 1 – 3 dB 2 – 6 dB 3 – 9 dB 4 – 12 dB	*Programmable Equalization and DC Gain* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
What is the receiver common mode voltage (RX V_{CM})?	The receiver common mode voltage is programmable to 0.82 V or 1.1 V.	*Programmable Common Mode Voltage* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Force signal detection.	This option is not available for selection in Basic mode. It is always enabled by default.	—
What is the signal detect and signal loss threshold?	This option is not available for selection in Basic mode.	—
Use external receiver termination.	This option is available if you want to use an external termination resistor instead of the differential on-chip termination (OCT). If checked, this option turns off the receiver OCT.	*Programmable Differential On-Chip Termination* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
What is the receiver termination resistance?	This option selects the receiver differential termination value. The settings allowed are 85 Ω, 100 Ω, 120 Ω, and 150 Ω.	*Programmable Differential On-Chip Termination* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

TX Analog Screen for Basic Mode

Figure 1–7 shows the **TX Analog** screen of the ALTGX MegaWizard Plug-In Manager for Basic mode.

Figure 1–7. MegaWizard Plug-In Manager - ALTGX (TX Analog Screen for Basic Mode)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Transmitter Analog Settings

What is the Transmitter Buffer Power (VCCH)? 1.4 V

What is the Transmitter Common Mode Voltage (Vcm)? 0.65 V

□ Use external Transmitter termination

Select the Transmitter termination resistance: 100 Ohms

What is the Voltage Output Differential (VOD) control setting? 3

What is the Pre-emphasis first post-tap setting(% of VOD)? 0

What is the Pre-emphasis pre-tap setting(% of VOD)? 0

What is the Pre-emphasis second post-tap setting(% of VOD)? 0

Table 1–5 describes the available options on the **TX Analog** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–5. MegaWizard Plug-In Manager Options (TX Analog Screen for Basic Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
What is the transmitter buffer power (V_{CCH})?	The available option is 1.4 V. It is up to the user to connect the correct voltage supply to the V_{CCH} pins on the board.	*Programmable Transmit Output Buffer Power* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
What is the transmitter common mode voltage (V_{CM})?	The available transmitter common mode voltage settings are 0.6 V, 0.65 V and 0.7 V.	*Common Mode Voltage (V_{CM}) Settings* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

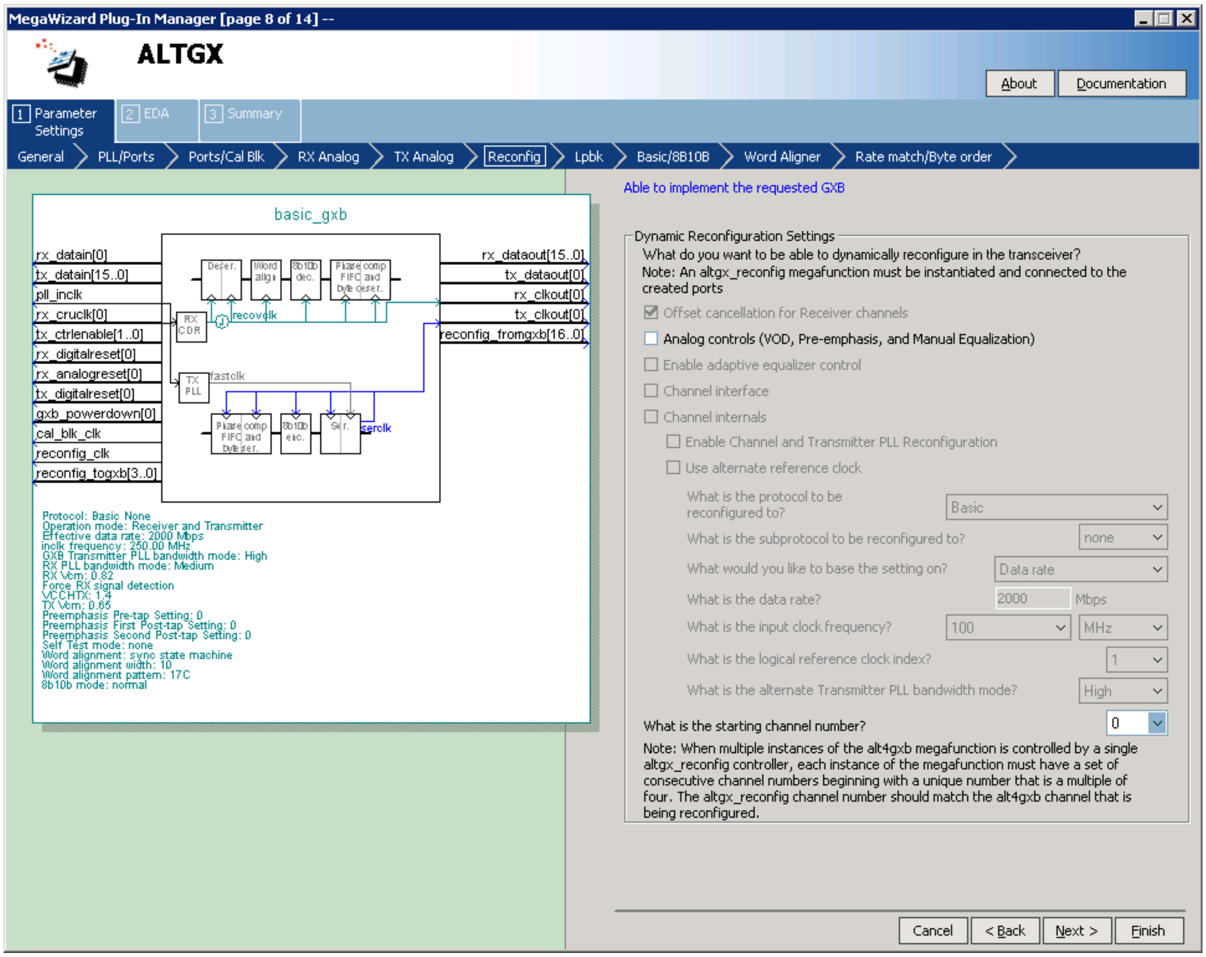
Table 1–5. MegaWizard Plug-In Manager Options (TX Analog Screen for Basic Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
Use external transmitter termination.	This option is available if you want to use an external termination resistor instead of the differential on-chip termination (OCT). Checking this option turns off the transmitter differential OCT.	*Programmable Transmitter Termination* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Select the transmitter termination resistance.	This option selects the transmitter differential termination value. The settings allowed are 85 Ω, 100 Ω, 120 Ω, and 150 Ω.	*Programmable Transmitter Termination* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the voltage output differential (VOD) control setting?	This option selects the VOD of the transmitter buffer. The available VOD settings change based on the transmitter termination resistance value.	*Programmable Output Differential Voltage* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the pre-emphasis first post-tap setting (% of VOD)?	This option sets the amount of pre-emphasis on the transmitter buffer using first post-tap.	*Programmable Pre-Emphasis* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the pre-emphasis pre-tap setting (% of VOD)?	This option sets the amount of pre-emphasis on the transmitter buffer using pre-tap.	*Programmable Pre-Emphasis* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the pre-emphasis second post-tap setting (% of VOD)?	This option sets the amount of pre-emphasis on the transmitter buffer using second post-tap.	*Programmable Pre-Emphasis* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Reconfig Screen for Basic Mode

Figure 1–8 shows the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager for Basic mode.

Figure 1–8. MegaWizard Plug-In Manager Options (Reconfig Screen for Basic Mode)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Dynamic Reconfiguration Settings

What do you want to be able to dynamically reconfigure in the transceiver?

Note: An altgx_reconfig megafunction must be instantiated and connected to the created ports

☑ Offset cancellation for Receiver channels

☐ Analog controls (VOD, Pre-emphasis, and Manual Equalization)

☐ Enable adaptive equalizer control

☐ Channel interface

☐ Channel internals

☐ Enable Channel and Transmitter PLL Reconfiguration

☐ Use alternate reference clock

What is the protocol to be reconfigured to? Basic

What is the subprotocol to be reconfigured to? none

What would you like to base the setting on? Data rate

What is the data rate? 2000 Mbps

What is the input clock frequency? 100 MHz

What is the logical reference clock index? 1

What is the alternate Transmitter PLL bandwidth mode? High

What is the starting channel number? 0

Note: When multiple instances of the alt4gxb megafunction is controlled by a single altgx_reconfig controller, each instance of the megafunction must have a set of consecutive channel numbers beginning with a unique number that is a multiple of four. The altgx_reconfig channel number should match the alt4gxb channel that is being reconfigured.

Table 1–6 describes the available options on the **Reconfig** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

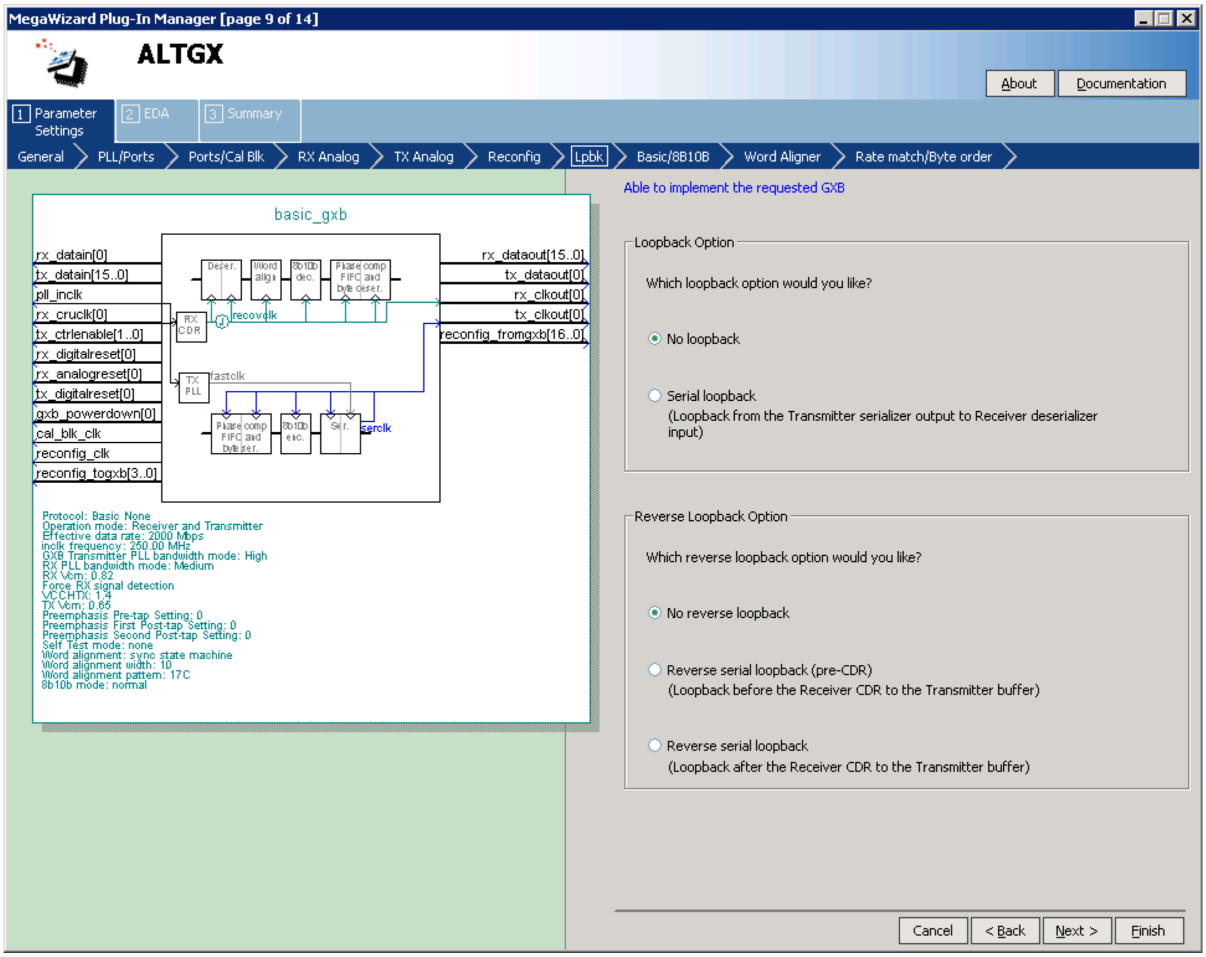
Table 1–6. MegaWizard Plug-In Manager Options (Page 7 for Basic Mode)

ALTGX Setting	Description	Reference
What do you want to be able to dynamically reconfigure in the transceiver?	Available options: Offset cancellation for receiver channels — Make sure that you connect a dynamic reconfiguration controller to all the transceiver channels in the design. This option is enabled by default for **Receiver only** and **Receiver and Transmitter** configurations. It is not available for **Transmitter only** configurations. Analog controls — dynamically reconfigures the PMA control settings like VOD, pre-emphasis, and equalization.	*Offset Cancellation Control for Receiver Channels* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
		PMA Controls Reconfiguration section in the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the starting channel number?	You must set the starting channel number of the first ALTGX MegaWizard instance controlled by the dynamic reconfiguration controller as **0**. Set the starting channel number of the consecutive ALTGX instances controlled by the same dynamic reconfiguration controller, if any, in multiples of 4.	*Logical Channel Addressing* section in the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook.*

Lpbk Screen for Basic Mode

Figure 1–9 shows the **Lpbk** screen of the ALTGX MegaWizard Plug-In Manager for Basic mode.

Figure 1–9. MegaWizard Plug-In Manager - ALTGX (Lpbk Screen for Basic Mode)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Loopback Option

Which loopback option would you like?

- ● No loopback
- ○ Serial loopback
  (Loopback from the Transmitter serializer output to Receiver deserializer input)

Reverse Loopback Option

Which reverse loopback option would you like?

- ● No reverse loopback
- ○ Reverse serial loopback (pre-CDR)
  (Loopback before the Receiver CDR to the Transmitter buffer)
- ○ Reverse serial loopback
  (Loopback after the Receiver CDR to the Transmitter buffer)

Table 1–7 describes the available options on the **Lpbk** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

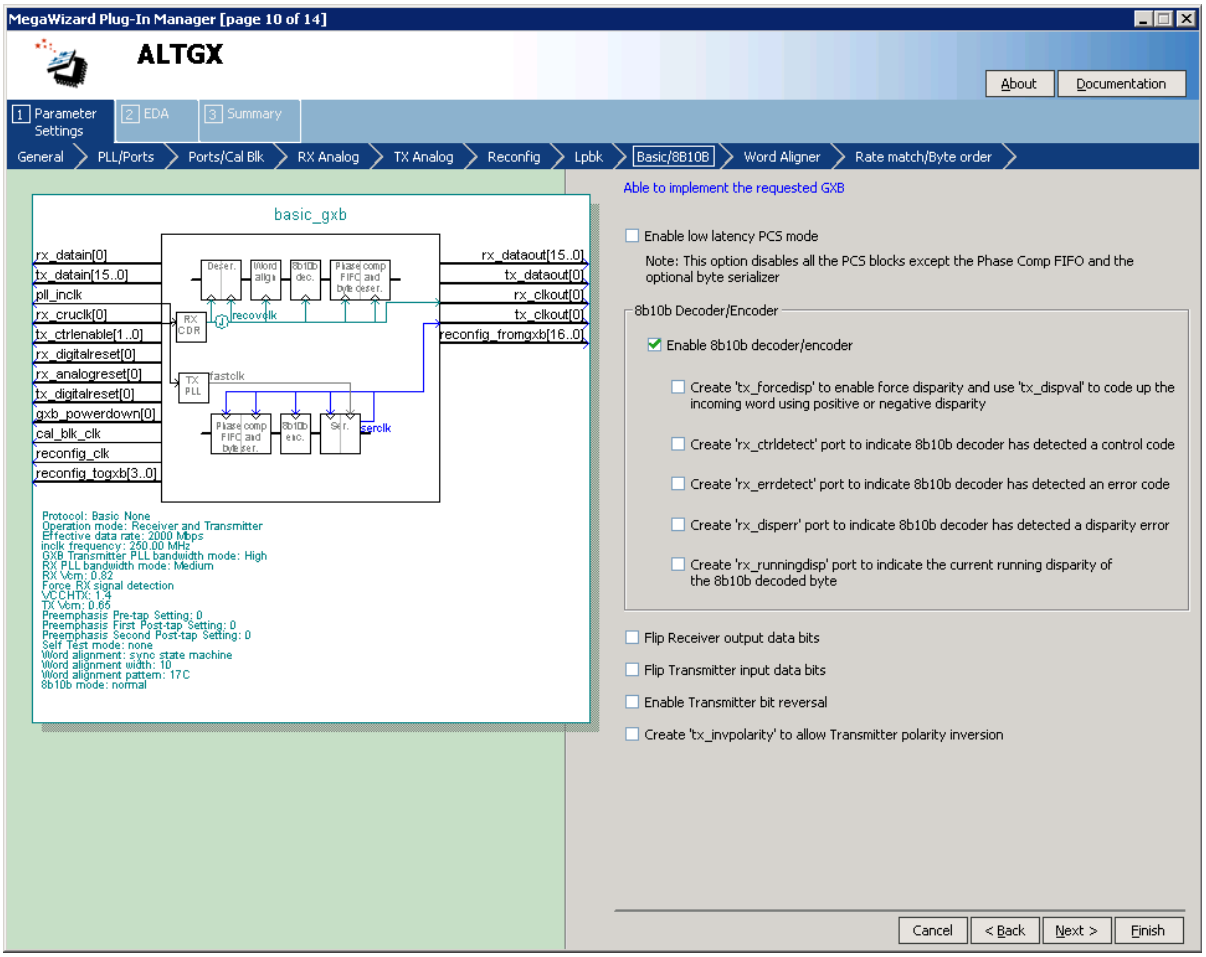
Table 1–7. MegaWizard Plug-In Manager Options (Lpbk Screen for Basic Mode)

ALTGX Setting	Description	Reference
Which loopback option would you like?	There are two options available in Basic mode: ■ No loopback — This is the default mode. ■ Serial loopback — If you select serial loopback, the `rx_seriallpbken` port is available to control the serial loopback feature dynamically. 1'b1 — enables serial loopback 1'b0 — disables serial loopback This signal is asynchronous to the receiver datapath.	*Serial Loopback* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Which reverse loopback option would you like?	There are three options available in Basic mode: ■ No reverse loopback — This is the default mode. ■ Reverse Serial loopback (pre-CDR) — This is the loopback before the receiver's CDR block to the transmitter buffer. The receiver path in the PCS is active but the transmitter side is not. ■ Reverse Serial loopback — This is a loopback after the receiver's CDR block to the transmitter buffer. The receiver path in the PCS is active but the transmitter side is not.	*Loopback Modes* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Basic/8B10B Screen

Figure 1–10 shows the **Basic/8B10B** screen of the MegaWizard Plug-In Manager for the Basic mode set up.

Figure 1–10. MegaWizard Plug-In Manager - ALTGX (Basic/8B10B Screen)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

☐ Enable low latency PCS mode

Note: This option disables all the PCS blocks except the Phase Comp FIFO and the optional byte serializer

8b10b Decoder/Encoder

☑ Enable 8b10b decoder/encoder

☐ Create 'tx_forcedisp' to enable force disparity and use 'tx_dispval' to code up the incoming word using positive or negative disparity

☐ Create 'rx_ctrldetect' port to indicate 8b10b decoder has detected a control code

☐ Create 'rx_errdetect' port to indicate 8b10b decoder has detected an error code

☐ Create 'rx_disperr' port to indicate 8b10b decoder has detected a disparity error

☐ Create 'rx_runningdisp' port to indicate the current running disparity of the 8b10b decoded byte

☐ Flip Receiver output data bits

☐ Flip Transmitter input data bits

☐ Enable Transmitter bit reversal

☐ Create 'tx_invpolarity' to allow Transmitter polarity inversion

Table 1–8 describes the available options on the **Basic/8B10B** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–8. MegaWizard Plug-In Manager Options (Basic/8B10B Screen for Basic Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
Enable low latency PCS mode.	This option disables all the PCS blocks except the Phase Comp FIFO and the optional byte serializer.	*Low Latency PCS Datapath* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable 8B/10B decoder/encoder.	This option is only available if the channel width is 8-bits, 16-bits, or 32-bits.	*8B/10B Decoder* and *8B/10B Encoder* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–8. MegaWizard Plug-In Manager Options (Basic/8B10B Screen for Basic Mode) (Part 2 of 3)

ALTGX Setting	Description	Reference
Create a `tx_forcedisp` to enable Force disparity and use `tx_dispval` to code up the incoming word using positive or negative disparity.	This option allows you to force the current running disparity to positive or negative depending on the `tx_dispval` signal level: ■ Negative current running disparity — When `tx_forcedisp` is asserted and `tx_dispval` is low. ■ Positive current running disparity — When `tx_forcedisp` is asserted and `tx_dispval` is high.	*8B/10B Encoder* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_ctrldetect` port to indicate 8B/10B decoder has detected a control code.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric. This signal indicates whether the decoded 8-bit code group is a data or control code group on this port. If the received 10-bit code group is one of the 12 control code groups (/Kx.y/) specified in IEEE802.3 specification, this signal is driven high. If the received 10-bit code group is a data code group (/Dx.y/), this signal is driven low. The signal width is 1, 2, and 4 bits for a channel width of 8-bits, 16-bits, and 32-bits, respectively.	*8B/10B Decoder* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_errdetect` port to indicate 8B/10B decoder has detected an error code.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric. This signal indicates an 8B/10B code group violation. It is asserted high if the received 10-bit code group has a code violation or disparity error. It is used along with the `rx_disperr` signal to differentiate between a code violation error and/or a disparity error. The signal width is 1, 2 and 4 bits for a channel width of 8-bits, 16-bits, and 32-bits, respectively.	*8B/10B Decoder* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_disperr` port to indicate 8B/10B decoder has detected a disparity code.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric. This signal is asserted high if the received 10-bit code or data group has a disparity error. When this signal goes high, `rx_errdetect` also gets asserted high. When this signal goes high, `rx_errdetect` also gets asserted high. The signal width is 1, 2, and 4 bits for a channel width of 8-bits, 16-bits, and 32-bits, respectively.	*8B/10B Decoder* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_running_disp` port to indicate the current running disparity of the 8B10B decoded byte.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric to indicate the current running disparity of the 8B/10B decoded byte.	—

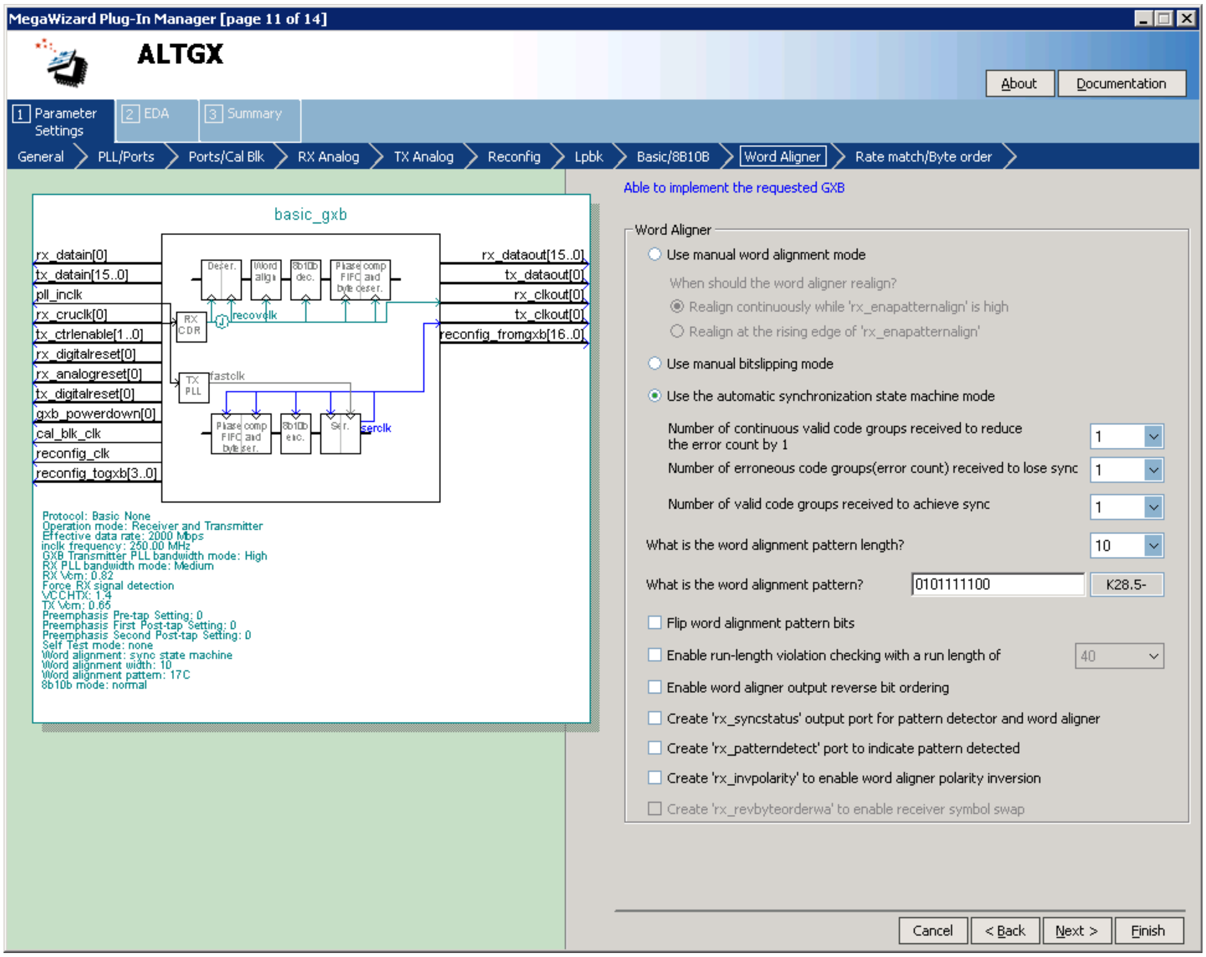
Table 1–8. MegaWizard Plug-In Manager Options (Basic/8B10B Screen for Basic Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
Flip receiver output data bits.	This option reverses the bit order of the parallel receiver data at a byte level at the output of the receiver phase compensation FIFO. For example, if the 16-bit parallel receiver data at the output of the receiver phase compensation FIFO is '10111100 10101101' (16'hBCAD), enabling this option reverses the data on `rx_dataout` port to '00111101 10110101' (16'h3DB5).	—
Flip transmitter input data bits.	This option reverses the bit order of the parallel transmitter data at a byte level at the input of the transmitter phase compensation FIFO. For example, if the 16-bit parallel transmitter data at `tx_datain` port is '10111100 10101101' (16'hBCAD), enabling this option reverses the input data to the transmitter phase compensation FIFO to '00111101 10110101' (16'h3DB5).	—
Enable transmitter bit reversal.	Enabling this option in: ■ Single-width mode — the 8-bit `D[7:0]` or 10-bit `D[9:0]` data at the input of the serializer gets rewired to `D[0:7]` or `D[0:9]`, respectively. ■ Double-width mode — the 16-bit `D[15:0]` or 20-bit `D[19:0]` data at the input of the serializer gets rewired to `D[0:15]` or `D[0:19]`, respectively. For example, if the 8-bit parallel data at the input of the serializer is '00111101', enabling this option reverses this serializer input data to '10111100.'	*Transmitter Bit Reversal* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_invpolarity` port to allow Transmitter polarity inversion.	This optional port allows you to dynamically reverse the polarity of every bit of the data word fed to the serializer in the transmitter data path. Use this option when the positive and negative signals of the differential output from the transmitter (`tx_dataout`) are erroneously swapped on the board.	*Transmitter Polarity Inversion* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Word Aligner Screen for Basic Mode

Figure 1–11 shows the **Word Aligner** screen of the MegaWizard Plug-In Manager for the Basic mode set up.

Figure 1–11. MegaWizard Plug-In Manager - ALTGX (Word Aligner Screen)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Word Aligner

○ Use manual word alignment mode

When should the word aligner realign?

● Realign continuously while 'rx_enapatternalign' is high

○ Realign at the rising edge of 'rx_enapatternalign'

○ Use manual bitslipping mode

● Use the automatic synchronization state machine mode

Number of continuous valid code groups received to reduce the error count by 1: 1

Number of erroneous code groups(error count) received to lose sync: 1

Number of valid code groups received to achieve sync: 1

What is the word alignment pattern length? 10

What is the word alignment pattern? 0101111100 K28.5-

□ Flip word alignment pattern bits

□ Enable run-length violation checking with a run length of 40

□ Enable word aligner output reverse bit ordering

□ Create 'rx_syncstatus' output port for pattern detector and word aligner

□ Create 'rx_patterndetect' port to indicate pattern detected

□ Create 'rx_invpolarity' to enable word aligner polarity inversion

□ Create 'rx_revbyteorderwa' to enable receiver symbol swap

Table 1–9 describes the available options on the **Word Aligner** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–9. MegaWizard Plug-In Manager Options (Word Aligner Screen for Basic Mode) (Part 1 of 4)

ALTGX Setting	Description	Reference
Use manual word alignment mode.	Enabling this option sets the word aligner in manual alignment mode. In manual alignment mode, the word aligner operation is controlled by the input signal `rx_enapatternalign`.	*Manual Alignment Mode Word Aligner in 8-bit PMA-PCS Interface Modes, Manual Alignment Mode Word Aligner in 10-bit PMA-PCS Interface Modes* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
When should the word aligner realign?	The available options differ based on: Single-width mode - The Quartus II ALTGX MegaWizard Plug-In Manager automatically sets the behavior of the `rx_enapatternalign` signal to be either: ■ Edge sensitive for 8-bit PMA-PCS interface ■ Level sensitive for 10-bit PMA-PCS interface Double-width mode - This option is not available for double-width mode. The behavior of the `rx_enapatternalign` signal is edge sensitive in double-width mode.	*Manual Alignment Mode Word Aligner in 8-bit PMA-PCS Interface Modes, Manual Alignment Mode Word Aligner in 10-bit PMA-PCS Interface Modes* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Use manual bitslipping mode.	This option sets the word aligner in Bit-Slip Mode. Enabling this option creates an input signal `rx_bitslip` to control the word aligner. At every rising edge of the `rx_bitslip` signal, the bit slip circuitry slips one bit into the received data stream, effectively shifting the word boundary by one bit.	*Bit-Slip Mode Word Aligner in 8-bit PMA-PCS Interface Modes, Bit-Slip Mode Word Aligner in 10-bit PMA-PCS Interface Mode, Bit-Slip Mode Word Aligner in 16-bit PMA-PCS Interface Modes,* and *Bit-Slip Mode Word Aligner in 20-bit PMA-PCS Interface Modes* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Use the Automatic synchronization state machine mode.	This option sets the word aligner in Automatic synchronization state machine mode. This mode is available only in the single-width mode for 8B/10B encoded data: ■ 10-bit PCS-PMA Interface where 8B/10B encoder is enabled Or ■ 10-bit PCS-PMA Interface where 8B/10B is disabled but the data is already 8B/10B encoded	*Automatic Synchronization State Machine Mode Word Aligner in 10-bit PMA-PCS Interface Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–9. MegaWizard Plug-In Manager Options (Word Aligner Screen for Basic Mode) (Part 2 of 4)

ALTGX Setting	Description	Reference
Number of valid code groups received to achieve synchronization.	Use this option in the automatic synchronization state machine mode to indicate the number of word alignment patterns that it must receive without intermediate erroneous code groups to achieve synchronization. The `rx_syncstatus` signal is driven high to indicate that synchronization has been achieved.	*Automatic Synchronization State Machine Mode Word Aligner in 10-bit PMA-PCS Interface Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Number of erroneous code groups (error count) received to lose synchronization.	Use this option in the automatic synchronization state machine mode to indicate the number of erroneous code groups (error count) that it must receive to lose synchronization. The loss-of-synch is indicated by the `rx_syncstatus` signal going low.	*Automatic Synchronization State Machine Mode Word Aligner in 10-bit PMA-PCS Interface Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Number of continuous valid code groups received to reduce the error count by 1.	Use this option in the automatic synchronization state machine mode to indicate the number of continuous valid code groups that it must receive between erroneous code groups to reduce the error count by 1. The `rx_syncstatus` stays high as long as the error count is less than the programmed error count.	*Automatic Synchronization State Machine Mode Word Aligner in 10-bit PMA-PCS Interface Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the word alignment pattern length?	This option sets the word alignment pattern length. The available choices depend on the following two conditions: ■ Whether the data is 8B/10B encoded or not ■ Which mode is used in single-width mode: →for 8-bit PCS-PMA Interface (8B/10B encoder disabled), only 16 bits is allowed. →for 10-bit PCS-PMA, 7 and 10 bits are allowed. ■ Which mode is used in double-width mode: →for 16-bit PCS-PMA Interface (8B/10B encoder disabled), 8, 16 and 32 bits are allowed. →for 20-bit PCS-PMA Interface, 7, 10 and 20 bits are allowed.	*Word Aligner in Single-Width Mode* and *Double-Width Mode Word Aligner* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the word alignment pattern?	Enter the word alignment pattern in MSB to LSB order with MSB at the left most bit position. The length of the alignment pattern is based on the option **What is the word alignment pattern length?** The word aligner restores the word boundary by looking for the pattern that you enter here. For example, if you want to set the word alignment pattern as /K28.5/: ■ You should enter the word alignment pattern length = 10 ■ You should enter '0101111100' (17C) in this option	*Word Aligner in Single-Width Mode* and *Double-Width Mode Word Aligner* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–9. MegaWizard Plug-In Manager Options (Word Aligner Screen for Basic Mode) (Part 3 of 4)

ALTGX Setting	Description	Reference
Flip word alignment pattern bits.	When this option is enabled, the Quartus II ALTGX MegaWizard Plug-In Manager flips the bit order of the pattern that you enter in the **What is the word alignment pattern?** option and uses the flipped version as the word alignment pattern. For example, if you enter '0101111100' (17C) as the word alignment pattern and enable this option, the word aligner uses '0011111010' as the word alignment pattern.	—
Enable run-length violation checking with a run length of:	This option creates the output signal `rx_rlv`. Enabling this option also activates the run-length violation circuit. If the number of continuous 1's and 0's exceeds the number that you set in this option, the run-length violation circuit asserts the `rx_rlv` signal. The `rx_rlv` signal is asynchronous to the receiver data path and is asserted for a minimum of two recovered clock cycles in single-width mode. Similarly, it is asserted for a minimum of three recovered clock cycles in double-width mode. The run length limits are as follows: Single-width mode — ■ 8-bit and 16-bit channel width: 4 to 128 in increments of 4 ■ 10-bit and 20-bit channel width: 5 to 160 in increments of 5 Double-width mode — ■ 16-bit and 32-bit channel width: 8 to 512 in increments of 8 ■ 20-bit and 40-bit channel width: 10 to 640 in increments of 10	*Programmable Run Length Violation Detection* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable word aligner output reverse bit ordering.	In manual bit-slip mode, this option creates an input port `rx_revbitorderwa` to dynamically reverse the bit order at the output of the receiver word aligner.	*Receiver Bit Reversal* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_syncstatus` output port for pattern detector and word aligner.	This is an output status signal that the word aligner forwards to the FPGA fabric to indicate that synchronization has been achieved. This signal is synchronous with the parallel receiver data on the `rx_dataout` port. This signal is not available in the Bit-Slip Mode. The signal width is 1, 2 and 4 bits for a channel width of 8-bits/10-bits, 16-bits/20-bits, and 32-bits/40-bits, respectively.	*Table 1-24 and Word Aligner in Single-Width Mode* and *Double-Width Mode Word Aligner* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_patterndetect` port to indicate pattern detected.	This is an output status signal that the word aligner forwards to the FPGA fabric to indicate that the word alignment pattern programmed has been detected in the current word boundary. The signal width is 1, 2, and 4 bits for a channel width of 8-bits/10-bits, 16-bits/20-bits, and 32-bits/40-bits, respectively.	*Table 1-24 and Word Aligner in Single-Width Mode* and *Double-Width Mode Word Aligner* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

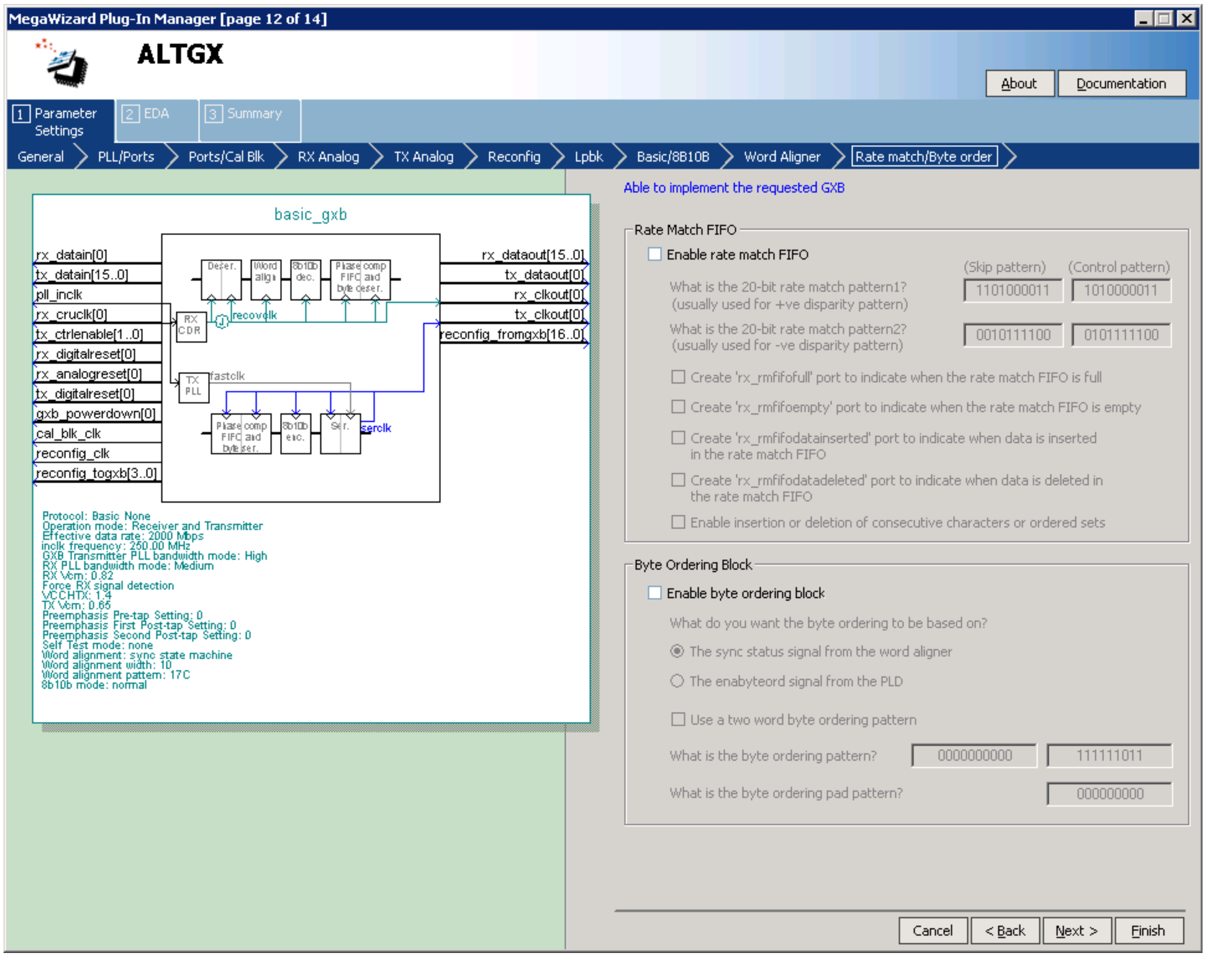
Table 1–9. MegaWizard Plug-In Manager Options (Word Aligner Screen for Basic Mode) (Part 4 of 4)

ALTGX Setting	Description	Reference
Create an `rx_invpolarity` port to enable word aligner polarity inversion.	This optional port allows you to dynamically reverse the polarity of every bit of the received data at the input of the word aligner. Use this option when the positive and negative signals of the differential input to the receiver (`rx_datain`) are erroneously swapped on the board.	*Receiver Polarity Inversion* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_revbyteorderwa` to enable Receiver symbol swap.	This is an optional input port which is available only in the double-width mode. It creates an `rx_revbyteorderwa` port to dynamically swap the MSByte and LSByte of the data at the output of the word aligner in the receiver data path. Enabling this option compensates for the erroneous swapping of bytes at the upstream transmitter and hence corrects the data received by the downstream systems. For example, if the 16-bit output of the word aligner is 0B0A, asserting this signal `rx_revbyteorderwa` swaps the two bytes and the output will now be 0A0B.	*Receiver Byte Reversal in Basic Double-Width Modes* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Rate Match/Byte Order Screen for Basic Mode

Figure 1–12 shows the **Rate Match/Byte Order** screen of the MegaWizard Plug-In Manager for the Basic mode set up.

Figure 1–12. MegaWizard Plug-In Manager - ALTGX (Rate Match/Byte Order Screen)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Rate Match FIFO

□ Enable rate match FIFO

	(Skip pattern)	(Control pattern)
What is the 20-bit rate match pattern1? (usually used for +ve disparity pattern)	1101000011	1010000011
What is the 20-bit rate match pattern2? (usually used for -ve disparity pattern)	0010111100	0101111100

□ Create 'rx_rmfifofull' port to indicate when the rate match FIFO is full

□ Create 'rx_rmfifoempty' port to indicate when the rate match FIFO is empty

□ Create 'rx_rmfifodatainserted' port to indicate when data is inserted in the rate match FIFO

□ Create 'rx_rmfifodatadeleted' port to indicate when data is deleted in the rate match FIFO

□ Enable insertion or deletion of consecutive characters or ordered sets

Byte Ordering Block

□ Enable byte ordering block

What do you want the byte ordering to be based on?

● The sync status signal from the word aligner

○ The enabyteord signal from the PLD

□ Use a two word byte ordering pattern

What is the byte ordering pattern? 0000000000 111111011

What is the byte ordering pad pattern? 000000000

Table 1–10 describes the available options on the **Rate Match/Byte Order** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–10. MegaWizard Plug-In Manager Options (Rate Match/Byte Order Screen for Basic Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
Enable rate match FIFO.	This option enables the rate match (clock rate compensation) FIFO. The rate match block consists of a 20-word deep FIFO. Depending on the PPM difference, the rate match FIFO controls insertion and deletion of skip characters based on the 20-bit rate match pattern you enter in the options: **What is the 20-bit rate match pattern1?** and **What is the 20-bit rate match pattern2?** To enable this block: ■ The transceiver channel must have both the transmitter and the receiver channels instantiated. You must select **Receiver and Transmitter** option in the **What is the operation mode?** field in the **General** screen. ■ You must also enable the 8B/10B encoder/decoder in **Basic/8B10B** screen. The rate match block is capable of compensating up to ±300 PPM difference between the upstream transmitter clock and the local receiver's input reference clock.	*Rate Match FIFO in Basic Single-Width Mode* and *Rate Match FIFO in Basic Double-Width Mode* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the 20-bit rate match pattern1? (usually used for +ve disparity pattern)	Enter a 10-bit skip pattern and a 10-bit control pattern. In the skip pattern field, you must choose a 10-bit code group that has neutral disparity. When the rate matcher receives the 10-bit control pattern followed by the 10-bit skip pattern, it inserts or deletes the 10-bit skip pattern as necessary to avoid the rate match FIFO overflow or underflow conditions. *(1)*	*Rate Match FIFO in Basic Single-Width Mode* and *Rate Match FIFO in Basic Double-Width Mode* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the 20-bit rate match pattern2? (usually used for -ve disparity pattern)	Enter a 10-bit skip pattern and a 10-bit control pattern. In the skip pattern field, you must choose a 10-bit code group that has neutral disparity. When the rate matcher receives the 10-bit control pattern followed by the 10-bit skip pattern, it inserts or deletes the 10-bit skip pattern as necessary to avoid the rate match FIFO from overflowing or under-runnning. *(1)*	*Rate Match FIFO in Basic Single-Width Mode* and *Rate Match FIFO in Basic Double-Width Mode* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable the `rx_rmfifofull` flag to indicate when the rate match FIFO is full.	This option creates the output port `rx_rmfifofull` when you enable the option **Enable Rate Match FIFO**. It is a status flag that the rate match block forwards to the FPGA fabric. It indicates when the rate match FIFO block is full (20 words). This signal remains high as long as the FIFO is full. It is asynchronous to the receiver data path.	*Rate Match FIFO in Basic Single-Width Mode* and *Rate Match FIFO in Basic Double-Width Mode* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–10. MegaWizard Plug-In Manager Options (Rate Match/Byte Order Screen for Basic Mode) (Part 2 of 3)

ALTGX Setting	Description	Reference
Enable the `rx_rmfifoempty` flag to indicate when the rate match FIFO is empty.	This option creates the output port `rx_rmfifoempty` when you enable the option **Enable Rate Match FIFO**. It is a status flag that the rate match block forwards to the FPGA fabric. It indicates when the rate match FIFO block is empty (5 words full). This signal remains high as long as the FIFO is empty. It is asynchronous to the receiver data path.	*Rate Match FIFO in Basic Single-Width Mode* and *Rate Match FIFO in Basic Double-Width Mode* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable the `rx_rmfifodatainserted` flag to indicate when data is inserted in the rate match FIFO.	This option creates the output port `rx_rmfifodatainserted` flag when you enable the option **Enable Rate Match FIFO**. It is a status flag that the rate match block forwards to the FPGA fabric. It indicates the insertion of skip patterns. For every deletion, this signal is high for one parallel clock cycle.	*Rate Match FIFO in Basic Single-Width Mode* and *Rate Match FIFO in Basic Double-Width Mode* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable the `rx_rmfifodatadeleted` flag to indicate when data is deleted from the rate match FIFO.	This option creates the output port `rx_rmfifodatadeleted` flag when you enable the option **Enable Rate Match FIFO**. It is a status flag that the rate match block forwards to the FPGA fabric. It indicates the deletion of skip patterns. For every insertion, this signal is high for one parallel clock cycle.	*Rate Match FIFO in Basic Single-Width Mode* and *Rate Match FIFO in Basic Double-Width Mode* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable byte ordering block.	This option enables the byte ordering block. It is available in both single-width and double-width modes. It is available only when the channel width is 16-bits/20-bits in single-width mode and 32-bits/40-bits in double-width mode. As soon as the byte ordering block sees the rising edge of the appropriate signal, it compares the LSByte coming out of the byte deserializer with the byte ordering pattern. If it doesn't match, the byte ordering block inserts the pad character that you enter in the option: **What is the byte ordering pad pattern?** such that the byte ordering pattern is seen in the LSByte position. Insertion of this pad character enables the byte ordering block to restore the correct byte order.	*Byte Ordering Block* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What do you want the byte ordering to be based on?	This option is available only when the byte ordering block is enabled. This option allows you to trigger the byte ordering block on the rising edge of either the `rx_syncstatus` signal or the user-controlled `rx_enabyteord` signal from the FPGA fabric.	*Byte Ordering Block* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the byte ordering pattern?	This option is available only when the byte ordering block is enabled. Enter the 10-bit pattern that the byte ordering block must place in the LSByte position of the receiver parallel data on the `rx_dataout` port.	*Byte Ordering Block* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–10. MegaWizard Plug-In Manager Options (Rate Match/Byte Order Screen for Basic Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
What is the byte ordering pad pattern?	When the byte ordering block does not find the byte ordering pattern in the LSByte position of the data coming out of the byte deseriazlier, it inserts this byte ordering pad pattern such that the byte ordering pattern is seen in the LSByte position of the receiver parallel data on the `rx_dataout` port. Insertion of this pad character enables the byte ordering block to restore the correct byte order.	*Byte Ordering Block* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

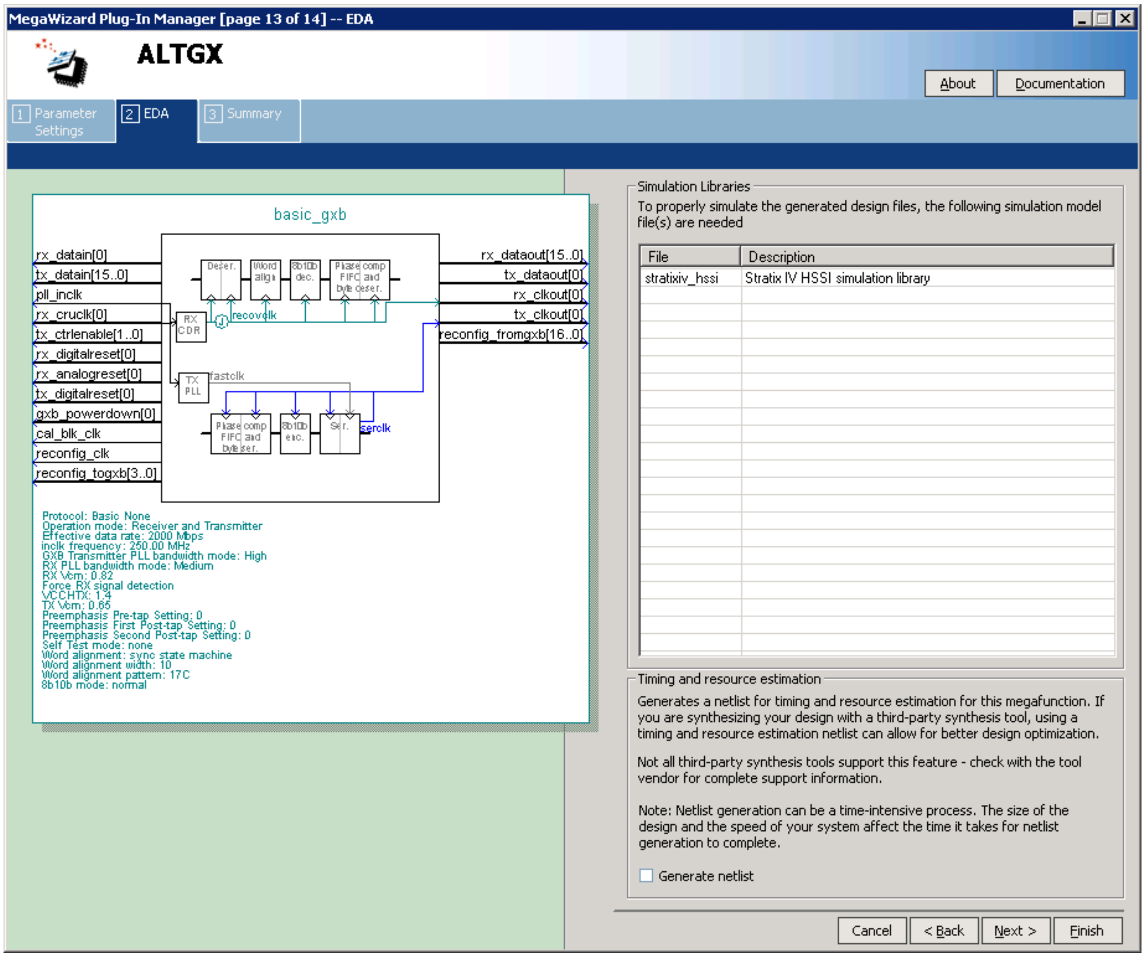
Note to Table 1–10:

(1) If you want the rate matcher to insert/delete both the positive and negative disparities of the 20-bit rate matching pattern, enter the positive disparity as pattern1 and negative disparity as pattern2.

EDA Screen for Basic Mode

Figure 1–13 shows the **EDA** screen of the MegaWizard Plug-In Manager for the Basic mode. The **Generate Netlist** option generates a netlist for the third party EDA synthesis tool to estimate timing and resource utilization for the ALTGX instance.

Figure 1–13. MegaWizard Plug-In Manager - ALTGX (EDA Screen for Basic Mode)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Simulation Libraries

To properly simulate the generated design files, the following simulation model file(s) are needed

File	Description
stratixiv_hssi	Stratix IV HSSI simulation library

Timing and resource estimation

Generates a netlist for timing and resource estimation for this megafunction. If you are synthesizing your design with a third-party synthesis tool, using a timing and resource estimation netlist can allow for better design optimization.

Not all third-party synthesis tools support this feature - check with the tool vendor for complete support information.

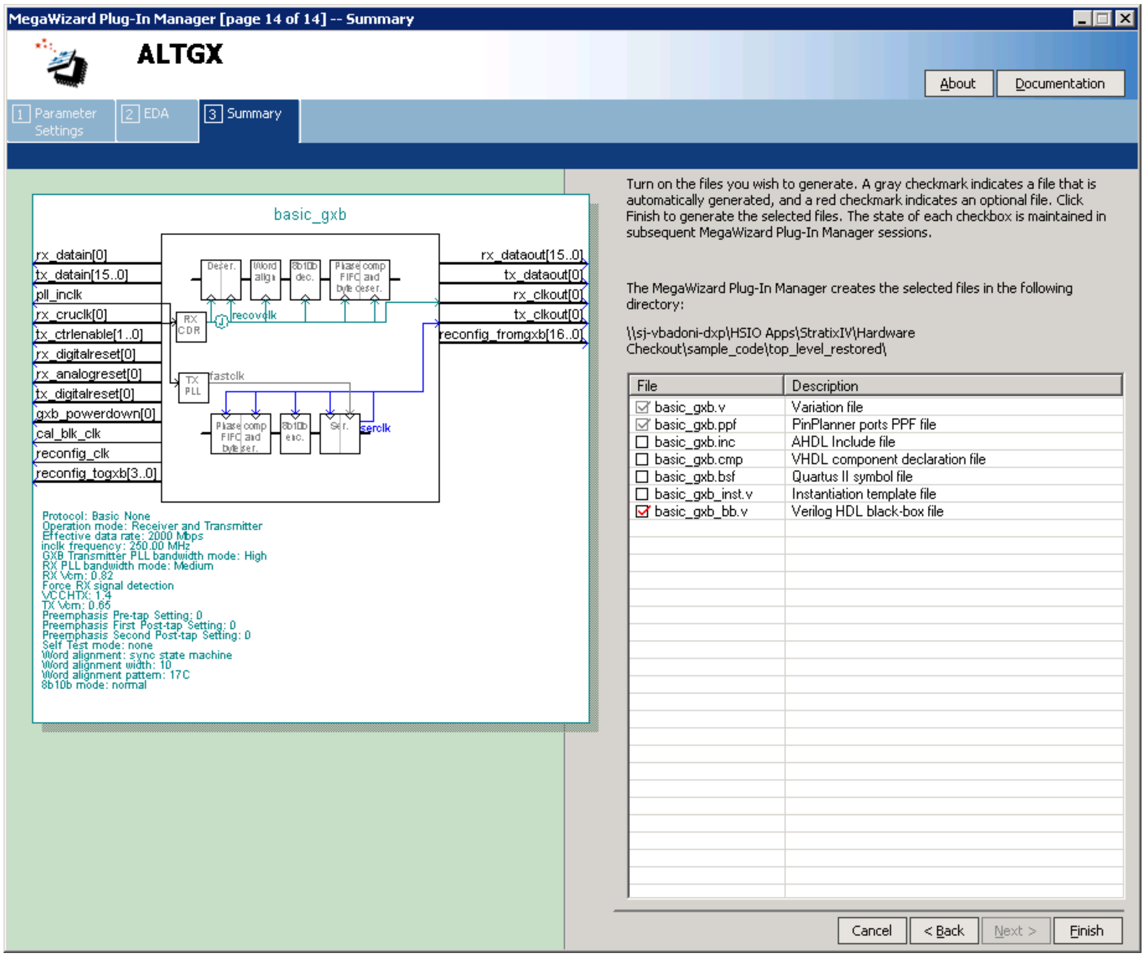
Note: Netlist generation can be a time-intensive process. The size of the design and the speed of your system affect the time it takes for netlist generation to complete.

☐ Generate netlist

Summary for Basic Mode

Figure 1–14 shows the **Summary** screen of the MegaWizard Plug-In Manager for the Basic mode. You can select optional files on this page. After you make your selections, click **Finish** to generate the files.

Figure 1–14. MegaWizard Plug-In Manager - ALTGX (Summary Screen)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Turn on the files you wish to generate. A gray checkmark indicates a file that is automatically generated, and a red checkmark indicates an optional file. Click Finish to generate the selected files. The state of each checkbox is maintained in subsequent MegaWizard Plug-In Manager sessions.

The MegaWizard Plug-In Manager creates the selected files in the following directory:

\\sj-vbadoni-dxp\HSIO Apps\StratixIV\Hardware Checkout\sample_code\top_level_restored\

File	Description
☑ basic_gxb.v	Variation file
☑ basic_gxb.ppf	PinPlanner ports PPF file
☐ basic_gxb.inc	AHDL Include file
☐ basic_gxb.cmp	VHDL component declaration file
☐ basic_gxb.bsf	Quartus II symbol file
☐ basic_gxb_inst.v	Instantiation template file
☑ basic_gxb_bb.v	Verilog HDL black-box file

Physical Interface for PCI-Express (PIPE) Mode

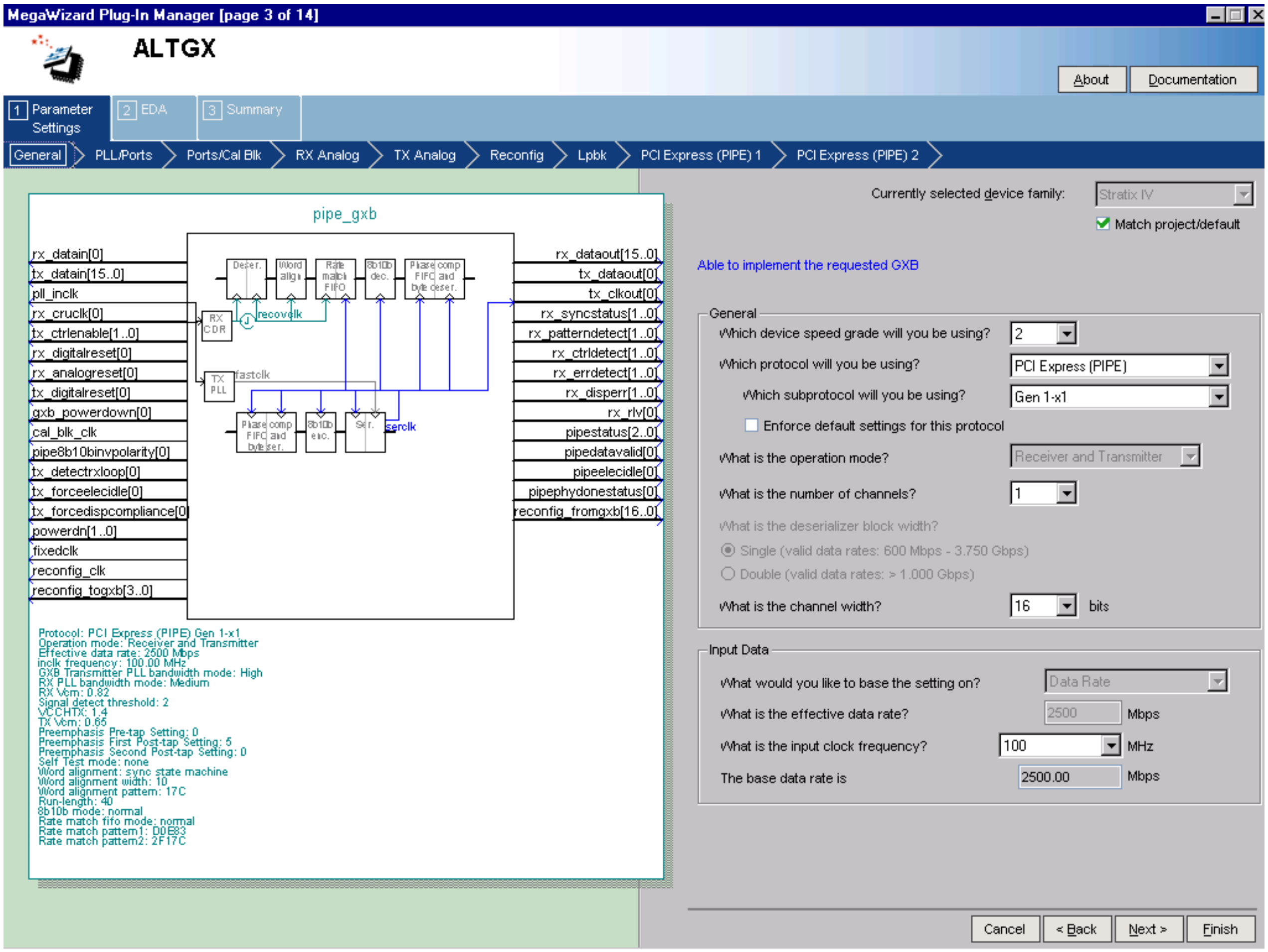
This section provides descriptions of the options available on the individual pages of the ALTGX MegaWizard Plug-In Manager for the PCI Express (PIPE) mode. The MegaWizard Plug-In Manager provides a warning if any of the settings you choose are illegal.

☞ The word aligner and rate matcher operations and patterns are pre-configured for the PCI Express (PIPE) mode and cannot be altered.

General Screen for PCI Express Mode

Figure 1–15 shows the **General** screen of the ALTGX MegaWizard Plug-In Manager for the PCI Express (PIPE) mode.

Figure 1–15. MegaWizard Plug-In Manager - ALTGX (General Screen)



Protocol: PCI Express (PIPE) Gen 1-x1
Operation mode: Receiver and Transmitter
Effective data rate: 2500 Mbps
inclk frequency: 100.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Signal detect threshold: 2
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 5
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
Run-length: 40
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: D0B83
Rate match pattern2: 2F17C

Currently selected device family: Stratix IV
☑ Match project/default

Able to implement the requested GXB

General
Which device speed grade will you be using? 2
Which protocol will you be using? PCI Express (PIPE)
Which subprotocol will you be using? Gen 1-x1
☐ Enforce default settings for this protocol
What is the operation mode? Receiver and Transmitter
What is the number of channels? 1
What is the deserializer block width?
● Single (valid data rates: 600 Mbps - 3.750 Gbps)
○ Double (valid data rates: > 1.000 Gbps)
What is the channel width? 16 bits

Input Data
What would you like to base the setting on? Data Rate
What is the effective data rate? 2500 Mbps
What is the input clock frequency? 100 MHz
The base data rate is 2500.00 Mbps

Table 1–11 describes the available options on the **General** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–11. MegaWizard Plug-In Manager Options (General Screen for PCI Express (PIPE) Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
Which device speed grade will you be using?	Select the speed grade of the device you will be using. The available speed grades are 2, 2x, 3, and 4.	—
Which protocol will you be using?	Determines the specific functional mode under which the transceiver operates. For the PCI Express (PIPE) mode, you must select the **PCI Express (PIPE)** protocol.	*PCI Express (PIPE) Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

Table 1–11. MegaWizard Plug-In Manager Options (General Screen for PCI Express (PIPE) Mode) (Part 2 of 3)

ALTGX Setting	Description	Reference
Which subprotocol will you be using?	In PCI Express (PIPE) mode, there are six different subprotocols: ■ Gen1 ×1 - The transceiver is configured as a single-lane PCI Express link for a 2.5 Gbps data rate. ■ Gen1 ×4 - The transceiver is configured as a four-lane PCI Express link for a data rate of 2.5 Gbps. ■ Gen1 ×8 - The transceiver is configured as an eight-lane PCI Express link for a data rate of 2.5 Gbps. ■ Gen2 ×1 - The transceiver is configured as a single-lane PCI Express link for a 5.0 Gbps data rate. ■ Gen2 ×4 - The transceiver is configured as a four-lane PCI Express link for a data rate of 5.0 Gbps. ■ Gen2 ×8 - The transceiver is configured as an eight-lane PCI Express link for a data rate of 5.0 Gbps.	*PCI Express (PIPE) Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Enforce default settings for this protocol.	Selecting this option skips the **PCI** screen in the PCI Express (PIPE) MegaWizard Plug-In Manager. The **PCI** screen allows you to select the PCI Express (PIPE)-specific ports for your design. If you select this option, all PCI Express (PIPE)-specific ports are used.	—
What is the operation mode?	Only the **Receiver and Transmitter** mode is allowed in the PCI Express (PIPE) mode. **Receiver only** and **Transmitter only** modes are not allowed.	—
What is the number of channels?	It is the number of channels required with the same configuration. In a ×4 subprotocol, the number of channels increments by 4. In a ×8 subprotocol, the number of channels increment by 8.	—
What is the deserializer block width?	PCI Express (PIPE) mode only operates in a single-width mode. Double-width mode is not available.	—
What is the channel width?	This option determines the FPGA fabric-Transceiver interface width. In PCI Express (PIPE) Gen1 (2.5 Gbps) mode, 8 and 16 bits are allowed. In the PCI Express (PIPE) Gen2 (5 Gbps) mode, only 16 bits is allowed.	—
What would you like to base the setting on?	This option is not available for selection in PCI Express (PIPE) mode.	—
What is the effective data rate?	This option is not available for selection in PCI Express (PIPE) mode. It defaults to 2500 Mbps for PCI Express (PIPE) Gen1 mode and 5000 Mbps for PCI Express (PIPE) Gen 2 mode.	—

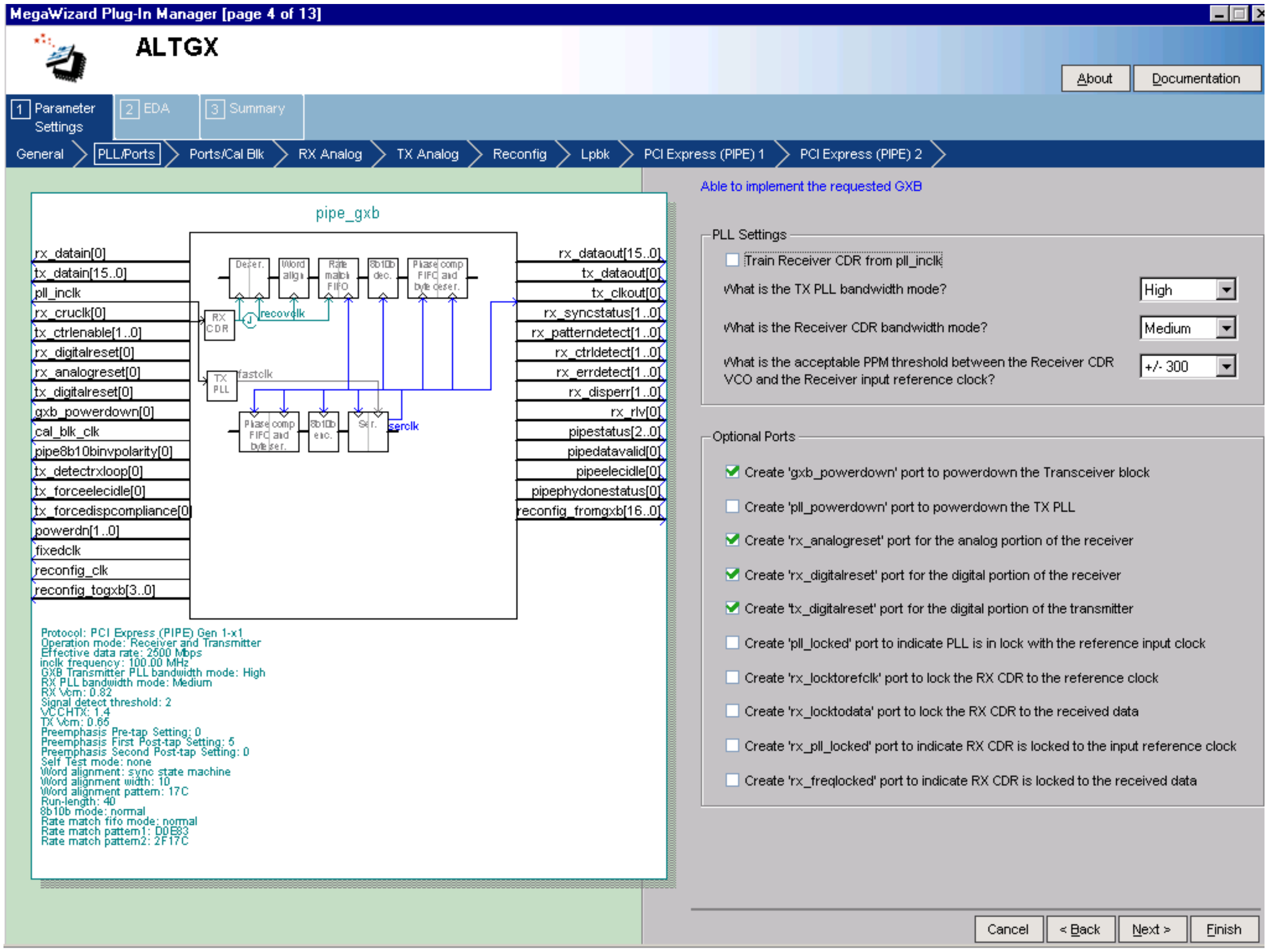
Table 1–11. MegaWizard Plug-In Manager Options (General Screen for PCI Express (PIPE) Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
What is the input clock frequency?	This option is not available for selection in PCI Express (PIPE) mode. The input reference clock frequency is fixed to 100 MHz in PCI Express (PIPE) mode.	*CMU PLL and Receiver CDR Input Reference Clock* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Specify base data rate.	This option is not available for selection because the data rate is fixed in PCI Express (PIPE) mode.	—

PLL/Ports Screen for PCI Express Mode

Figure 1–16 shows the **PLL/Ports** screen of the ALTGX MegaWizard Plug-In Manager for PCI Express (PIPE) mode.

Figure 1–16. MegaWizard Plug-In Manager - ALTGX (PLL/Ports Screen for PCI Express Mode)



Protocol: PCI Express (PIPE) Gen 1-x1
Operation mode: Receiver and Transmitter
Effective data rate: 2500 Mbps
inclk frequency: 100.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Signal detect threshold: 2
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 5
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
Run-length: 40
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: D0E83
Rate match pattern2: 2F17C

Able to implement the requested GXB

PLL Settings

☐ Train Receiver CDR from pll_inclk

What is the TX PLL bandwidth mode? High

What is the Receiver CDR bandwidth mode? Medium

What is the acceptable PPM threshold between the Receiver CDR VCO and the Receiver input reference clock? +/- 300

Optional Ports

☑ Create 'gxb_powerdown' port to powerdown the Transceiver block
☐ Create 'pll_powerdown' port to powerdown the TX PLL
☑ Create 'rx_analogreset' port for the analog portion of the receiver
☑ Create 'rx_digitalreset' port for the digital portion of the receiver
☑ Create 'tx_digitalreset' port for the digital portion of the transmitter
☐ Create 'pll_locked' port to indicate PLL is in lock with the reference input clock
☐ Create 'rx_locktorefclk' port to lock the RX CDR to the reference clock
☐ Create 'rx_locktodata' port to lock the RX CDR to the received data
☐ Create 'rx_pll_locked' port to indicate RX CDR is locked to the input reference clock
☐ Create 'rx_freqlocked' port to indicate RX CDR is locked to the received data

Table 1–12 describes the available options on the **PLL/ports** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–12. MegaWizard Plug-In Manager Options (PLL/Ports Screen for PCI Express (PIPE) Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Train receiver CDR from `pll_inclk`.	If you select this option, the input reference clock to the CMU PLL trains the receiver CDR.	Table 1-2 in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the TX PLL bandwidth mode?	The appropriate bandwidth settings will be fixed after characterization.	*PLL Bandwidth Setting* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the receiver CDR bandwidth mode?	The appropriate bandwidth settings will be fixed after characterization.	—
What is the acceptable PPM threshold between the Receiver CDR VCO and the Receiver input reference clock?	In automatic lock mode, the CDR remains in LTD mode as long as the PPM difference between the CDR VCO output clock and the input reference clock is less than the PPM value that you set in this option. If the PPM difference is greater than the PPM value that you set in this option, the CDR switches to LTR mode. The range of values available in this option is ±62.5 ppm to ±1000 ppm. *(1)*	*Automatic Lock Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `gxb_powerdown` port to power down the Transceiver block.	This is an optional signal. When asserted, it powers down the entire transceiver block. If none of the channels are instantiated in a transceiver block, the Quartus II software automatically powers down the entire transceiver block.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `pll_powerdown` port to power down the TX PLL.	Each transceiver block has two CMU PLLs. Each CMU PLL has a dedicated power down signal called `pll_powerdown`. This signal powers down the CMU PLL.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_analogreset` port for the analog portion of the receiver.	Receiver analog reset port available in **Receiver only** and **Receiver and Transmitter** operation modes. Resets part of the analog portion of the receiver CDR in the receiver channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles. In ×1 mode, each channel has an independent `rx_analogreset` signal. In ×4 and ×8 modes, all bonded channels have a common `rx_analogreset` signal.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–12. MegaWizard Plug-In Manager Options (PLL/Ports Screen for PCI Express (PIPE) Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
Create an `rx_digitalreset` port for the digital portion of the receiver.	Receiver digital reset port available in **Receiver only** and **Receiver and Transmitter** operation modes. Resets the PCS portion of the receiver channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.In ×1 mode, each channel has an independent `rx_digitalreset` signal. In ×4 and ×8 modes, all bonded channels have a common `rx_digitalreset` signal.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_digitalreset` port for the digital portion of the transmitter.	Transmitter digital reset port available in **Transmitter only** and **Receiver and Transmitter** operation modes. Resets the PCS portion of the transmitter channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles. In ×1 mode, each channel has an independent `tx_digitalreset` signal. In ×4 and ×8 modes, all bonded channels have a common `tx_digitalreset` signal.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `pll_locked` port to indicate PLL is in lock with the input reference clock.	Each CMU PLL has a dedicated `pll_locked` signal that is fed to the FPGA fabric to indicate when the PLL is locked to the input reference clock.	*Transceiver Reset Sequences* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_locktorefclk` port to lock the RX CDR to the input reference clock.	This is an optional port. When this signal is asserted high, the LTR/LTD controller forces the receiver CDR to lock to the phase and frequency of the input reference clock. *(1)*, *(2)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_locktodata` port to lock the RX CDR to the received data.	This is an optional port. When this signal is asserted high, the LTR/LTD controller forces the receiver CDR to lock to the received data. *(1)*, *(2)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_pll_locked` port to indicate RX CDR is locked to the input reference clock.	In LTR mode, this signal is asserted high to indicate that the receiver CDR has locked to the phase and frequency of the input reference clock. In LTD mode, this signal has no significance. *(1)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_freqlocked` port to indicate RX CDR is locked to the received data.	This signal gets asserted high to indicate that the receiver CDR has switched from LTR to LTD mode. This signal has relevance only in automatic lock mode and may be required to control the transceiver resets, as discussed in the *User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.* *(1)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Notes to Table 1–12:

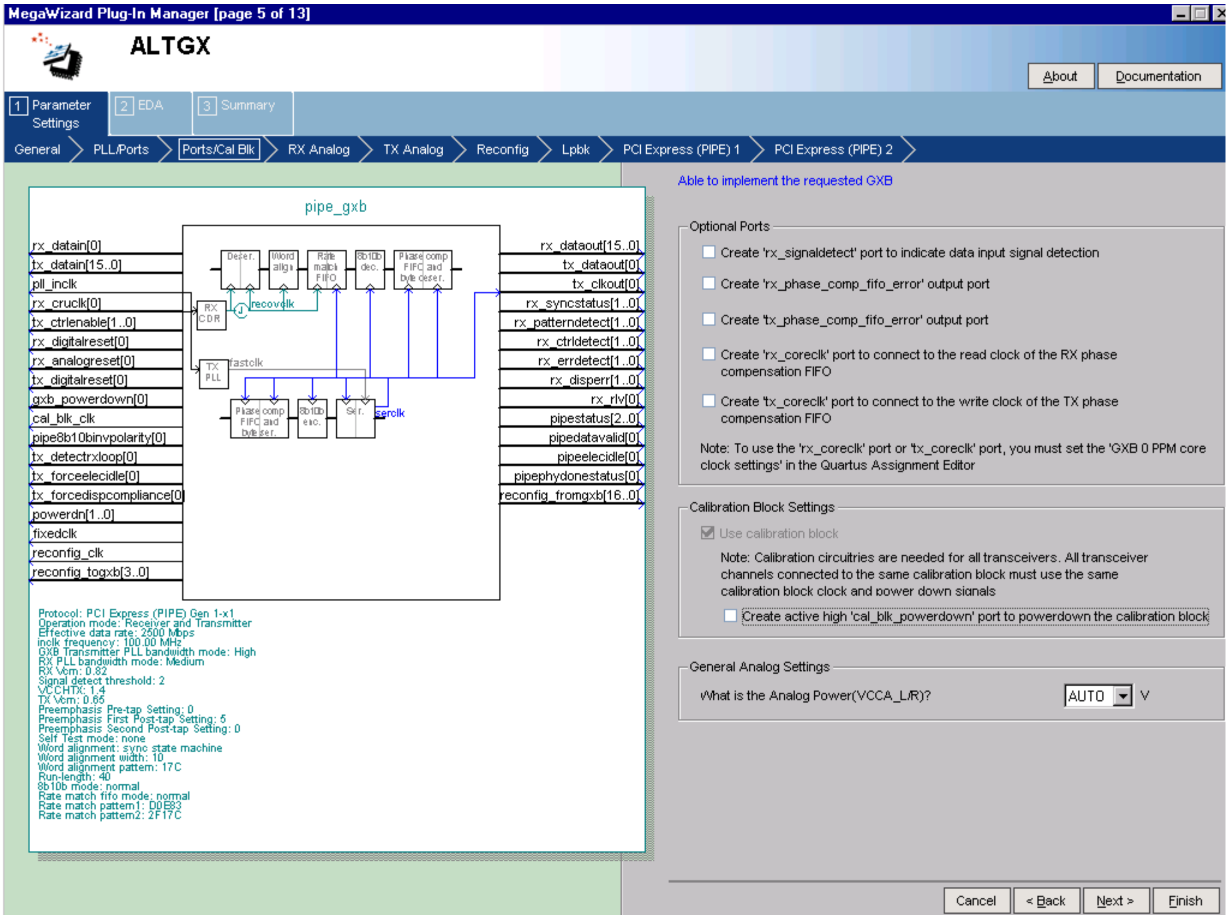
(1) LTR mode is lock-to-reference mode and LTD mode is lock-to-data mode.

(2) When `rx_locktorefclk` and `rx_locktodata` are both asserted high, `rx_locktodata` takes precedence over `rx_locktorefclk`, forcing the CDR to lock to the received data. When both these signals are de-asserted, the LTR/LTD controller is configured in automatic lock mode.

Ports/Cal Blk Screen for PCI Express Mode

Figure 1–17 shows the **Ports/Cal Blk** screen of the ALTGX MegaWizard Plug-In Manager for PCI Express mode.

Figure 1–17. MegaWizard Plug-In Manager - ALTGX (Ports/Cal Blk Screen for PCI Express Mode)



Protocol: PCI Express (PIPE) Gen 1-x1
Operation mode: Receiver and Transmitter
Effective data rate: 2500 Mbps
inclk frequency: 100.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Signal detect threshold: 2
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 5
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
Run-length: 40
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: D0E83
Rate match pattern2: 2F17C

Able to implement the requested GXB

Optional Ports

- □ Create 'rx_signaldetect' port to indicate data input signal detection
- □ Create 'rx_phase_comp_fifo_error' output port
- □ Create 'tx_phase_comp_fifo_error' output port
- □ Create 'rx_coreclk' port to connect to the read clock of the RX phase compensation FIFO
- □ Create 'tx_coreclk' port to connect to the write clock of the TX phase compensation FIFO

Note: To use the 'rx_coreclk' port or 'tx_coreclk' port, you must set the 'GXB 0 PPM core clock settings' in the Quartus Assignment Editor

Calibration Block Settings

☑ Use calibration block

Note: Calibration circuitries are needed for all transceivers. All transceiver channels connected to the same calibration block must use the same calibration block clock and power down signals

□ Create active high 'cal_blk_powerdown' port to powerdown the calibration block

General Analog Settings

What is the Analog Power(VCCA_L/R)? AUTO V

Table 1–13 describes the available options on the **Ports/Cal Blk** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–13. MegaWizard Plug-In Manager Options (Ports/Cal Blk Screen for PCI Express (PIPE) Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Create an `rx_signaldetect` port to indicate data input signal detection.	This option enables the `rx_signaldetect` output port, which indicates if a signal conforms to the signal detecting settings in the signal detect circuit. The signal level circuit senses if the specified voltage level exists at the receiver input buffer. This signal is asynchronous to the receiver data path. This port's functionality depends on the options **Force signal detect** and **What is the signal detect and signal loss threshold?**	*Signal Threshold Detection Circuitry* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_phase_comp_fifo_error` output port.	This optional output port indicates receiver phase compensation FIFO overflow or underrun condition.	*Receiver Phase Compensation FIFO Error Flag* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_phase_comp_fifo_error` output port.	This optional output port indicates transmitter phase compensation FIFO overflow or underrun condition.	*TX Phase Compensation FIFO Status Signal* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_coreclk` port to connect to the read clock of the RX phase compensation FIFO.	The parallel output data from the receiver can be clocked using this optional input port. It allows you to clock the read side of the Receiver Phase Compensation FIFO with a user-provided clock (FPGA fabric clock, FPGA fabric-Transceiver interface clock, or input reference clock).	*Stratix IV Transceiver Clocking* in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_coreclk` port to connect to the write clock of the TX phase compensation FIFO.	The parallel transmitter data generated in the FPGA fabric can be clocked using this optional input port. It allows you to clock the write side of the Transmitter Phase Compensation FIFO with a user-provided clock (FPGA fabric clock, FPGA fabric-Transceiver interface clock, or input reference clock).	*Stratix IV Transceiver Clocking* in volume 2 of the *Stratix IV Device Handbook.*
Use calibration block.	Calibration block is always enabled.	*Calibration Blocks* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

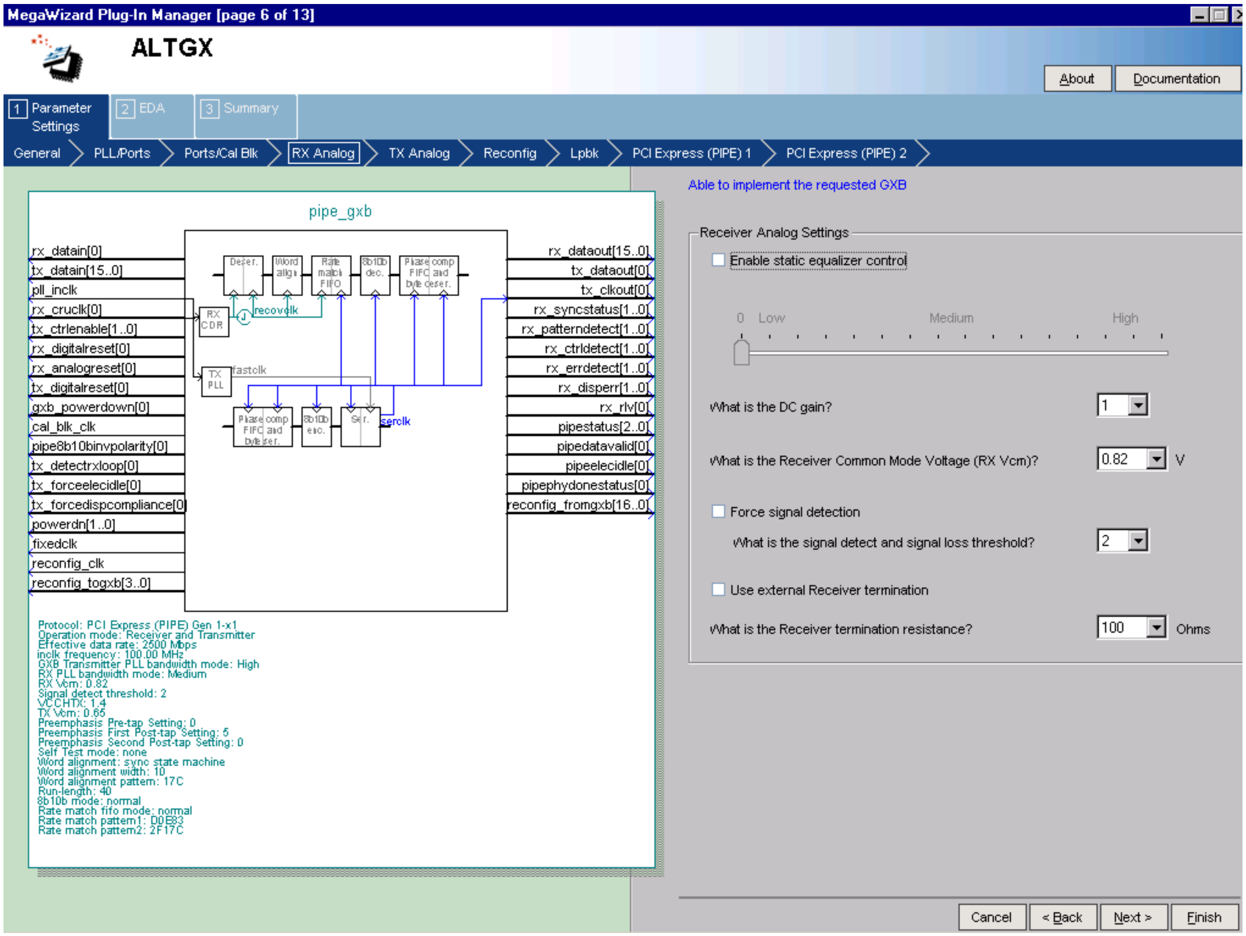
Table 1–13. MegaWizard Plug-In Manager Options (Ports/Cal Blk Screen for PCI Express (PIPE) Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
Create an active high `cal_blk_powerdown` to power down the calibration block.	Asserting this signal high powers down the calibration block. A high-to-low transition on this signal restarts calibration.	*Input Signals to the Calibration Block* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the analog power ($V_{CCA\_L/R}$)?	The options available for selection in PCI Express (PIPE) Gen 1 mode are 2.5 V, 3.0 V, and AUTO. The options available for selection in PCI Express (PIPE) Gen 2 mode are 3.0 V and AUTO. 3.0 V : up to 8.5 Gbps 2.5 V : up to 4.25 Gbps AUTO : The ALTGX MegaWizard Plug-In Manager will automatically set V_{CCA} to 2.5 V and V_{CCH} (transmitter buffer voltage) to 1.5 V for the VCO base data rates less than 4.25 Gbps. -OR- V_{CCA} to 3.0 V and V_{CCH} to 1.4 V for the VCO base data rates greater than 4.25 Gbps.	*General Requirements to Combine Channels* section in the *Configuring Multiple Protocols and Data Rates* chapter in volume 2 of the *Stratix IV Device Handbook.*

RX Analog Screen for PCI Express Mode

Figure 1–18 shows the **RX Analog** screen of the ALTGX MegaWizard Plug-In Manager for PCI Express mode.

Figure 1–18. MegaWizard Plug-In Manager - ALTGX (RX Analog Screen for PCI Express Mode)



Protocol: PCI Express (PIPE) Gen 1-x1
Operation mode: Receiver and Transmitter
Effective data rate: 2500 Mbps
inclk frequency: 100.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Signal detect threshold: 2
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 5
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
Run-length: 40
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: D0E83
Rate match pattern2: 2F17C

Able to implement the requested GXB

Receiver Analog Settings

□ Enable static equalizer control

What is the DC gain? 1

What is the Receiver Common Mode Voltage (RX Vcm)? 0.82 V

□ Force signal detection

What is the signal detect and signal loss threshold? 2

□ Use external Receiver termination

What is the Receiver termination resistance? 100 Ohms

Table 1–14 describes the available options on the **RX Analog** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

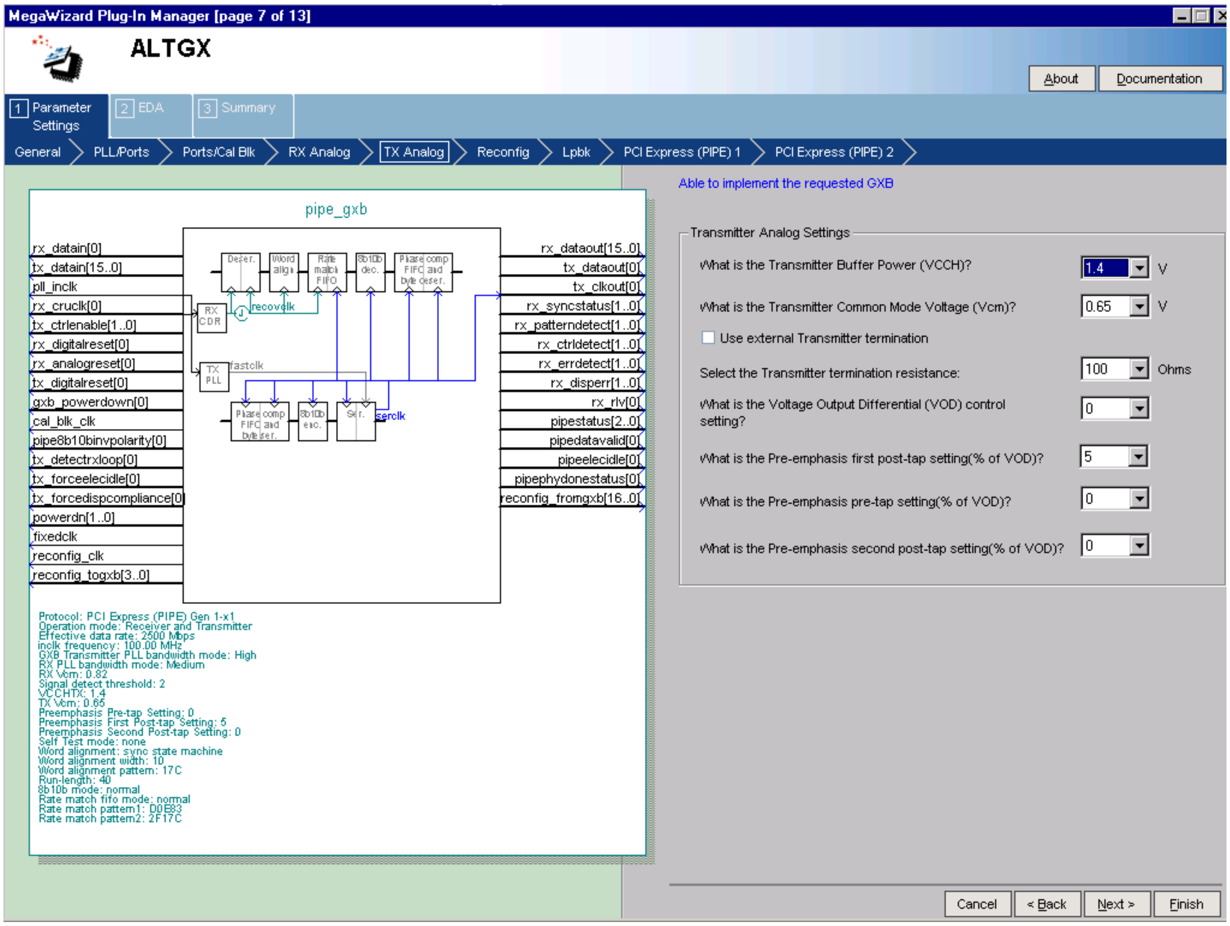
Table 1–14. MegaWizard Plug-In Manager Options (RX Analog Screen for PCI Express Mode

ALTGX Setting	Description	Reference
Enable static equalizer control.	This option enables the static equalizer settings.	*Programmable Equalization and DC Gain* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the DC gain?	This DC gain option is set to 1 in the PCI Express (PIPE) mode.	*Programmable Equalization and DC Gain* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the receiver common mode voltage (RX V_{CM})?	The receiver common mode voltage is set to 0.82 V.	*Programmable Common Mode Voltage* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Force signal detection.	This option disables the signal threshold detect circuit. The receiver CDR will no longer depend on the signal detect criterion to switch from lock-to-reference to lock-to-data mode. If you have selected the input port `rx_signaldetect` and enabled this option, the `rx_signaldetect` will remain high.	*Signal Threshold Detection Circuitry* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the signal detect and signal loss threshold?	Use this option when the forced signal detection option is not enabled, to set the trip point of the signal threshold detect circuit. The levels are to be determined after characterization.	*Signal Threshold Detection Circuitry* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Use external receiver termination.	This option is available if you want to use an external termination resistor instead of the differential on-chip termination (OCT). If checked, this option turns off the receiver OCT.	*Programmable Differential On-Chip Termination* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the receiver termination resistance?	This option selects the receiver differential termination value. The settings allowed are 85 Ω, 100 Ω, 120 Ω, and 150 Ω.	*Programmable Differential On-Chip Termination* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

TX Analog Screen for PCI Express Mode

Figure 1–19 shows the **TX Analog** screen of the ALTGX MegaWizard Plug-In Manager for PCI Express mode.

Figure 1–19. MegaWizard Plug-In Manager - ALTGX (TX Analog Screen for PCI Express Mode)



Protocol: PCI Express (PIPE) Gen 1-x1
Operation mode: Receiver and Transmitter
Effective data rate: 2500 Mbps
inclk frequency: 100.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Signal detect threshold: 2
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 5
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
Run-length: 40
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: D0E83
Rate match pattern2: 2F17C

Able to implement the requested GXB

Transmitter Analog Settings

What is the Transmitter Buffer Power (VCCH)? 1.4 V

What is the Transmitter Common Mode Voltage (Vcm)? 0.65 V

□ Use external Transmitter termination

Select the Transmitter termination resistance: 100 Ohms

What is the Voltage Output Differential (VOD) control setting? 0

What is the Pre-emphasis first post-tap setting(% of VOD)? 5

What is the Pre-emphasis pre-tap setting(% of VOD)? 0

What is the Pre-emphasis second post-tap setting(% of VOD)? 0

Table 1–15 describes the available options on the **TX Analog** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–15. **MegaWizard Plug-In Manager Options (TX Analog Screen for PCI Express (PIPE) Mode) (Part 1 of 2)**

ALTGX Setting	Description	Reference
What is the transmitter buffer power (V_{CCH})?	In PCI Express (PIPE) mode, this value is fixed to 1.4 V. It is up to the user to connect the correct voltage supply to the V_{CCH} pins on the board.	*Programmable Transmit Output Buffer Power* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the transmitter common mode voltage (V_{CM})?	In PCI Express (PIPE) mode, the transmitter common mode voltage is fixed to 0.7 V.	*Common Mode Voltage (V_{CM}) Settings* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–15. **MegaWizard Plug-In Manager Options (TX Analog Screen for PCI Express (PIPE) Mode) (Part 2 of 2)**

ALTGX Setting	Description	Reference
Use external transmitter termination.	This option is available if you want to use an external termination resistor instead of the differential on-chip termination (OCT). Checking this option turns off the transmitter differential OCT.	*Programmable Transmitter Termination* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Select the transmitter termination resistance.	This option selects the transmitter differential termination value. The settings allowed are 85 Ω, 100 Ω, 120 Ω, and 150 Ω.	*Programmable Transmitter Termination* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the voltage output differential (VOD) control setting?	This option selects the VOD of the transmitter buffer. The available VOD settings change based on the transmitter termination resistance value.	*Programmable Output Differential Voltage* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the pre-emphasis pre-tap setting (% of VOD)?	This option sets the amount of pre-emphasis on the transmitter buffer using pre-tap.	*Programmable Pre-Emphasis* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the pre-emphasis first post-tap setting (% of VOD)?	This option sets the amount of pre-emphasis on the transmitter buffer using first post-tap.	*Programmable Pre-Emphasis* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the pre-emphasis second post-tap setting (% of VOD)?	This option sets the amount of pre-emphasis on the transmitter buffer using second post-tap.	*Programmable Pre-Emphasis* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

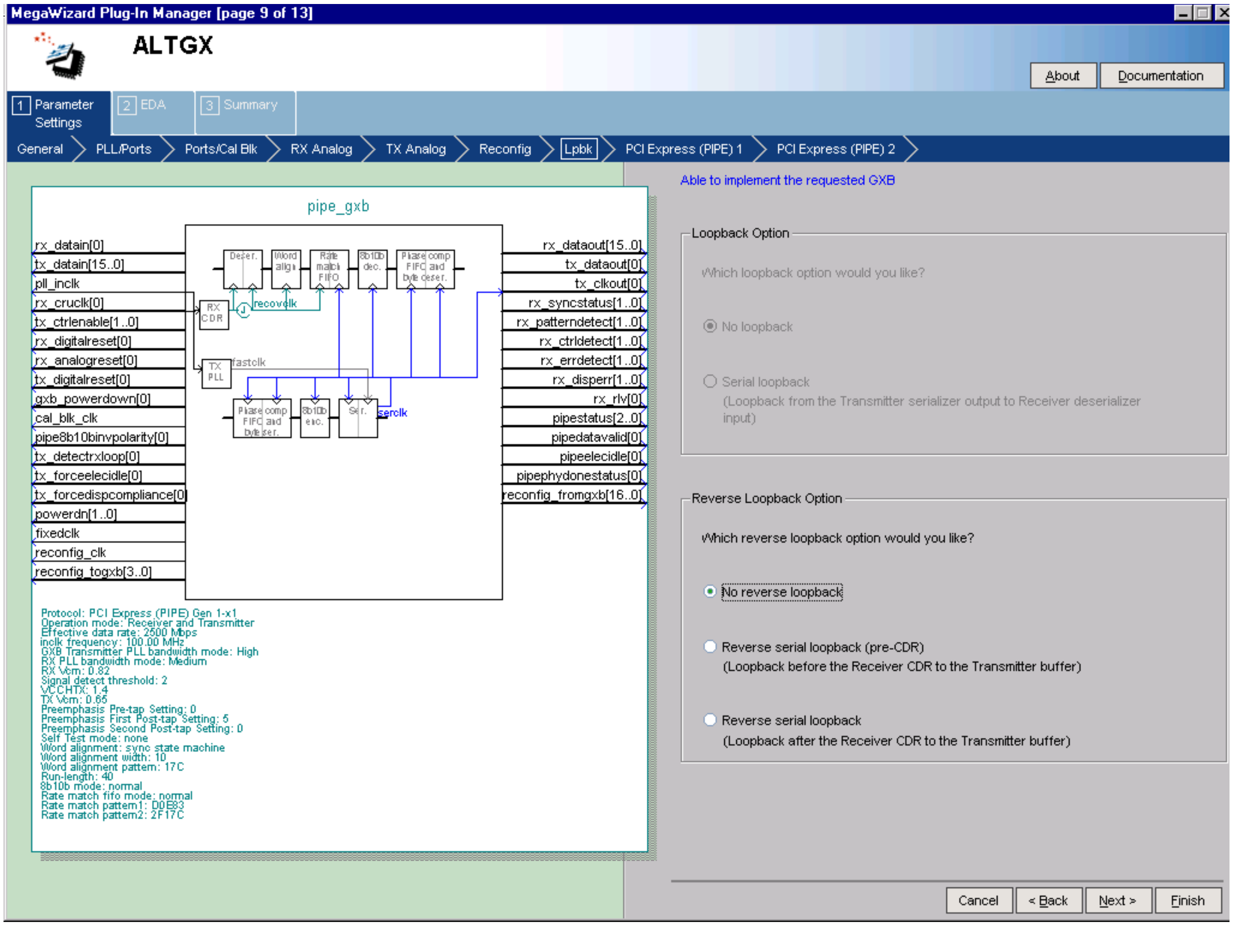
Reconfig Screen for PCI Express Mode

Figure 1–8 shows the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager for the PCI Express (PIPE) mode. Table 1–6 explains each of the parameter settings available for the **Reconfig** screen.

Lpbk Screen for PCI Express Mode

Figure 1–20 displays the **Lpbk** screen of the ALTGX MegaWizard Plug-In Manager.

Figure 1–20. MegaWizard Plug-In Manager - Lpbk Screen for PCI Express Mode



Protocol: PCI Express (PIPE) Gen 1-x1
Operation mode: Receiver and Transmitter
Effective data rate: 2500 Mbps
inclk frequency: 100.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Signal detect threshold: 2
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 5
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
Run-length: 40
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: D0B83
Rate match pattern2: 2F17C

Able to implement the requested GXB

Loopback Option

Which loopback option would you like?

● No loopback

○ Serial loopback
(Loopback from the Transmitter serializer output to Receiver deserializer input)

Reverse Loopback Option

Which reverse loopback option would you like?

● No reverse loopback

○ Reverse serial loopback (pre-CDR)
(Loopback before the Receiver CDR to the Transmitter buffer)

○ Reverse serial loopback
(Loopback after the Receiver CDR to the Transmitter buffer)

Table 1–16 shows information about the **Lpbk** screen for PCI Express mode.

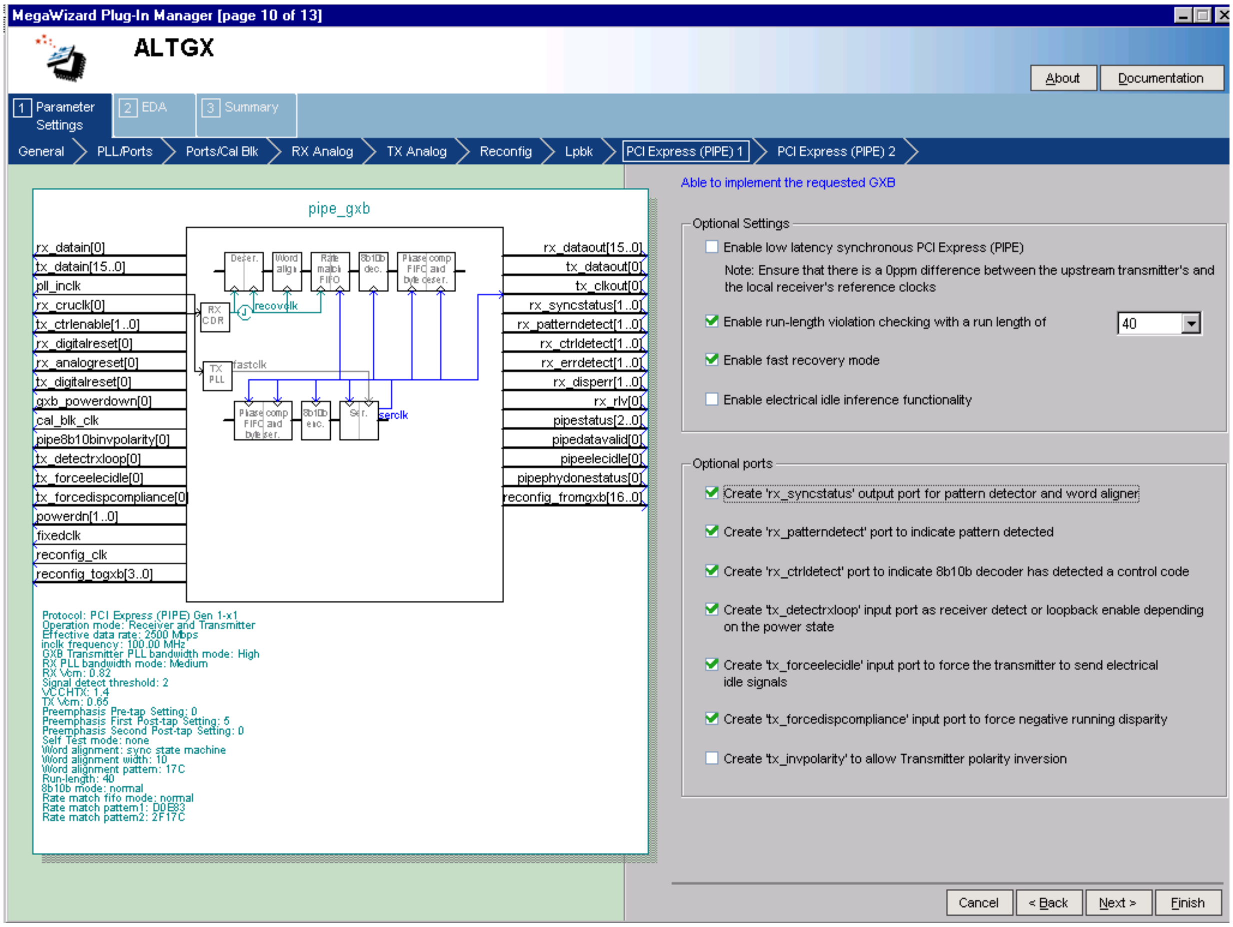
Table 1–16. MegaWizard Plug-In Manager Options (Lpbk Screen for PCI Express Mode)

ALTGX Setting	Description	Reference
Which loopback option would you like?	This option is not available for PCI Express (PIPE) mode.	—
Which reverse loopback option would you like?	There are three options available in Basic mode: ■ No reverse loopback — This is the default mode. ■ Reverse Serial loopback (pre-CDR) — This is the loopback before the receiver's CDR block to the transmitter buffer. The receiver path in the PCS is active but the transmitter side is not. ■ Reverse Serial loopback — This is a loopback after the receiver's CDR block to the transmitter buffer. The receiver path in the PCS is active but the transmitter side is not.	*Loopback Modes* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

PCI Express (PIPE) 1 Screen

Figure 1–21 shows the **PCI Express (PIPE) 1** screen of the MegaWizard Plug-In Manager for the PCI Express (PIPE) mode selection. If the **Enforce default settings for this protocol** option is selected, this page does not appear in the MegaWizard Plug-In Manager.

Figure 1–21. MegaWizard Plug-In Manager - ALTGX (PCI Express (PIPE) 1 Screen)



Protocol: PCI Express (PIPE) Gen 1-x1
Operation mode: Receiver and Transmitter
Effective data rate: 2500 Mbps
inclk frequency: 100.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Signal detect threshold: 2
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 5
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
Run-length: 40
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: D0E83
Rate match pattern2: 2F17C

Able to implement the requested GXB

Optional Settings

□ Enable low latency synchronous PCI Express (PIPE)

Note: Ensure that there is a 0ppm difference between the upstream transmitter's and the local receiver's reference clocks

☑ Enable run-length violation checking with a run length of 40

☑ Enable fast recovery mode

□ Enable electrical idle inference functionality

Optional ports

☑ Create 'rx_syncstatus' output port for pattern detector and word aligner

☑ Create 'rx_patterndetect' port to indicate pattern detected

☑ Create 'rx_ctrldetect' port to indicate 8b10b decoder has detected a control code

☑ Create 'tx_detectrxloop' input port as receiver detect or loopback enable depending on the power state

☑ Create 'tx_forceelecidle' input port to force the transmitter to send electrical idle signals

☑ Create 'tx_forcedispcompliance' input port to force negative running disparity

□ Create 'tx_invpolarity' to allow Transmitter polarity inversion

Table 1–17 describes the available options on the **PCI Express (PIPE) 1** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–17. MegaWizard Plug-In Manager Options (PCI Express (PIPE) 1 Screen for PCI Express (PIPE) Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
Enable low latency synchronous PCI Express (PIPE).	This option puts the rate match FIFO into a low latency mode which forces the system into a 0 ppm mode. Ensure that there is a 0 ppm difference between the upstream transmitter's and the local receiver's input reference clocks.	*Low-Latency PCI Express (PIPE) Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable run-length violation checking with a run length of.	This option creates the output signal `rx_rlv`. Enabling this option also activates the run-length violation circuit. If the number of continuous 1's and 0's exceeds the number that you set in this option, the run-length violation circuit asserts the `rx_rlv` signal. The `rx_rlv` signal is asynchronous to the receiver data path. For both 8-bit and 16-bit channel widths, the run length limits are 5 to 160 in increments of 5.	*Programmable Run Length Violation Detection* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable fast recovery mode.	This option enables the CDR control block. When this block is enabled, the `rx_locktodata` and `rx_locktorefclk` signals are disabled.	*Fast Recovery Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable electrical idle inference functionality.	You can enable the electrical idle inference module by selecting this option. In the PCI express (PIPE) mode, the PCS has an optional electrical idle inference module designed to implement the electrical idle inference conditions specified in PCI express base specification 2.0. Enabling this option creates the `rx_elecidleinfersel[2:0]` input signal. The electrical idle Inference module infers electrical idle depending on the logic level driven on `rx_elecidleinfersel[2:0]` input signal. For the electrical idle Inference module to correctly infer an electrical idle condition in each LTSSM sub-state, you must drive the `rx_elecidleinfersel[2:0]` signal appropriately.	*Electrical Idle Inference* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_syncstatus` output port for pattern detector and word aligner.	The ALTGX MegaWizard Plug-In Manager automatically configures the word aligner in automatic synchronization state machine mode for PCI express (PIPE) mode. This is an output status signal that the word aligner forwards to the FPGA fabric to indicate that synchronization has been achieved. This signal is synchronous with the parallel receiver data on the `rx_dataout` port. The signal width is 1 and 2 bits for a channel width of 8-bits and 16-bits, respectively.	*Table 1-24 and Automatic Synchronization State Machine Mode Word Aligner in 10-bit PMA-PCS Interface Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–17. MegaWizard Plug-In Manager Options (PCI Express (PIPE) 1 Screen for PCI Express (PIPE) Mode) (Part 2 of 3)

ALTGX Setting	Description	Reference
Create an `rx_patterndetect` output port to indicate pattern detected.	This is an output status signal that the word aligner forwards to the FPGA fabric to indicate that the word alignment pattern programmed has been detected in the current word boundary. The signal width is 1 and 2 bits for a channel width of 8-bits and 16-bits, respectively.	*Table 1-24* and *Automatic Synchronization State Machine Mode Word Aligner in 10-bit PMA-PCS Interface Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_ctrldetect` port to indicate 8B/10B decoder has detected a control code.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric. This signal indicates whether the decoded 8-bit code group is a data or control code group on this port. If the received 10-bit code group is one of the 12 control code groups (/Kx.y/) specified in IEEE802.3 specification, this signal is driven high. If the received 10-bit code group is a data code group (/Dx.y/), this signal is driven low. The signal width is 1 and 2 bits for a channel width of 8-bits and 16-bits, respectively.	*8B/10B Decoder* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_detectrxloopback` input port as Receiver detect or loopback enable, depending on the power state.	Depending on the power-down mode, asserting this signal enables either the receiver detect operation or the loopback mode. *(1)*	*Receiver Detection* section and the *PCI Express (PIPE) Reverse Parallel Loopback* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_forceelecidle` input port to force the Transmitter to send Electrical Idle signals.	Enabling this port sets the transmitter buffer in electrical idle mode. This port is available in all PCI express power-down modes and has a specific use in each mode. *(1)*	*Transmitter Buffer Electrical Idle* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_forcedispcompliance` input port to force negative running disparity.	A high level on this port forces the associated parallel transmitter data on the `tx_datain` port to be transmitted with negative current running disparity. ■ For 8-bit transceiver channel width configurations, you must drive `tx_forcedispcompliance[1:0]` high in the same parallel clock cycle as the first /K28.5/ of the compliance pattern on `tx_datain` port. ■ For 16-bit transceiver channel width configurations, you must drive only the LSB of the `tx_forcedispcompliance[1:0]`high in the same parallel clock cycle as /K28.5/D21.5/ of the compliance pattern on `tx_datain` port.	*Compliance Pattern Transmitter Support* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–17. MegaWizard Plug-In Manager Options (PCI Express (PIPE) 1 Screen for PCI Express (PIPE) Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
Create a `tx_invpolarity` port to allow Transmitter polarity inversion.	This optional port allows you to dynamically reverse the polarity of every bit of the data word fed to the serializer in the transmitter data path. Use this option when the positive and negative signals of the differential output from the transmitter (`tx_dataout`) are erroneously swapped on the board.	*Transmitter Polarity Inversion* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

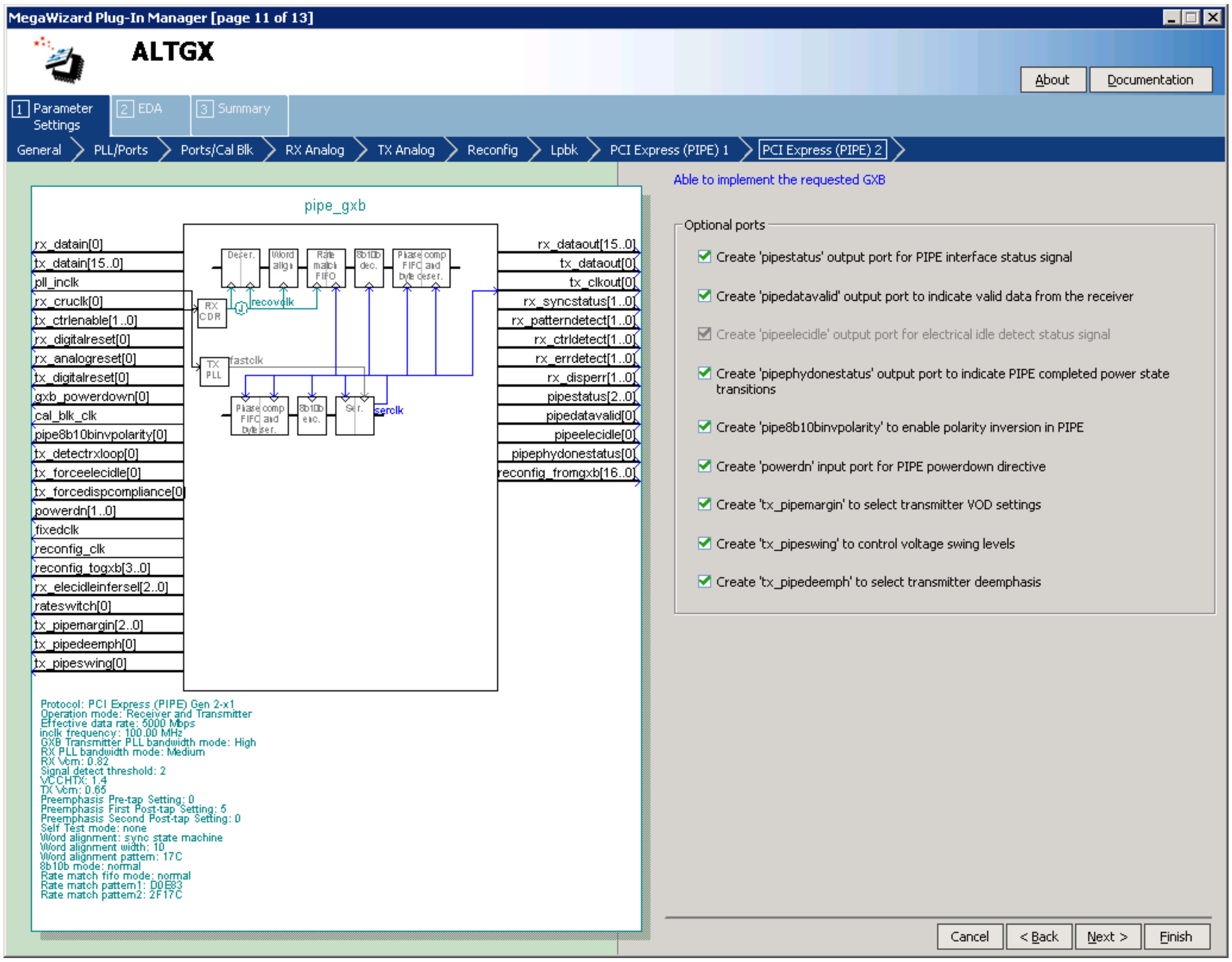
Note to Table 1–17:

(1) Refer to the table '*Power States and Functions Allowed in Each Power State*' in the *PIPE Interface* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

PCI Express (PIPE) 2 Screen

Figure 1–21 shows the **PCI Express (PIPE) 2** screen of the MegaWizard Plug-In Manager for the PCI Express (PIPE) mode selection. If the **Enforce default settings for this protocol** option is selected, this page does not appear in the MegaWizard Plug-In Manager.

Figure 1–22. MegaWizard Plug-In Manager - ALTGX (PCI Express (PIPE) 2 Screen)



Protocol: PCI Express (PIPE) Gen 2-x1
Operation mode: Receiver and Transmitter
Effective data rate: 5000 Mbps
inclk frequency: 100.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Signal detect threshold: 2
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 5
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: D0E83
Rate match pattern2: 2F17C

Able to implement the requested GXB

Optional ports

- ☑ Create 'pipestatus' output port for PIPE interface status signal
- ☑ Create 'pipedatavalid' output port to indicate valid data from the receiver
- ☑ Create 'pipeelecidle' output port for electrical idle detect status signal
- ☑ Create 'pipephydonestatus' output port to indicate PIPE completed power state transitions
- ☑ Create 'pipe8b10binvpolarity' to enable polarity inversion in PIPE
- ☑ Create 'powerdn' input port for PIPE powerdown directive
- ☑ Create 'tx_pipemargin' to select transmitter VOD settings
- ☑ Create 'tx_pipeswing' to control voltage swing levels
- ☑ Create 'tx_pipedeemph' to select transmitter deemphasis

Table 1–18 describes the available options on the **PCI Express (PIPE) 2** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–18. MegaWizard Plug-In Manager Options (PCI Express (PIPE) 2 Screen for PCI Express (PIPE) Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Create a `pipestatus` output port for PIPE interface status signal.	The PIPE interface block receives status signals from the transceiver channel PCS and PMA blocks and encodes the status on a 3-bit output signal `pipestatus[2:0]` which is forwarded to the FPGA fabric. The encoding of the status signals on `pipestatus[2:0]` is compliant to the PIPE v2.00 specification: ■ 000 - Received data OK ■ 001 - One SKP symbol added ■ 010 - One SKP symbol deleted ■ 011 - Receiver detected ■ 100 - 8B/10B decode error ■ 101 - Elastic buffer (rate match FIFO) overflow ■ 110 - Elastic buffer (rate match FIFO) underflow ■ 111 - Received disparity error	*Receiver Status* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `pipedatavalid` output port to indicate valid data from the receiver.	This is an output status port which indicates that the receiver parallel data on the `rx_dataout` port is valid.	—
Create a `pipeelecidle` output port for Electrical Idle detect status signal.	Enabling this option creates the `pipeelecidle` output status port which is forwarded to the FPGA fabric. ■ If you select **Enable Electrical Idle Inference Module,** then the `pipeelecidle` signal is driven high when the electrical idle inference module infers an electrical idle condition depending on the logic driven on the `rx_elecidleinfersel[2:0]` port. Otherwise, it is driven low. ■ If you do not select **Enable Electrical Idle Inference Module**, then the `rx_signaldetect` output signal from the signal threshold detection circuitry is inverted and driven on the `pipeelecidle` port. The `pipeelecidle` signal is asynchronous to the receiver data path.	*Electrical Idle Inference* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `pipephydonestatus` output port to indicate PIPE completed power state transitions.	This is an output status signal forwarded to the FPGA fabric. The completion of various PHY functions like receiver detection, power state transition, clock switch, and rate switch are indicated on this `pipephydonestatus` signal by driving this signal high for one parallel clock cycle.	*PCI Express (PIPE) Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–18. MegaWizard Plug-In Manager Options (PCI Express (PIPE) 2 Screen for PCI Express (PIPE) Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
Create a `pipe8b/10binvpolarity` port to enable polarity inversion in PIPE.	This optional port allows you to dynamically reverse every bit of the received data at the input of the 8B/10B decoder.	*PCI Express (PIPE) Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `powerdn` input port for PIPE powerdown directive.	Enabling this option creates an input control port `powerdn[1:0]` for each transceiver channel. Depending on the logic levels driven on this port, the PIPE Interface block drives the transceiver channel into one of the power states as follows: 00 – P0 (Normal operation) 01 – P0s (low recovery time power saving 10 – P1 (high recovery time power saving) 11 – P2 (lowest power saving state)	*Power State Management* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_pipemargin` input to select the transmitter VOD level.	This option creates a `tx_pipemargin[2:0]` input port which is available only in PCI Express (PIPE) Gen 2 (5 Gbps) mode. It selects the transmitter VOD level: ■ 3'b000: Normal operating range ■ 3'b001: Full swing: 800 – 1200 mV Half swing: 400 – 700 mV ■ 3'b010: To be determined ■ 3'b011: To be determined ■ 3'b100: Full swing: 200 – 400 mV Half swing: 100 – 200 mV ■ 3'b101: Full swing: 200 – 400 mV Half swing: 100 – 200 mV ■ 3'b110: Full swing: 200 – 400 mV Half swing: 100 – 200 mV ■ 3'b111: Full swing: 200 – 400 mV Half swing: 100 – 200 mV	—
Create a `tx_pipeswing` input to select the transmitter voltage swing level.	This option creates a `tx_pipeswing` input port which is available only in PCI Express (PIPE) Gen 2 (5 Gbps) mode. It controls the transmitter voltage swing level. ■ 1'b0: - full swing ■ 1'b1: - half swing	—
Create a `tx_pipedeemph` input to select the transmitter deemphasis.	This option creates a `tx_pipedeemph` input port which is available only in PCI Express (PIPE) Gen 2 (5 Gbps) mode. It selects the transmitter deemphasis: ■ 1'b0: -6 dB ■ 1'b1: -3.5 dB	—

EDA Screen for PCI Express Mode

Figure 1–13 describes the **EDA** screen of the MegaWizard Plug-In Manager for the PCI Express mode. The **Generate Netlist** option generates a netlist for the third party EDA synthesis tool to estimate timing and resource utilization for the ALTGX instance.

Summary for PCI Express Mode

Figure 1–14 describes the **Summary** screen of the MegaWizard Plug-In Manager for the PCI Express mode. You can select optional files on this page. After you make your selections, click **Finish** to generate the files.

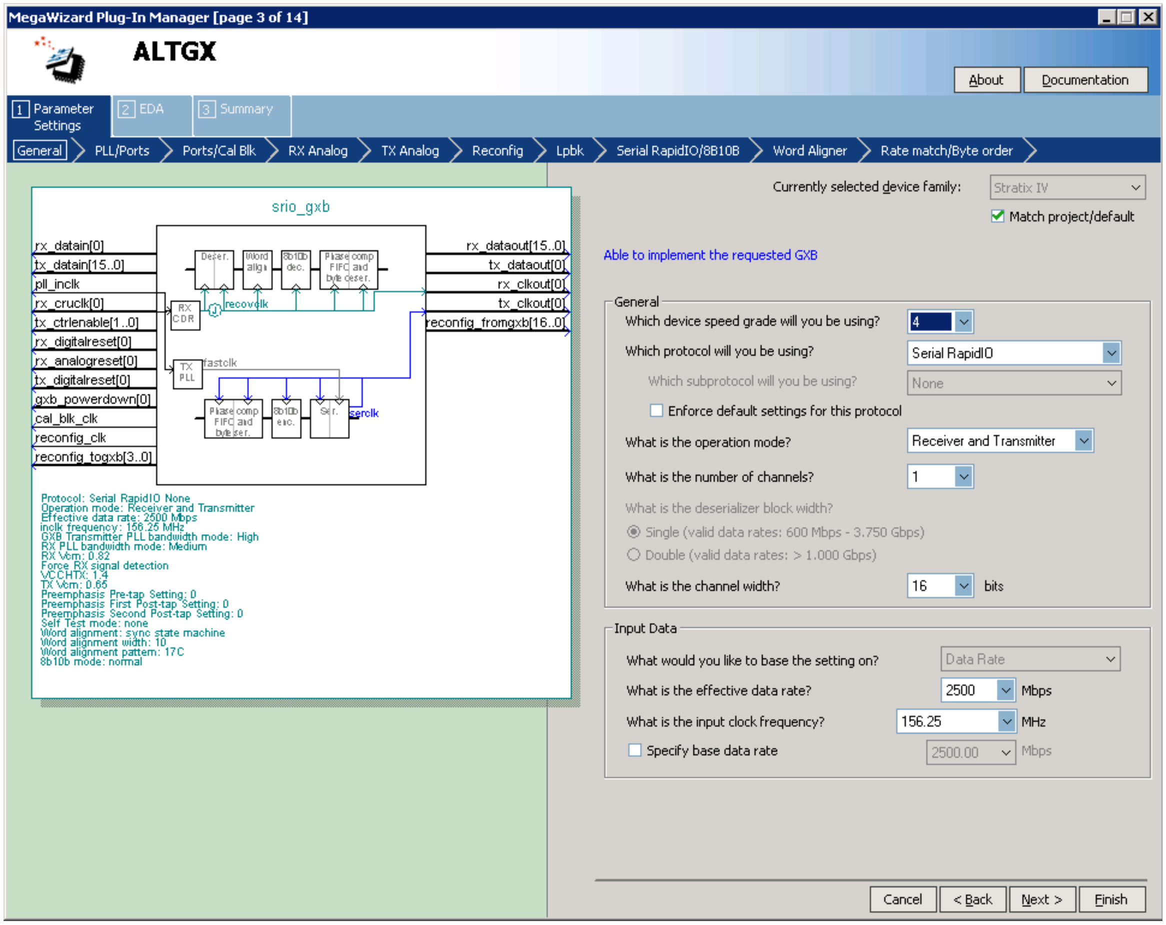
Serial RapidIO Mode

This section provides descriptions of the options available on the individual pages of the ALTGX MegaWizard Plug-In Manager for Serial RapidIO mode. The MegaWizard Plug-In Manager provides a warning if any of the settings you choose are illegal.

General Screen for Serial RapidIO Mode

Figure 1–23 shows the **General** screen of the ALTGX MegaWizard Plug-In Manager for Serial RapidIO mode.

Figure 1–23. MegaWizard Plug-In Manager - ALTGX (General Screen for Serial RapidIO Mode)



Protocol: Serial RapidIO None
Operation mode: Receiver and Transmitter
Effective data rate: 2500 Mbps
inclk frequency: 156.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Currently selected device family: Stratix IV
☑ Match project/default

Able to implement the requested GXB

General
Which device speed grade will you be using? 4
Which protocol will you be using? Serial RapidIO
Which subprotocol will you be using? None
☐ Enforce default settings for this protocol
What is the operation mode? Receiver and Transmitter
What is the number of channels? 1
What is the deserializer block width?
● Single (valid data rates: 600 Mbps - 3.750 Gbps)
○ Double (valid data rates: > 1.000 Gbps)
What is the channel width? 16 bits

Input Data
What would you like to base the setting on? Data Rate
What is the effective data rate? 2500 Mbps
What is the input clock frequency? 156.25 MHz
☐ Specify base data rate 2500.00 Mbps

Table 1–19 describes the available options on the **General** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–19. MegaWizard Plug-In Manager Options (General Screen for Serial RapidIO Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Which device speed grade will you be using?	Select the speed grade of the device you will be using. The available speed grades are 2, 2x, 3, and 4.	*Table 1-18* in the *DC and Switching Characteristics of the Stratix IV Device Family* chapter in volume 5 of the *Stratix IV Device Handbook*.
Which protocol will you be using?	Determines the specific functional mode under which the transceiver operates. For Serial RapidIO mode, you must select the **Serial RapidIO** option.	*Serial RapidIO Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Which subprotocol will you be using?	This option is not available in Serial RapidIO mode.	—
Enforce default settings for this protocol.	This option is not available in Serial RapidIO mode.	—

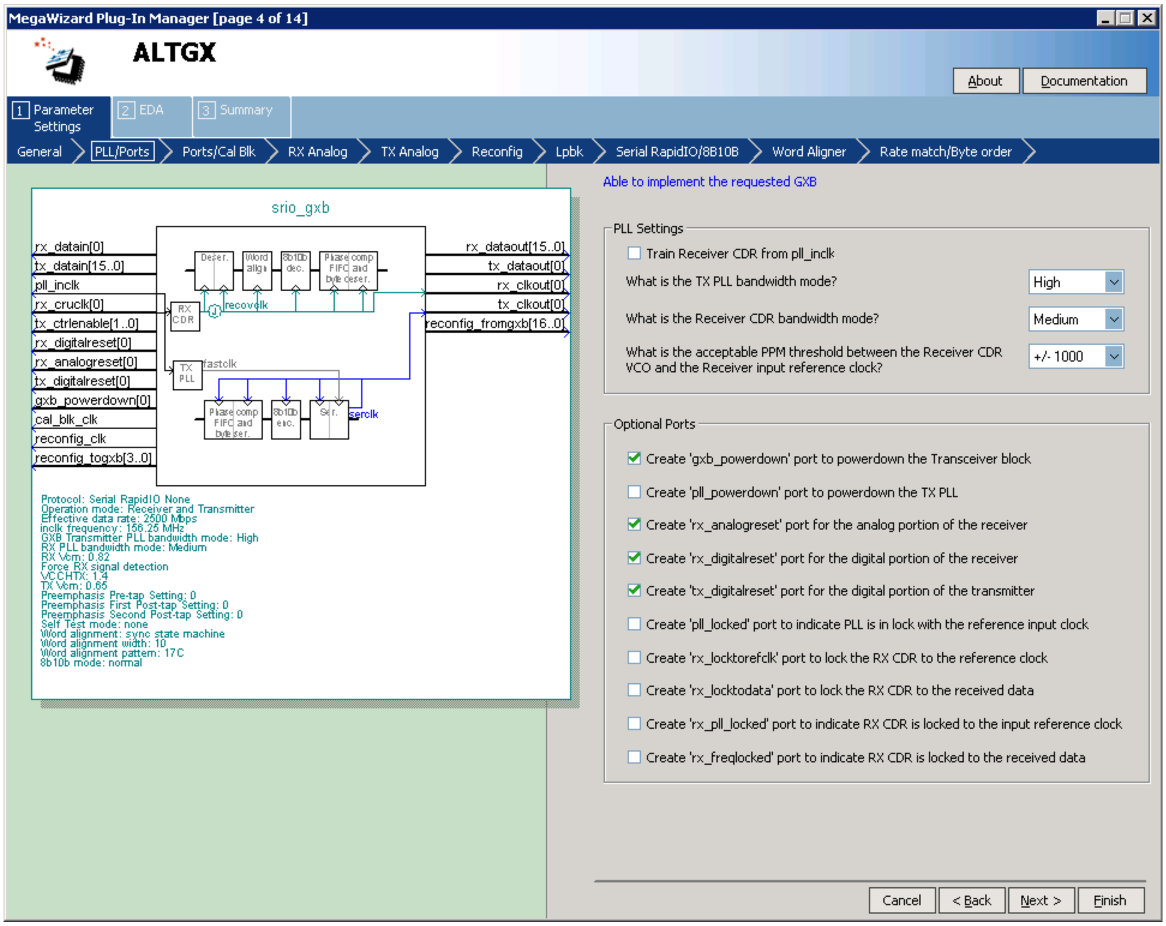
Table 1–19. MegaWizard Plug-In Manager Options (General Screen for Serial RapidIO Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
What is the operation mode?	The available operation modes are **Receiver only**, **Transmitter only**, and **Receiver and Transmitter**.	—
What is the number of channels?	It is the number of channels required with the same configuration. This option determines how many identical channels this ALTGX instance contains.	—
What is the deserializer block width?	Serial RapidIO mode only operates in a single-width mode. Double-width mode is not allowed.	—
What is the channel width?	The channel width is fixed to 16 in Serial RapidIO mode.	*Byte Serializer* and *Deserializer* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What would you like to base the setting on?	This option is not available for selection in Serial RapidIO mode.	—
What is the effective data rate?	The allowed data rates are 1250 Mbps, 2500 Mbps, and 3125 Mbps in Serial RapidIO mode. Enter one of these three data rates in this option.	—
What is the input clock frequency?	This option provides the available input reference clock frequencies depending on whether your effective serial data rate is 1250 Mbps, 2500 Mbps, or 3125 Mbps and the available multiplier settings.	*CMU PLL and Receiver CDR Input Reference Clock* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Specify base data rate.	This option is not available for selection in this mode. The ALTGX MegaWizard Plug-In Manager provides you the base data rate options for the VCO in the CMU PLL and receiver CDR in this option.	—

PLL/Ports for Serial RapidIO Mode

Figure 1–24 shows the **PLL/Ports** screen of the ALTGX MegaWizard Plug-In Manager for Serial RapidIO mode.

Figure 1–24. MegaWizard Plug-In Manager - ALTGX (PLL/Ports Screen)



Protocol: Serial RapidIO None
Operation mode: Receiver and Transmitter
Effective data rate: 2500 Mbps
inclk frequency: 156.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

PLL Settings

- ☐ Train Receiver CDR from pll_inclk
- What is the TX PLL bandwidth mode? High
- What is the Receiver CDR bandwidth mode? Medium
- What is the acceptable PPM threshold between the Receiver CDR VCO and the Receiver input reference clock? +/- 1000

Optional Ports

- ☑ Create 'gxb_powerdown' port to powerdown the Transceiver block
- ☐ Create 'pll_powerdown' port to powerdown the TX PLL
- ☑ Create 'rx_analogreset' port for the analog portion of the receiver
- ☑ Create 'rx_digitalreset' port for the digital portion of the receiver
- ☑ Create 'tx_digitalreset' port for the digital portion of the transmitter
- ☐ Create 'pll_locked' port to indicate PLL is in lock with the reference input clock
- ☐ Create 'rx_locktorefclk' port to lock the RX CDR to the reference clock
- ☐ Create 'rx_locktodata' port to lock the RX CDR to the received data
- ☐ Create 'rx_pll_locked' port to indicate RX CDR is locked to the input reference clock
- ☐ Create 'rx_freqlocked' port to indicate RX CDR is locked to the received data

Table 1–20 describes the available options on the **PLL/ports** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–20. MegaWizard Plug-In Manager Options (PLL/Ports Screen for Serial RapidIO Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
Train receiver CDR from `pll_inclk`.	If you select this option, the input reference clock to the CMU PLL trains the receiver CDR.	*Table 1-2* in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
What is the TX PLL bandwidth mode?	The appropriate bandwidth settings will be fixed after characterization.	*PLL Bandwidth Setting* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
What is the receiver CDR bandwidth mode?	The appropriate bandwidth settings will be fixed after characterization.	—

Table 1–20. MegaWizard Plug-In Manager Options (PLL/Ports Screen for Serial RapidIO Mode) (Part 2 of 3)

ALTGX Setting	Description	Reference
What is the acceptable PPM threshold between the receiver CDR VCO and the receiver input reference clock?	In automatic lock mode, the CDR remains in LTD mode as long as the PPM difference between the CDR VCO output clock and the input reference clock is less than the PPM value that you set in this option. If the PPM difference is greater than the PPM value that you set in this option, the CDR switches to LTR mode. The range of values available in this option is ±62.5 ppm to ±1000 ppm. *(1)*	*Automatic Lock Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `gxb_powerdown` port to power down the transceiver block.	This is an optional signal. When asserted, it powers down the entire transceiver block. If none of the channels are instantiated in a transceiver block, the Quartus II software automatically powers down the entire transceiver block.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `pll_powerdown` port to power down the TX PLL.	Each transceiver block has two CMU PLLs. Each CMU PLL has a dedicated power down signal called `pll_powerdown`. This signal powers down the CMU PLL.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_analogreset` port for the analog portion of the receiver.	Receiver analog reset port available in **Receiver only** and **Receiver and Transmitter** operation modes. Resets part of the analog portion of the receiver CDR in the receiver channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_digitalreset` port for the digital portion of the receiver.	Receiver digital reset port available in **Receiver only** and **Receiver and Transmitter** operation modes. Resets the PCS portion of the receiver channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_digitalreset` port for the digital portion of the transmitter.	Transmitter digital reset port available in **Transmitter only** and **Receiver and Transmitter** operation modes. Resets the PCS portion of the transmitter channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `pll_locked` port to indicate PLL is in lock with the input reference clock.	Each CMU PLL has a dedicated `pll_locked` signal that is fed to the FPGA fabric to indicate when the PLL is locked to the input reference clock.	*Transceiver Reset Sequences* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_locktorefclk` port to lock the RX CDR to the input reference clock.	This is an optional port. When this signal is asserted high, the LTR/LTD controller forces the receiver CDR to lock to the phase and frequency of the input reference clock. *(1)*, *(2)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `rx_locktodata` port to lock the RX CDR to the received data.	This is an optional port. When this signal is asserted high, the LTR/LTD controller forces the receiver CDR to lock to the received data. *(1)*, *(2)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–20. MegaWizard Plug-In Manager Options (PLL/Ports Screen for Serial RapidIO Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
Create an `rx_pll_locked` port to indicate RX CDR is locked to the input reference clock.	In LTR mode, this signal is asserted high to indicate that the receiver CDR has locked to the phase and frequency of the input reference clock. In LTD mode, this signal has no significance. *(1)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_freqlocked` port to indicate RX CDR is locked to the received data.	This signal gets asserted high to indicate that the receiver CDR has switched from LTR to LTD mode. This signal has relevance only in automatic lock mode and may be required to control the transceiver resets, as discussed in the *User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook. (1)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Notes to Table 1–20:

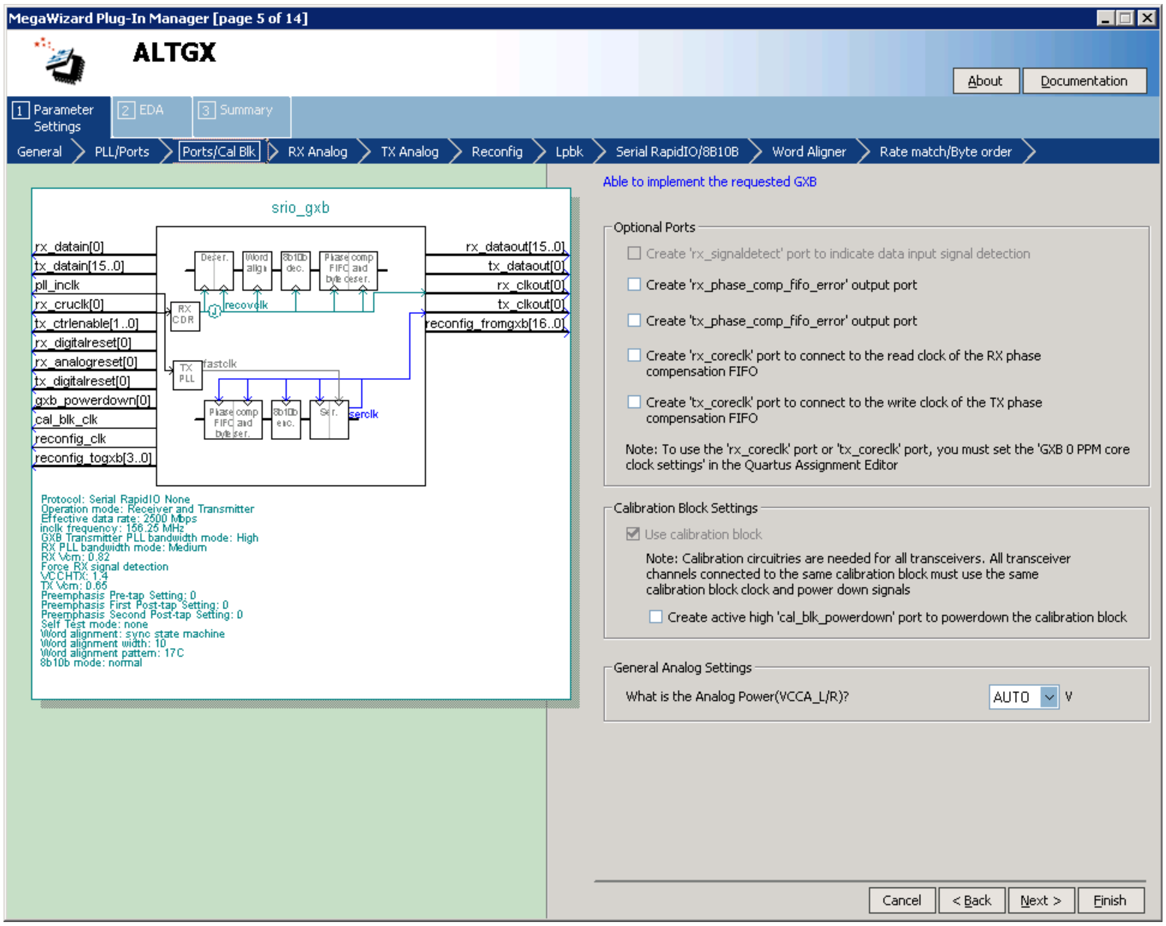
(1) LTR mode is lock-to-reference mode and LTD mode is lock-to-data mode.

(2) When `rx_locktorefclk` and `rx_locktodata` are both asserted high, `rx_locktodata` takes precedence over `rx_locktorefclk`, forcing the CDR to lock to the received data. When both these signals are de-asserted, the LTR/LTD controller is configured in automatic lock mode.

Ports/Cal Blk Screen for Serial RapidIO Mode

Figure 1–25 shows the **Ports/Cal Blk** screen of the ALTGX MegaWizard Plug-In Manager for Serial RapidIO mode.

Figure 1–25. MegaWizard Plug-In Manager - ALTGX (Ports/Cal Blk Screen)



Protocol: Serial RapidIO None
Operation mode: Receiver and Transmitter
Effective data rate: 2500 Mbps
inclk frequency: 156.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Optional Ports

- ☐ Create 'rx_signaldetect' port to indicate data input signal detection
- ☐ Create 'rx_phase_comp_fifo_error' output port
- ☐ Create 'tx_phase_comp_fifo_error' output port
- ☐ Create 'rx_coreclk' port to connect to the read clock of the RX phase compensation FIFO
- ☐ Create 'tx_coreclk' port to connect to the write clock of the TX phase compensation FIFO

Note: To use the 'rx_coreclk' port or 'tx_coreclk' port, you must set the 'GXB 0 PPM core clock settings' in the Quartus Assignment Editor

Calibration Block Settings

- ☑ Use calibration block

Note: Calibration circuitries are needed for all transceivers. All transceiver channels connected to the same calibration block must use the same calibration block clock and power down signals

- ☐ Create active high 'cal_blk_powerdown' port to powerdown the calibration block

General Analog Settings

What is the Analog Power(VCCA_L/R)? AUTO V

Table 1–21 describes the available options on the **Ports/Cal Blk** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–21. MegaWizard Plug-In Manager Options (Ports/Cal Blk Screen for Serial RapidIO Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Create an `rx_signaldetect` port to indicate data input signal detection.	This port is not available in Serial RapidIO mode.	*Signal Threshold Detection Circuitry* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create an `rx_phase_comp_fifo_error` output port.	This optional output port indicates receiver phase compensation FIFO overflow or underrun condition.	*Receiver Phase Compensation FIFO Error Flag* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

Table 1–21. MegaWizard Plug-In Manager Options (Ports/Cal Blk Screen for Serial RapidIO Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
Create a `tx_phase_comp_fifo_error` output port.	This optional output port indicates transmitter phase compensation FIFO overflow or underrun condition.	*TX Phase Compensation FIFO Status Signal* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_coreclk` port to connect to the read clock of the RX phase compensation FIFO.	The parallel output data from the receiver can be clocked using this optional input port. It allows you to clock the read side of the receiver phase compensation FIFO with a user-provided clock (FPGA fabric clock, FPGA fabric-transceiver interface clock, or input reference clock).	*FPGA Fabric-Transceiver Interface Clocking* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_coreclk` port to connect to the write clock of the TX phase compensation FIFO.	The parallel transmitter data generated in the FPGA fabric can be clocked using this optional input port. It allows you to clock the write side of the transmitter phase compensation FIFO with a user-provided clock (FPGA fabric clock, FPGA fabric-transceiver interface clock, or input reference clock).	*FPGA Fabric-Transceiver Interface Clocking* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Use calibration block.	Calibration block is always enabled.	*Calibration Blocks* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an active high `cal_blk_powerdown` to power down the calibration block.	Asserting this signal high powers down the calibration block. A high-to-low transition on this signal restarts calibration.	*Input Signals to the Calibration Block* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the analog power ($V_{CCA\_L/R}$)?	The options available for selection are 2.5 V, 3.0 V and AUTO: 3.0 V : upto 8.5 Gbps 2.5 V : upto 4.25 Gbps AUTO : The ALTGX MegaWizard Plug-In Manager will automatically set V_{CCA} to 2.5 V and V_{CCH} (transmitter buffer voltage) to 1.5 V for the VCO base data rates less than 4.25 Gbps. -OR- V_{CCA} to 3.0 V and V_{CCH} (transmitter buffer voltage) to 1.4 V for the VCO base data rates greater than 4.25 Gbps.	*General Requirements to Combine Channels* section in the *Configuring Multiple Protocols and Data Rates* chapter in volume 2 of the *Stratix IV Device Handbook.*

RX Analog Screen for Serial RapidIO Mode

Figure 1–6 describes the **RX Analog** screen of the ALTGX MegaWizard Plug-In Manager for the Serial RapidIO mode. Table 1–4 explains each of the parameter settings available for the **RX Analog** screen.

TX Analog Screen for Serial RapidIO Mode

Figure 1–7 describes the **TX Analog** screen of the ALTGX MegaWizard Plug-In Manager for the Serial RapidIO mode. Table 1–5 explains each of the parameter settings available for the **TX Analog** screen.

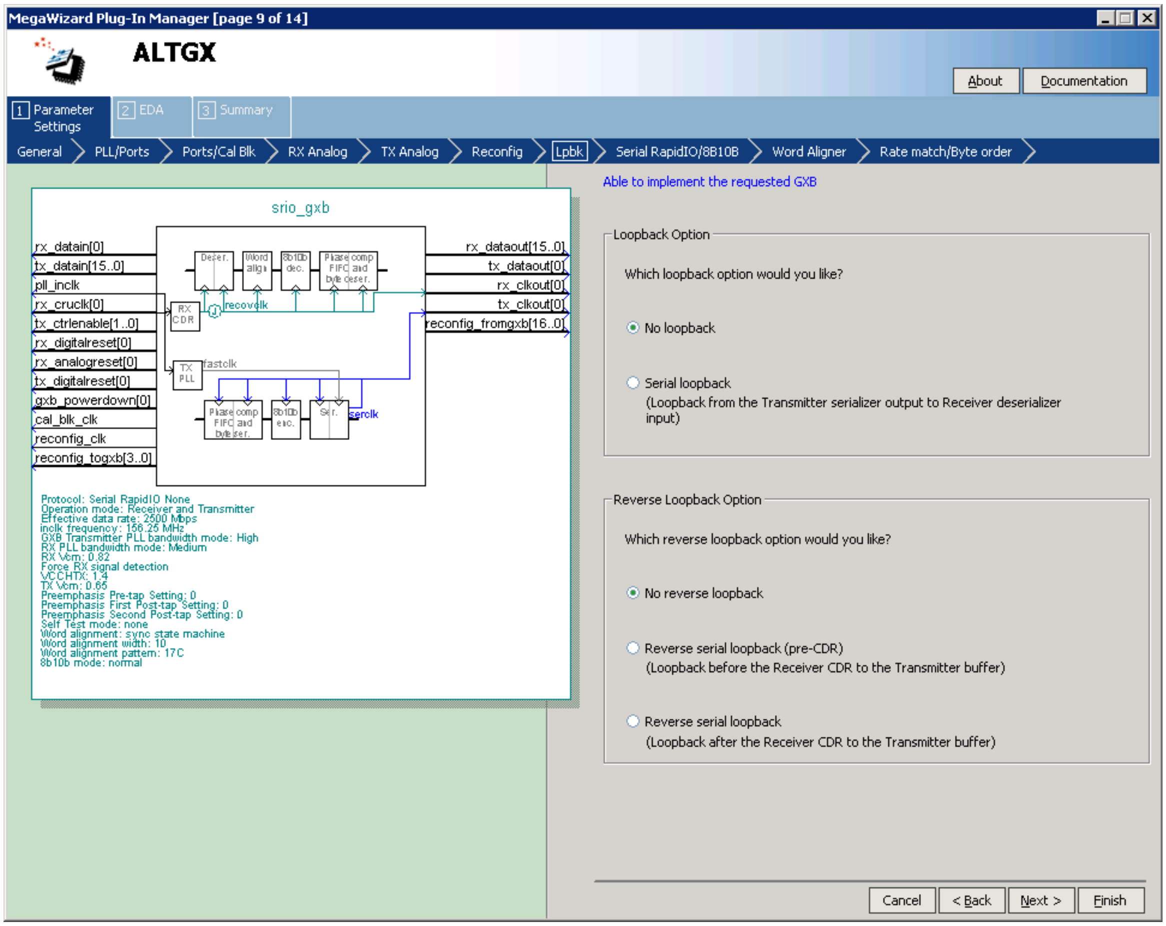
Reconfig Screen for Serial RapidIO Mode

Figure 1–8 describes the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager for the Serial RapidIO mode. Table 1–6 explains each of the parameter settings available for the **Reconfig** screen.

Lpbk Screen for Serial RapidIO Mode

Figure 1–26 shows the **Lpbk** screen of the ALTGX MegaWizard Plug-In Manager for Serial RapidIO mode.

Figure 1–26. MegaWizard Plug-In Manager - ALTGX (Lpbk Screen)



Protocol: Serial RapidIO None
Operation mode: Receiver and Transmitter
Effective data rate: 2500 Mbps
inclk frequency: 156.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Loopback Option

Which loopback option would you like?

● No loopback

○ Serial loopback
(Loopback from the Transmitter serializer output to Receiver deserializer input)

Reverse Loopback Option

Which reverse loopback option would you like?

● No reverse loopback

○ Reverse serial loopback (pre-CDR)
(Loopback before the Receiver CDR to the Transmitter buffer)

○ Reverse serial loopback
(Loopback after the Receiver CDR to the Transmitter buffer)

Table 1–22 describes the available options on the **Lpbk** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

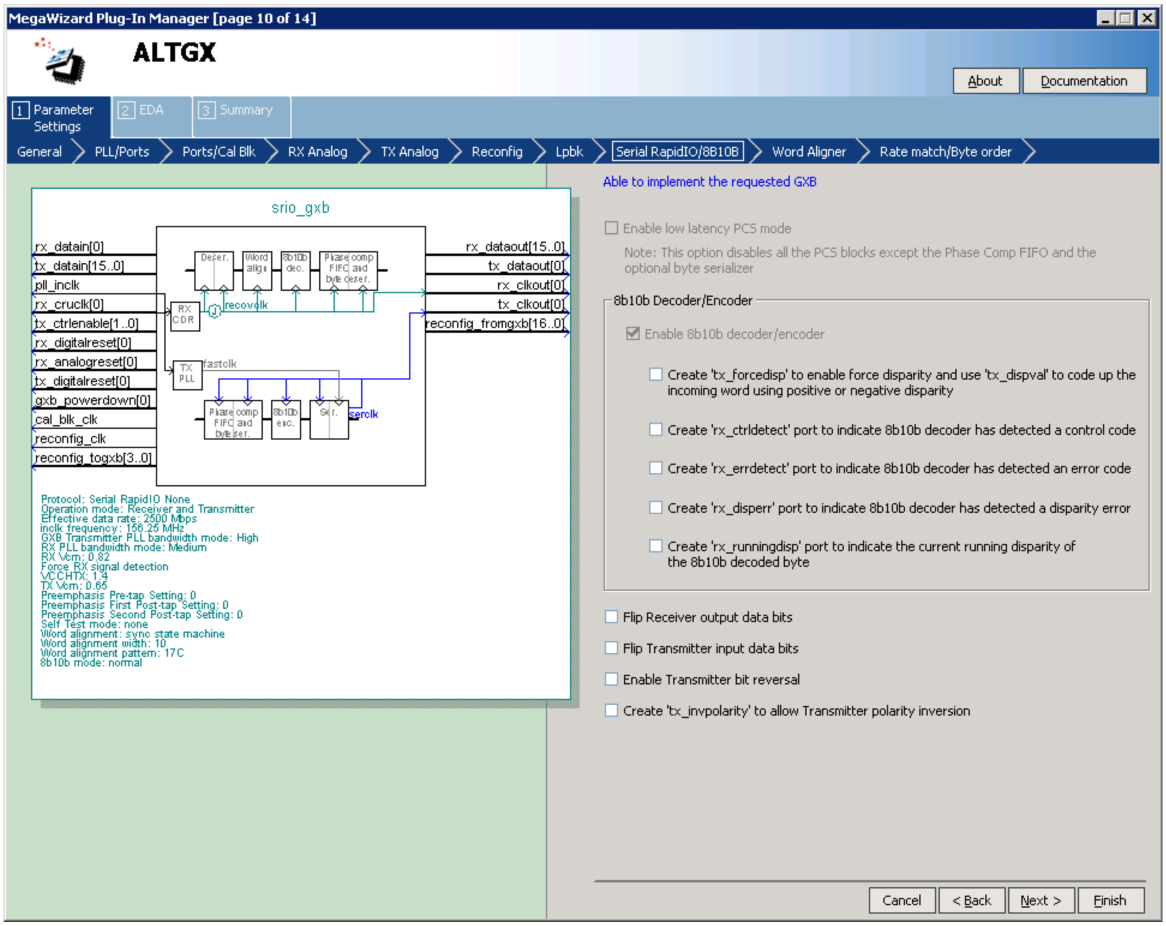
Table 1–22. MegaWizard Plug-In Manager Options (Lpbk Screen for Serial RapidIO Mode)

ALTGX Setting	Description	Reference
Which loopback option would you like?	There are two options available in Serial RapidIO mode: ■ No loopback — this is the default mode. ■ Serial loopback — if you select serial loopback, the `rx_seriallpbken` port is available to control the serial loopback feature dynamically. 1'b1 — enables serial loopback 1'b0 — disables serial loopback This signal is asynchronous to the receiver datapath.	*Serial Loopback* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Which reverse loopback option would you like?	There are three options available in Basic mode: ■ No reverse loopback — This is the default mode. ■ Reverse Serial loopback (pre-CDR) — This is the loopback before the receiver's CDR block to the transmitter buffer. The receiver path in the PCS is active but the transmitter side is not. ■ Reverse Serial loopback — This is a loopback after the receiver's CDR block to the transmitter buffer. The receiver path in the PCS is active but the transmitter side is not.	*Loopback Modes* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Serial RapidIO/8B10B Screen for Serial RapidIO Mode

Figure 1–27 shows the **Serial RapidIO/8B10B** screen of the MegaWizard Plug-In Manager for the Serial RapidIO mode.

Figure 1–27. MegaWizard Plug-In Manager - ALTGX (Serial RapidIO/8B10B Screen)



Protocol: Serial RapidIO None
Operation mode: Receiver and Transmitter
Effective data rate: 2500 Mbps
inclk frequency: 156.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

□ Enable low latency PCS mode

Note: This option disables all the PCS blocks except the Phase Comp FIFO and the optional byte serializer

8b10b Decoder/Encoder

☑ Enable 8b10b decoder/encoder

□ Create 'tx_forcedisp' to enable force disparity and use 'tx_dispval' to code up the incoming word using positive or negative disparity

□ Create 'rx_ctrldetect' port to indicate 8b10b decoder has detected a control code

□ Create 'rx_errdetect' port to indicate 8b10b decoder has detected an error code

□ Create 'rx_disperr' port to indicate 8b10b decoder has detected a disparity error

□ Create 'rx_runningdisp' port to indicate the current running disparity of the 8b10b decoded byte

□ Flip Receiver output data bits

□ Flip Transmitter input data bits

□ Enable Transmitter bit reversal

□ Create 'tx_invpolarity' to allow Transmitter polarity inversion

Table 1–23 describes the available options on the **Serial RapidIO/8B10B** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–23. MegaWizard Plug-In Manager Options (Serial RapidIO/8B10B Screen for Serial RapidIO Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
Enable 8B/10B decoder/encoder.	The 8B/10B decoder/encoder is always enabled in Serial RapidIO mode.	*8B/10B Decoder* and *8B/10B Encoder* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create a `tx_forcedisp` to enable force disparity and use `tx_dispval` to code the incoming word using positive or negative disparity.	This option is not available in Serial RapidIO mode.	—

Table 1–23. MegaWizard Plug-In Manager Options (Serial RapidIO/8B10B Screen for Serial RapidIO Mode) (Part 2 of 3)

ALTGX Setting	Description	Reference
Create an `rx_ctrldetect` port to indicate 8B/10B decoder has detected a control code.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric. This signal indicates whether the decoded 8-bit code group is a data or control code group on this port. If the received 10-bit code group is one of the 12 control code groups (/Kx.y/) specified in IEEE802.3 specification, this signal is driven high. If the received 10-bit code group is a data code group (/Dx.y/), this signal is driven low. The `rx_ctrldetect`signal is 2 bits wide. The LSB and MSB of `rx_ctrldetect`signal correspond to the LSByte and MSByte of the `rx_dataout` signal, respectively.	*8B/10B Decoder* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_errdetect` port to indicate 8B/10B decoder has detected an error code.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric. This signal indicates an 8B/10B code group violation. It is asserted high if the received 10-bit code group has a code violation or disparity error. It is used along with the `rx_disperr` signal to differentiate between a code violation error and/or a disparity error. The `rx_errdetect` signal is 2 bits wide. The LSB and MSB of `rx_errdetect` signal correspond to the LSByte and MSByte of the `rx_dataout` signal, respectively.	*8B/10B Decoder* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_disperr` port to indicate 8B/10B decoder has detected a disparity code.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric.This signal is asserted high if the received 10-bit code or data group has a disparity error. When this signal goes high, `rx_errdetect` also gets asserted high. The `rx_disperr` signal is 2-bits wide. The LSB and MSB of the `rx_disperr` signal correspond to the LSByte and MSByte of the `rx_dataout` signal, respectively.	—
Create an `rx_running_disp` port to indicate the current running disparity of the 8B10B decoded byte.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric to indicate the current running disparity of the 8b/10b decoded byte.	—
Flip receiver output data bits.	This option reverses the bit order of the parallel receiver data at a byte level at the output of the receiver phase compensation FIFO. For example, if the 16-bit parallel receiver data at the output of the receiver phase compensation FIFO is '10111100 10101101' (16'hBCAD), enabling this option reverses the data on `rx_dataout` port to '00111101 10110101' (16'h3DB5).	—

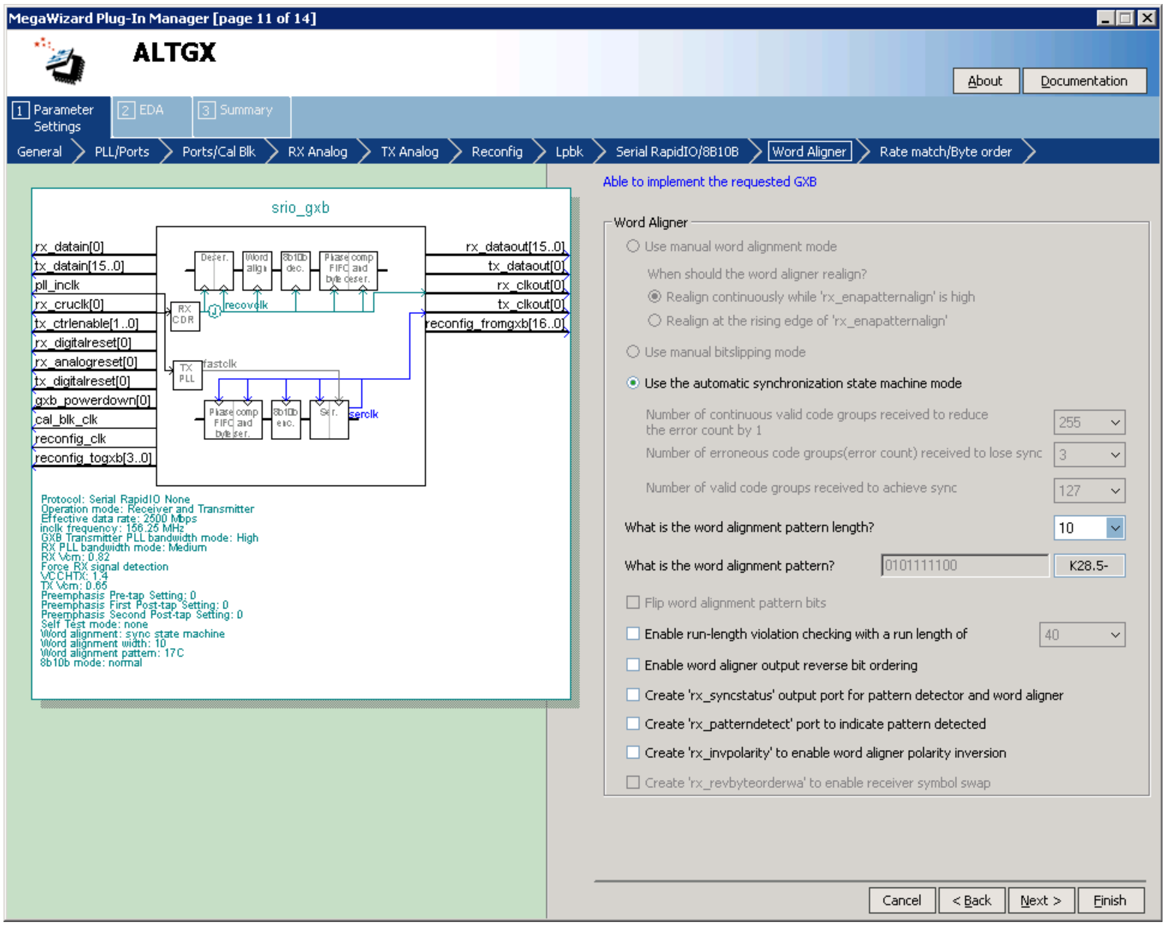
Table 1–23. MegaWizard Plug-In Manager Options (Serial RapidIO/8B10B Screen for Serial RapidIO Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
Flip transmitter input data bits.	This option reverses the bit order of the parallel transmitter data at a byte level at the input of the transmitter phase compensation FIFO. For example, if the 16-bit parallel transmitter data at `tx_datain` port is '10111100 10101101' (16'hBCAD), enabling this option reverses the input data to the transmitter phase compensation FIFO to '00111101 10110101' (16'h3DB5).	—
Enable transmitter bit reversal.	Enabling this option in reverses every bit of the 10-bit `D[9:0]` data at the input of the serializer and rewires it to `D[0:9]`. For example, if the 10-bit parallel data at the input of the serializer is '0101111100', enabling this option reverses this serializer input data to '0011111010'.	*Transmitter Bit Reversal* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create a `tx_invpolarity` port to allow Transmitter polarity inversion.	This optional port allows you to dynamically reverse the polarity of every bit of the data word fed to the serializer in the transmitter data path. Use this option when the positive and negative signals of the differential output from the transmitter (`tx_dataout`) are erroneously swapped on the board.	*Transmitter Polarity Inversion* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

Word Aligner Screen for Serial RapidIO Mode

Figure 1–28 shows the **Word Aligner** screen of the MegaWizard Plug-In Manager for the Serial RapidIO mode.

Figure 1–28. MegaWizard Plug-In Manager - ALTGX (Word Aligner Screen for Serial RapidIO Mode)



Protocol: Serial RapidIO None
Operation mode: Receiver and Transmitter
Effective data rate: 2500 Mbps
inclk frequency: 156.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Word Aligner

○ Use manual word alignment mode

When should the word aligner realign?

● Realign continuously while 'rx_enapatternalign' is high

○ Realign at the rising edge of 'rx_enapatternalign'

○ Use manual bitslipping mode

● Use the automatic synchronization state machine mode

Number of continuous valid code groups received to reduce the error count by 1: 255

Number of erroneous code groups(error count) received to lose sync: 3

Number of valid code groups received to achieve sync: 127

What is the word alignment pattern length? 10

What is the word alignment pattern? 0101111100 K28.5-

□ Flip word alignment pattern bits

□ Enable run-length violation checking with a run length of 40

□ Enable word aligner output reverse bit ordering

□ Create 'rx_syncstatus' output port for pattern detector and word aligner

□ Create 'rx_patterndetect' port to indicate pattern detected

□ Create 'rx_invpolarity' to enable word aligner polarity inversion

□ Create 'rx_revbyteorderwa' to enable receiver symbol swap

Table 1–24 describes the available options on the **Word Aligner** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–24. MegaWizard Plug-In Manager Options (Word Aligner Screen for Serial RapidIO Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
Use manual word alignment mode.	This option is not available in Serial RapidIO mode as the word aligner operates in automatic synchronization state machine mode.	—
Use manual bitslipping mode.	This option is not available in Serial RapidIO mode as the word aligner operates in automatic synchronization state machine mode.	—

Table 1–24. MegaWizard Plug-In Manager Options (Word Aligner Screen for Serial RapidIO Mode) (Part 2 of 3)

ALTGX Setting	Description	Reference
Use the automatic synchronization state machine mode?	This option is always enabled in Serial RapidIO mode as the word aligner operates in automatic synchronization state machine mode.	*Automatic Synchronization State Machine Mode Word Aligner in 10-bit PMA-PCS Interface Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Number of valid code groups received to achieve synchronization?	The ALTGX MegaWizard Plug-In Manager forces this field to **127** to comply with the Serial RapidIO specification. When the word aligner receives 127 valid word alignment patterns without receiving intermediate erroneous code groups, the `rx_syncstatus` signal is driven high to indicate that synchronization has been achieved.	*Automatic Synchronization State Machine Mode Word Aligner in 10-bit PMA-PCS Interface Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Number of erroneous code groups (error count) received to lose synchronization?	The ALTGX MegaWizard Plug-In Manager forces this field to **3** to comply with the Serial RapidIO specification. When the word aligner receives 3 erroneous code (error count = 3), the `rx_syncstatus` signal is driven low to indicate loss of synchronization.	*Automatic Synchronization State Machine Mode Word Aligner in 10-bit PMA-PCS Interface Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Number of continuous valid code groups received to reduce the error count by 1?	The ALTGX MegaWizard Plug-In Manager forces this field to **255** to comply with the Serial RapidIO specification. When the word aligner receives 255 valid code groups between erroneous code groups, the error count is reduced by 1. The `rx_syncstatus` stays high as long as the error count is lesser than the programmed error count.	*Automatic Synchronization State Machine Mode Word Aligner in 10-bit PMA-PCS Interface Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the word alignment pattern length?	The ALTGX MegaWizard Plug-In Manager only allows a 10-bit wide word alignment pattern.	*Word Aligner* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the word alignment pattern?	The ALTGX MegaWizard Plug-In Manager defaults the word alignment pattern to K28.5- (10'b0101111100). The word aligner looks for the programmed word alignment pattern and its complement.	*Word Aligner* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Flip word alignment pattern bits.	When this option is enabled, the ALTGX MegaWizard Plug-In Manager flips the bit order of the programmed word alignment pattern in the **What is the word alignment pattern?** option and uses the flipped version as the word alignment pattern. For example, if '0101111100' (17C) is the word alignment pattern and you enable this option, the word aligner uses '0011111010' as the word alignment pattern.	—

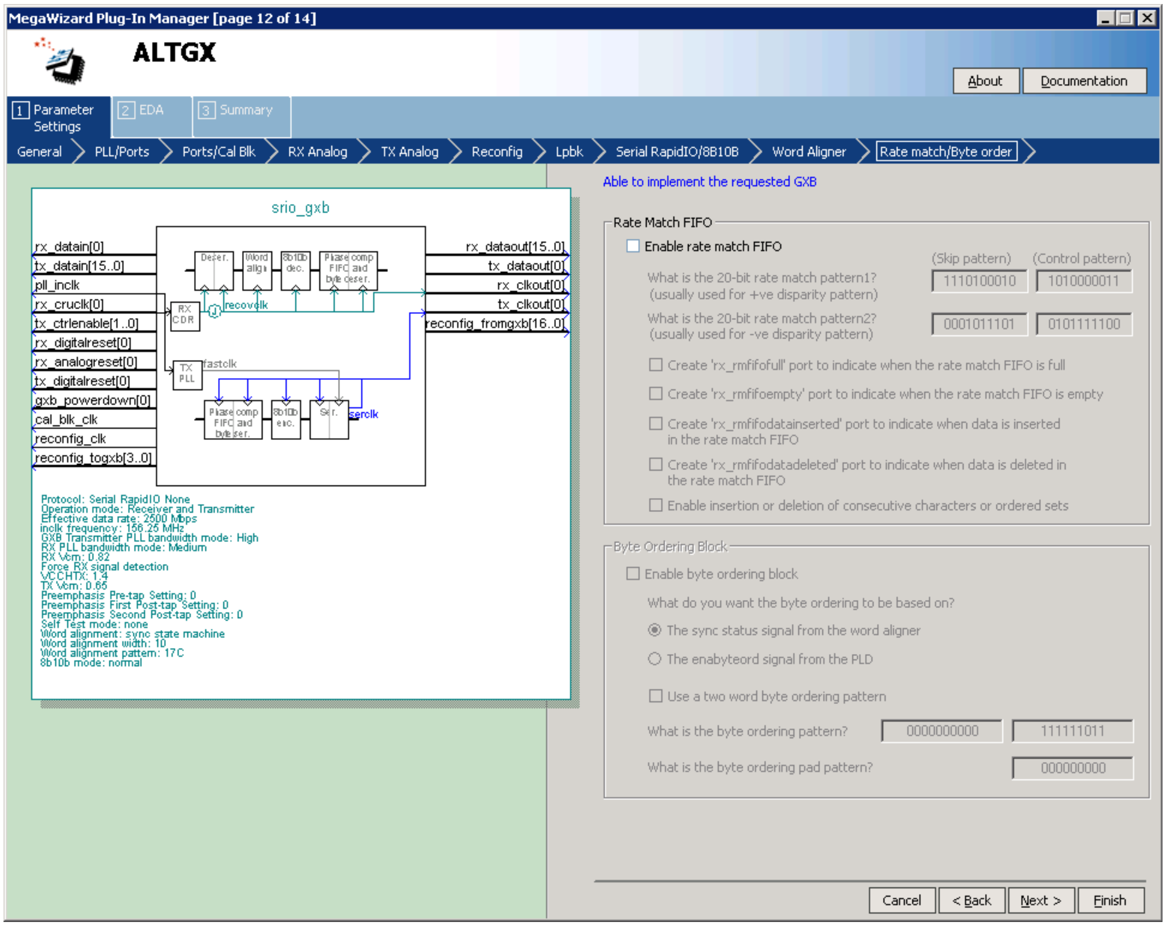
Table 1–24. MegaWizard Plug-In Manager Options (Word Aligner Screen for Serial RapidIO Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
Enable run-length violation checking with a run length of.	This option creates the output signal `rx_rlv`. Enabling this option also activates the run-length violation circuit. If the number of continuous 1's and 0's exceeds the number that you set in this option, the run-length violation circuit asserts the `rx_rlv` signal. The `rx_rlv` signal is asynchronous to the receiver data path. To ensure that the `rx_rlv` signal is captured reliably by the FPGA fabric, it is asserted for a minimum of two parallel clock cycles. For the channel width of 16-bits, the run length limits are 5 to 160 in increments of 5 in Serial RapidIO mode.	*Programmable Run Length Violation Detection* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable word aligner output reverse bit ordering.	This option is not available in this mode.	—
Create an `rx_syncstatus` output port for pattern detector and word aligner.	This is an output status signal that the word aligner forwards to the FPGA fabric to indicate that synchronization has been achieved. This signal is synchronous with the parallel receiver data on the `rx_dataout` port. This signal is not available in the bit-slip mode. The `rx_syncstatus` signal is 2 bits wide. The LSB and MSB of `rx_syncstatus` signal correspond to the LSByte and MSByte of the `rx_dataout` signal, respectively.	*Table 1-24 and Automatic Synchronization State Machine Mode Word Aligner in 10-bit PMA-PCS Interface Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_patterndetect` port to indicate pattern detected.	This is an output status signal that the word aligner forwards to the FPGA fabric to indicate that the word alignment pattern programmed has been detected in the current word boundary. The `rx_patterndetect` signal is 2 bits wide. The LSB and MSB of `rx_patterndetect` signal correspond to the LSByte and MSByte of the `rx_dataout` signal, respectively.	*Table 1-24 and Automatic Synchronization State Machine Mode Word Aligner in 10-bit PMA-PCS Interface Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_invpolarity` port to enable word aligner polarity inversion.	This optional port allows you to dynamically reverse the polarity of every bit of the received data at the input of the word aligner. Use this option when the positive and negative signals of the differential input to the receiver (`rx_datain`) are erroneously swapped on the board.	*Receiver Polarity Inversion* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Rate Match/Byte Order Screen for Serial RapidIO Mode

Figure 1–29 shows the **Rate Match/Byte Order** screen of the MegaWizard Plug-In Manager for the Serial RapidIO mode.

Figure 1–29. MegaWizard Plug-In Manager - ALTGX (Rate Match/Byte Order Screen for Serial RapidIO Mode)



Protocol: Serial RapidIO None
Operation mode: Receiver and Transmitter
Effective data rate: 2500 Mbps
inclk frequency: 156.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal

Able to implement the requested GXB

Rate Match FIFO

□ Enable rate match FIFO

	(Skip pattern)	(Control pattern)
What is the 20-bit rate match pattern1? (usually used for +ve disparity pattern)	1110100010	1010000011
What is the 20-bit rate match pattern2? (usually used for -ve disparity pattern)	0001011101	0101111100

□ Create 'rx_rmfifofull' port to indicate when the rate match FIFO is full

□ Create 'rx_rmfifoempty' port to indicate when the rate match FIFO is empty

□ Create 'rx_rmfifodatainserted' port to indicate when data is inserted in the rate match FIFO

□ Create 'rx_rmfifodatadeleted' port to indicate when data is deleted in the rate match FIFO

□ Enable insertion or deletion of consecutive characters or ordered sets

Byte Ordering Block

□ Enable byte ordering block

What do you want the byte ordering to be based on?

● The sync status signal from the word aligner

○ The enabyteord signal from the PLD

□ Use a two word byte ordering pattern

What is the byte ordering pattern? 0000000000 111111011

What is the byte ordering pad pattern? 000000000

Table 1–25 describes the available options on the **Rate Match/Byte Order** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–25. MegaWizard Plug-In Manager Options (Rate Match/Byte Order Screen for Serial RapidIO Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
Enable rate match FIFO.	This option enables the rate match (clock rate compensation) FIFO. The rate match block consists of a 20-word deep FIFO. Depending on the PPM difference, the rate match FIFO controls insertion and deletion of skip characters based on the 20-bit rate match pattern you enter in the options: **What is the 20-bit rate match pattern1?** and **What is the 20-bit rate match pattern2?** To enable this block: The transceiver channel must have both the transmitter and the receiver channel instantiated. You must select the **Receiver and Transmitter** option in the **What is the operation mode?** field in the **General** screen.	*Rate Match FIFO in Basic Single-Width Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the 20-bit rate match pattern1? (usually used for +ve disparity pattern)	Enter a 10-bit skip pattern and a 10-bit control pattern. In the skip pattern field, you must choose a 10-bit code group that has neutral disparity. When the rate matcher receives the 10-bit control pattern followed by the 10-bit skip pattern, it inserts or deletes the 10-bit skip pattern as necessary to avoid the rate match FIFO overflow or underflow conditions. *(1)*	*Rate Match FIFO in Basic Single-Width Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the 20-bit rate match pattern2? (usually used for -ve disparity pattern)	Enter a 10-bit skip pattern and a 10-bit control pattern. In the skip pattern field, you must choose a 10-bit code group that has neutral disparity. When the rate matcher receives the 10-bit control pattern followed by the 10-bit skip pattern, it inserts or deletes the 10-bit skip pattern as necessary to avoid the rate match FIFO from overflowing or under-runnning. *(1)*	*Rate Match FIFO in Basic Single-Width Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–25. MegaWizard Plug-In Manager Options (Rate Match/Byte Order Screen for Serial RapidIO Mode) (Part 2 of 3)

ALTGX Setting	Description	Reference
Enable `rx_rmfifofull` flag to indicate when the rate match FIFO is full.	This option creates the output port `rx_rmfifofull` when you enable the option **Enable Rate Match FIFO**. It is a status flag that the rate match block forwards to the FPGA fabric. It indicates when the rate match FIFO block is full (20 words). This signal remains high as long as the FIFO is full. It is asynchronous to the receiver data path.	*Rate Match (Clock Compensation) FIFO* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable `rx_rmfifoempty` flag to indicate when the rate match FIFO is empty.	This option creates the output port `rx_rmfifoempty` when you enable the option **Enable Rate Match FIFO**. It is a status flag that the rate match block forwards to the FPGA fabric. It indicates when the rate match FIFO block is empty (5 words). This signal remains high as long as the FIFO is empty. It is asynchronous to the receiver data path.	*Rate Match (Clock Compensation) FIFO* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable `rx_rmfifodatainserted` flag to indicate when data is inserted in the rate match FIFO.	This option creates the output port `rx_rmfifodatainserted` flag when you enable the option **Enable Rate Match FIFO**. It is a status flag that the rate match block forwards to the FPGA fabric. It indicates the insertion of skip patterns. For every deletion, this signal is high for one parallel clock cycle. This signal is asynchronous to the receiver data path.	*Rate Match (Clock Compensation) FIFO* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–25. MegaWizard Plug-In Manager Options (Rate Match/Byte Order Screen for Serial RapidIO Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
Enable `rx_rmfifodatadeleted` flag to indicate when data is deleted from the rate match FIFO.	This option creates the output port `rx_rmfifodatadeleted` when you enable the option **Enable Rate Match FIFO**. It is a status flag that the rate match block forwards to the FPGA fabric. It indicates the deletion of skip patterns. For every insertion, this signal is high for one parallel clock cycle. This signal is asynchronous to the receiver data path.	*Rate Match (Clock Compensation) FIFO* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable byte ordering block.	This option is not available in the Serial Rapid IO mode.	—
Create an `rx_revbyteorderwa` to enable receiver symbol swap.	This option is not available in the Serial Rapid IO mode.	—

Note to Table 1–25:

(1) If you want the rate matcher to insert/delete both the positive and negative disparities of the 20-bit rate matching pattern, you can enter the positive disparity as pattern1 and negative disparity as pattern2.

EDA Screen for Serial RapidIO Mode

Figure 1–13 describes the **EDA** screen of the MegaWizard Plug-In Manager for the Serial RapidIO mode. The **Generate Netlist** option generates a netlist for the third party EDA synthesis tool to estimate timing and resource utilization for the ALTGX instance.

Summary for Serial RapidIO Mode

Figure 1–14 describes the **Summary** screen of the MegaWizard Plug-In Manager for the Serial RapidIO mode. You can select optional files on this page. After you make your selections, click **Finish** to generate the files.

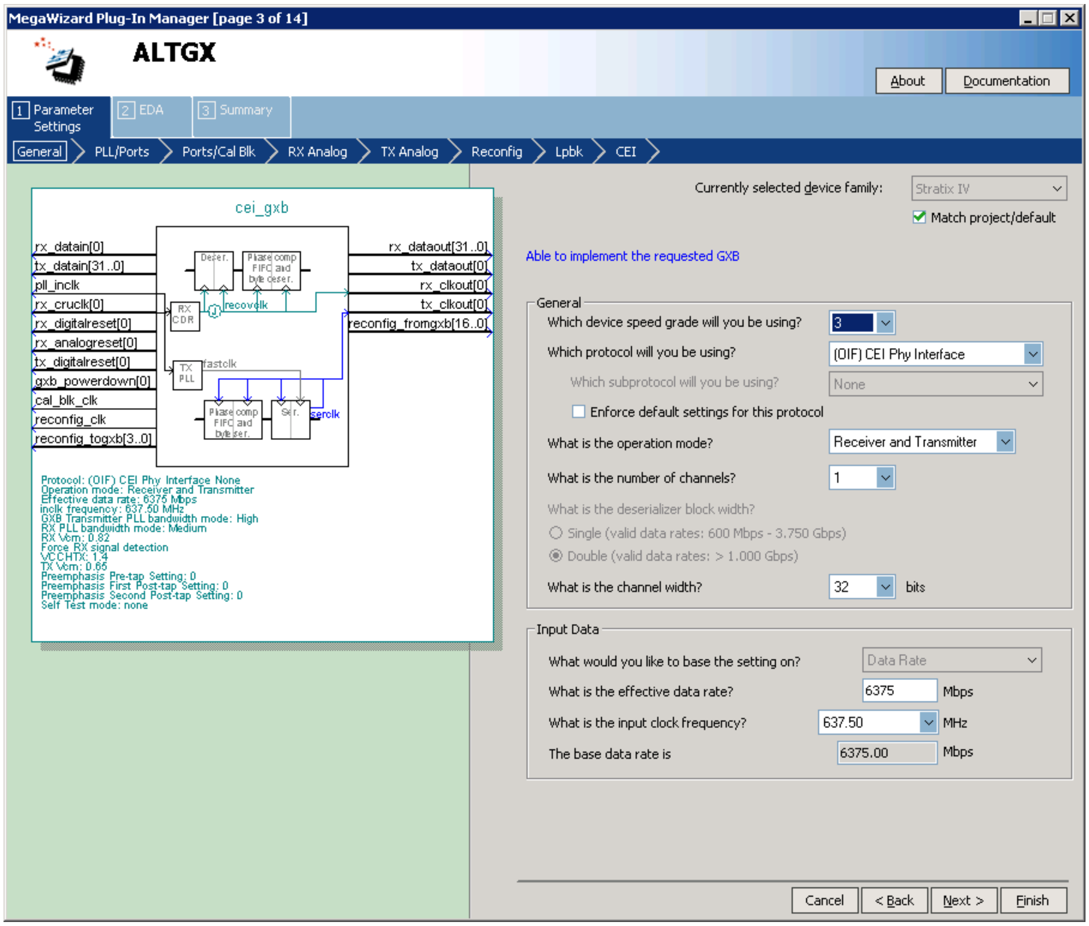
(OIF) CEI PHY Interface Mode

This section provides descriptions of the options available on the individual pages of the ALTGX MegaWizard Plug-In Manager for the (OIF) CEI PHY Interface mode. The MegaWizard Plug-In Manager provides a warning if any of the settings you choose are illegal.

General Screen for (OIF) CEI PHY Interface Mode

Figure 1–30 shows the **General** screen of the ALTGX MegaWizard Plug-In Manager for (OIF) CEI PHY Interface mode.

Figure 1–30. MegaWizard Plug-In Manager - ALTGX (General Screen)



Protocol: (OIF) CEI Phy Interface None
Operation mode: Receiver and Transmitter
Effective data rate: 6375 Mbps
inclk frequency: 637.50 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none

Currently selected device family: Stratix IV
☑ Match project/default

Able to implement the requested GXB

General

Which device speed grade will you be using? 3
Which protocol will you be using? (OIF) CEI Phy Interface
Which subprotocol will you be using? None
☐ Enforce default settings for this protocol
What is the operation mode? Receiver and Transmitter
What is the number of channels? 1
What is the deserializer block width?
○ Single (valid data rates: 600 Mbps - 3.750 Gbps)
● Double (valid data rates: > 1.000 Gbps)
What is the channel width? 32 bits

Input Data

What would you like to base the setting on? Data Rate
What is the effective data rate? 6375 Mbps
What is the input clock frequency? 637.50 MHz
The base data rate is 6375.00 Mbps

Table 1–26 describes the available options on the **General** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–26. MegaWizard Plug-In Manager Options (General Screen for [OIF] CEI PHY Interface Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Which device speed grade will you be using?	Select the speed grade of the device you will be using. The available speed grades are 2, 2x, 3, and 4. Based on the speed grade you select, the corresponding Stratix IV device can operate at the following maximum speeds: -2 => 8.5 Gbps -2x, -3 : 6.5 Gbps -4 => 5 Gbps (Therefore, you cannot select Stratix IV device with this speed grade for the (OIF) CEI PHY Interface Mode)	—
Which protocol will you be using?	Selects the specific protocol or modes that the transceiver operates under. For the (OIF) CEI PHY Interface mode, you must select the **(OIF) CEI PHY Interface** protocol.	*Table 1-18* in the *DC and Switching Characteristics of the Stratix IV Device Family* chapter in volume 5 of the *Stratix IV Device Handbook* and *Functional Modes* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Which subprotocol will you be using?	This option does not apply to the (OIF) CEI PHY Interface mode.	—
Enforce default settings for this protocol.	If this option is checked, all (OIF) CEI PHY Interface specific ports are used.	—
What is the operation mode?	The available operation modes are **Receiver only**, **Transmitter only**, and **Receiver and Transmitter**.	—
What is the number of channels?	This selects how many duplicate channels this ALTGX instance contains. In (OIF) CEI PHY Interface mode, the number of channels increments by 1.	—
What is the deserializer block width?	The (OIF) CEI PHY Interface operates in double-width mode only. Single-width mode is not allowed.	—
What is the channel width?	This option selects the FPGA fabric-Transceiver width. Only 32 bits are allowed in (OIF) CEI PHY Interface mode.	*Byte Serializer* and *Byte Deserializer* sections in the Stratix IV Transceiver Architecture chapter in volume 2 of the *Stratix IV Device Handbook.*
What would you like to base the setting on?	This option is not available for selection in (OIF) CEI PHY Interface mode.	—
What is the effective data rate?	The allowed effective data rate is between 3135 Mbps and 6375 Mbps in (OIF) CEI PHY Interface mode. Enter the transceiver channel's serial data rate in this field.	—

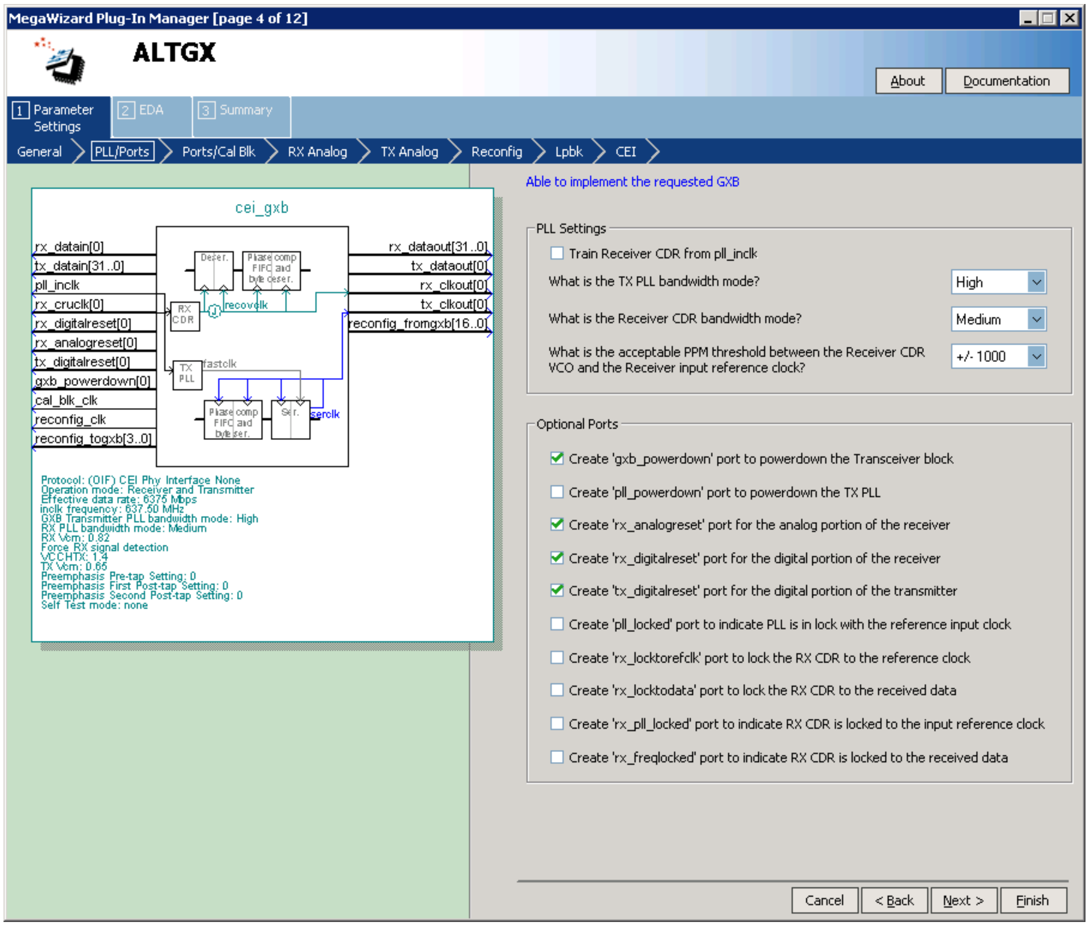
Table 1–26. MegaWizard Plug-In Manager Options (General Screen for [OIF] CEI PHY Interface Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
What is the input clock frequency?	Based on the effective data rate value you enter in the **What is the effective data rate?** field, the ALTGX MegaWizard Plug-In Manager determines the input reference clock frequencies depending on the available multiplier settings.	*CMU PLL and Receiver CDR Input Reference Clock* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Specify base data rate.	This option is not available for selection in this mode. The ALTGX MegaWizard Plug-In Manager provides you the base data rate options for the VCO in the CMU PLL and receiver CDR in this option.	—

PLL/Ports Screen for (OIF) CEI PHY Interface Mode

Figure 1–31 shows the **PLL/ports** screen of the ALTGX MegaWizard Plug-In Manager for (OIF) CEI PHY Interface mode.

Figure 1–31. MegaWizard Plug-In Manager - ALTGX (PLL/Ports Screen)



Protocol: (OIF) CEI Phy Interface None
Operation mode: Receiver and Transmitter
Effective data rate: 6375 Mbps
inclk frequency: 637.50 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none

Able to implement the requested GXB

PLL Settings

☐ Train Receiver CDR from pll_inclk

What is the TX PLL bandwidth mode? High

What is the Receiver CDR bandwidth mode? Medium

What is the acceptable PPM threshold between the Receiver CDR VCO and the Receiver input reference clock? +/- 1000

Optional Ports

☑ Create 'gxb_powerdown' port to powerdown the Transceiver block

☐ Create 'pll_powerdown' port to powerdown the TX PLL

☑ Create 'rx_analogreset' port for the analog portion of the receiver

☑ Create 'rx_digitalreset' port for the digital portion of the receiver

☑ Create 'tx_digitalreset' port for the digital portion of the transmitter

☐ Create 'pll_locked' port to indicate PLL is in lock with the reference input clock

☐ Create 'rx_locktorefclk' port to lock the RX CDR to the reference clock

☐ Create 'rx_locktodata' port to lock the RX CDR to the received data

☐ Create 'rx_pll_locked' port to indicate RX CDR is locked to the input reference clock

☐ Create 'rx_freqlocked' port to indicate RX CDR is locked to the received data

Table 1–27 describes the available options on the **PLL/ports** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–27. MegaWizard Plug-In Manager Options (PLL/Ports Screen for (OIF) CEI PHY Interface Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
Train receiver CDR from `pll_inclk`.	If you select this option, the input reference clock to the CMU PLL trains the receiver CDR.	*Table 1-2* in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the TX PLL bandwidth mode?	The appropriate bandwidth settings will be fixed after characterization.	*PLL Bandwidth Setting* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the receiver CDR bandwidth mode?	The appropriate bandwidth settings will be fixed after characterization.	—

Table 1–27. MegaWizard Plug-In Manager Options (PLL/Ports Screen for (OIF) CEI PHY Interface Mode) (Part 2 of 3)

ALTGX Setting	Description	Reference
What is the acceptable PPM threshold between the receiver CDR VCO and the receiver input reference clock?	In automatic lock mode, the CDR remains in LTD mode as long as the PPM difference between the CDR VCO output clock and the input reference clock is less than the PPM value that you set in this option. If the PPM difference is greater than the PPM value that you set in this option, the CDR switches to LTR mode. The range of values available in this option is ±62.5 ppm to ±1000 ppm. *(1)*	*Automatic Lock Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `gxb_powerdown` port to power down the Transceiver block.	This is an optional signal. When asserted, it powers down the entire transceiver block. If none of the channels are instantiated in a transceiver block, the Quartus II software automatically powers down the entire transceiver block.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `pll_powerdown` port to power down the TX PLL.	Each transceiver block has two CMU PLLs. Each CMU PLL has a dedicated power down signal called `pll_powerdown`. This signal powers down the CMU PLL.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_analogreset` port for the analog portion of the receiver.	Receiver analog reset port available in **Receiver only** and **Receiver and Transmitter** operation modes. Resets part of the analog portion of the receiver CDR in the receiver channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_digitalreset` port for the digital portion of the receiver.	Receiver digital reset port available in **Receiver only** and **Receiver and Transmitter** operation modes. Resets the PCS portion of the receiver channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_digitalreset` port for the digital portion of the transmitter.	Transmitter digital reset port available in **Transmitter only** and **Receiver and Transmitter** operation modes. Resets the PCS portion of the transmitter channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `pll_locked` port to indicate PLL locked to the input reference clock.	Each CMU PLL has a dedicated `pll_locked` signal that is fed to the FPGA fabric to indicate when the PLL is locked to the input reference clock.	*Transceiver Reset Sequences* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_locktorefclk` port to lock the RX CDR to the input reference clock.	This is an optional port. When this signal is asserted high, the LTR/LTD controller forces the receiver CDR to lock to the phase and frequency of the input reference clock. *(1), (2)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–27. MegaWizard Plug-In Manager Options (PLL/Ports Screen for (OIF) CEI PHY Interface Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
Create an `rx_locktodata` port to lock the RX CDR to the received data.	This is an optional port. When this signal is asserted high, the LTR/LTD controller forces the receiver CDR to lock to the received data. *(1), (2)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_pll_locked` port to indicate RX CDR is locked to the input reference clock.	In LTR mode, this signal is asserted high to indicate that the receiver CDR has locked to the phase and frequency of the input reference clock. In LTD mode, this signal has no significance. *(1)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_freqlocked` port to indicate RX CDR is locked to the received data.	This signal gets asserted high to indicate that the receiver CDR has switched from LTR to LTD mode. This signal has relevance only in automatic lock mode and may be required to control the transceiver resets, as discussed in the *User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook. (1)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

***Notes to* Table 1–27:**

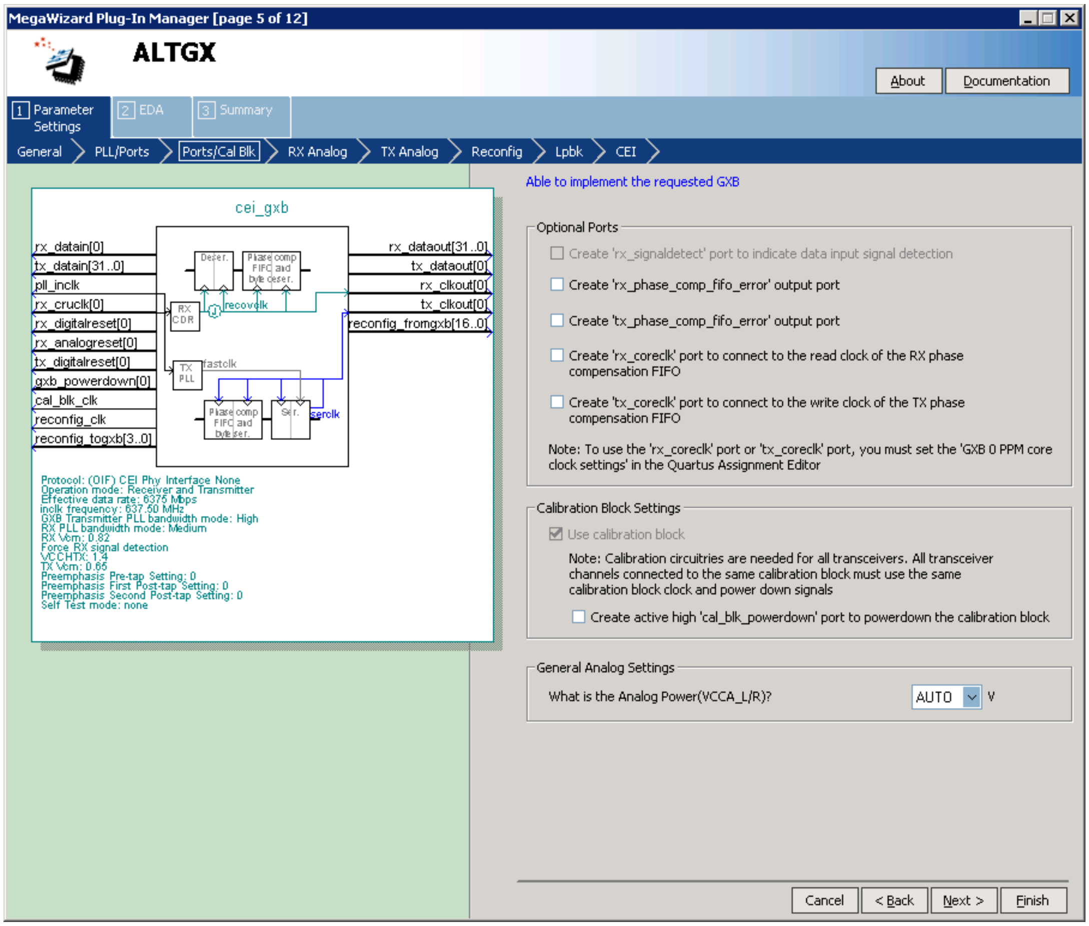
(1) LTR mode is lock-to-reference mode and LTD mode is lock-to-data mode.

(2) When `rx_locktorefclk` and `rx_locktodata` are both asserted high, `rx_locktodata` takes precedence over `rx_locktorefclk`, forcing the CDR to lock to the received data. When both these signals are de-asserted, the LTR/LTD controller is configured in automatic lock mode.

Ports/Cal Blk Screen for (OIF) CEI PHY Interface Mode

Figure 1–32 shows the **Ports/Cal Blk** screen of the ALTGX MegaWizard Plug-In Manager for (OIF) CEI PHY Interface mode.

Figure 1–32. MegaWizard Plug-In Manager - ALTGX (Ports/Cal Blk Screen for (OIF) CEI PHY Interface Mode)



Protocol: (OIF) CEI Phy Interface None
Operation mode: Receiver and Transmitter
Effective data rate: 6375 Mbps
inclk frequency: 637.50 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none

Able to implement the requested GXB

Optional Ports

□ Create 'rx_signaldetect' port to indicate data input signal detection

□ Create 'rx_phase_comp_fifo_error' output port

□ Create 'tx_phase_comp_fifo_error' output port

□ Create 'rx_coreclk' port to connect to the read clock of the RX phase compensation FIFO

□ Create 'tx_coreclk' port to connect to the write clock of the TX phase compensation FIFO

Note: To use the 'rx_coreclk' port or 'tx_coreclk' port, you must set the 'GXB 0 PPM core clock settings' in the Quartus Assignment Editor

Calibration Block Settings

☑ Use calibration block

Note: Calibration circuitries are needed for all transceivers. All transceiver channels connected to the same calibration block must use the same calibration block clock and power down signals

□ Create active high 'cal_blk_powerdown' port to powerdown the calibration block

General Analog Settings

What is the Analog Power(VCCA_L/R)? AUTO V

Table 1–28 describes the available options on the **Ports/Cal Blk** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–28. MegaWizard Plug-In Manager Options (Ports/Cal Blk Screen for [OIF] CEI PHY Interface Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Create an `rx_signaldetect` port to indicate data input signal detection.	This port is not available in the (OIF) CEI PHY interface mode.	*Signal Threshold Detection Circuitry* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create an `rx_phase_comp_fifo_error` output port.	This optional output port indicates receiver phase compensation FIFO overflow or underrun condition.	*Receiver Phase Compensation FIFO Error Flag* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

Table 1–28. MegaWizard Plug-In Manager Options (Ports/Cal Blk Screen for [OIF] CEI PHY Interface Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
Create a `tx_phase_comp_fifo_error` output port.	This optional output port indicates Transmitter Phase Compensation FIFO overflow or underrun condition.	*TX Phase Compensation FIFO Status Signal* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_coreclk` port to connect to the read clock of the RX phase compensation FIFO.	The parallel output data from the receiver can be clocked using this optional input port. It allows you to clock the read side of the receiver phase compensation FIFO with a user-provided clock (FPGA fabric clock, FPGA fabric-transceiver interface clock or input reference clock).	*FPGA Fabric-Transceiver Interface Clocking* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_coreclk` port to connect to the write clock of the TX phase compensation FIFO.	The parallel transmitter data generated in the FPGA fabric can be clocked using this optional input port. It allows you to clock the write side of the transmitter phase compensation FIFO with a user-provided clock (FPGA fabric clock, FPGA fabric-transceiver interface clock, or input reference clock).	*FPGA Fabric-Transceiver Interface Clocking* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Use calibration block.	Calibration block is always enabled.	*Calibration Blocks* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an active high `cal_blk_powerdown` to power down the calibration block.	Asserting this signal high powers down the calibration block. A high-to-low transition on this signal restarts calibration.	*Input Signals to the Calibration Block* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the analog power ($V_{CCA\_L/R}$)?	The options available for selection are 2.5 V, 3.0 V and AUTO: 3.0 V : upto 8.5 Gbps 2.5 V : upto 4.25 Gbps AUTO : The ALTGX MegaWizard Plug-In Manager will automatically set V_{CCA} to 2.5 V and V_{CCH} (transmitter buffer voltage) to 1.5V for the VCO base data rates less than 4.25 Gbps. -OR- V_{CCA} to 3.0 V and V_{CCH} (transmitter buffer voltage) to 1.4 V for the VCO base data rates greater than 4.25 Gbps.	*General Requirements to Combine Channels* section in the *Configuring Multiple Protocols and Data Rates* chapter in volume 2 of the *Stratix IV Device Handbook.*

RX Analog Screen for (OIF) CEI PHY Interface Mode

Figure 1–6 describes the **RX Analog** screen of the ALTGX MegaWizard Plug-In Manager for the (OIF) CEI PHY Interface mode. Table 1–4 explains each of the parameter settings available for the **RX Analog** screen.

TX Analog Screen for (OIF) CEI PHY Interface Mode

Figure 1–7 describes the **TX Analog** screen of the ALTGX MegaWizard Plug-In Manager for the (OIF) CEI PHY Interface mode. Table 1–5 explains each of the parameter settings available for the **TX Analog** screen.

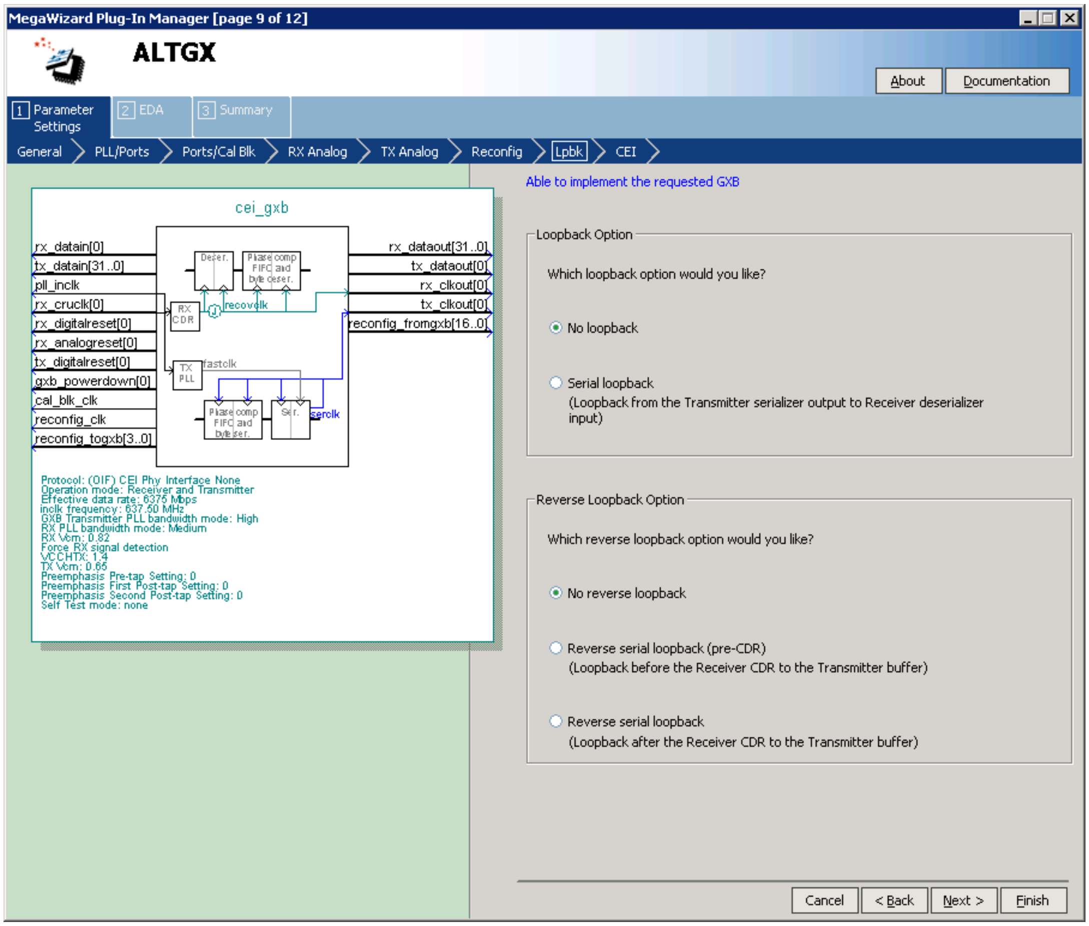
Reconfig Screen for (OIF) CEI PHY Interface Mode

Figure 1–8 describes the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager for the (OIF) CEI PHY Interface mode. Table 1–6 explains each of the parameter settings available for the **Reconfig** screen.

Lpbk Screen for (OIF) CEI PHY Interface Protocol Selection

Figure 1–33 shows the **Lpbk** screen of the MegaWizard Plug-In Manager for the (OIF) CEI PHY Interface protocol selection.

Figure 1–33. MegaWizard Plug-In Manager - ALTGX (Lpbk Screen for (OIF) CEI PHY Interface Protocol Selection)



Protocol: (OIF) CEI Phy Interface None
Operation mode: Receiver and Transmitter
Effective data rate: 6375 Mbps
inclk frequency: 637.50 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none

Able to implement the requested GXB

Loopback Option

Which loopback option would you like?

● No loopback

○ Serial loopback
(Loopback from the Transmitter serializer output to Receiver deserializer input)

Reverse Loopback Option

Which reverse loopback option would you like?

● No reverse loopback

○ Reverse serial loopback (pre-CDR)
(Loopback before the Receiver CDR to the Transmitter buffer)

○ Reverse serial loopback
(Loopback after the Receiver CDR to the Transmitter buffer)

Table 1–29 describes the available options on the **Lpbk** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

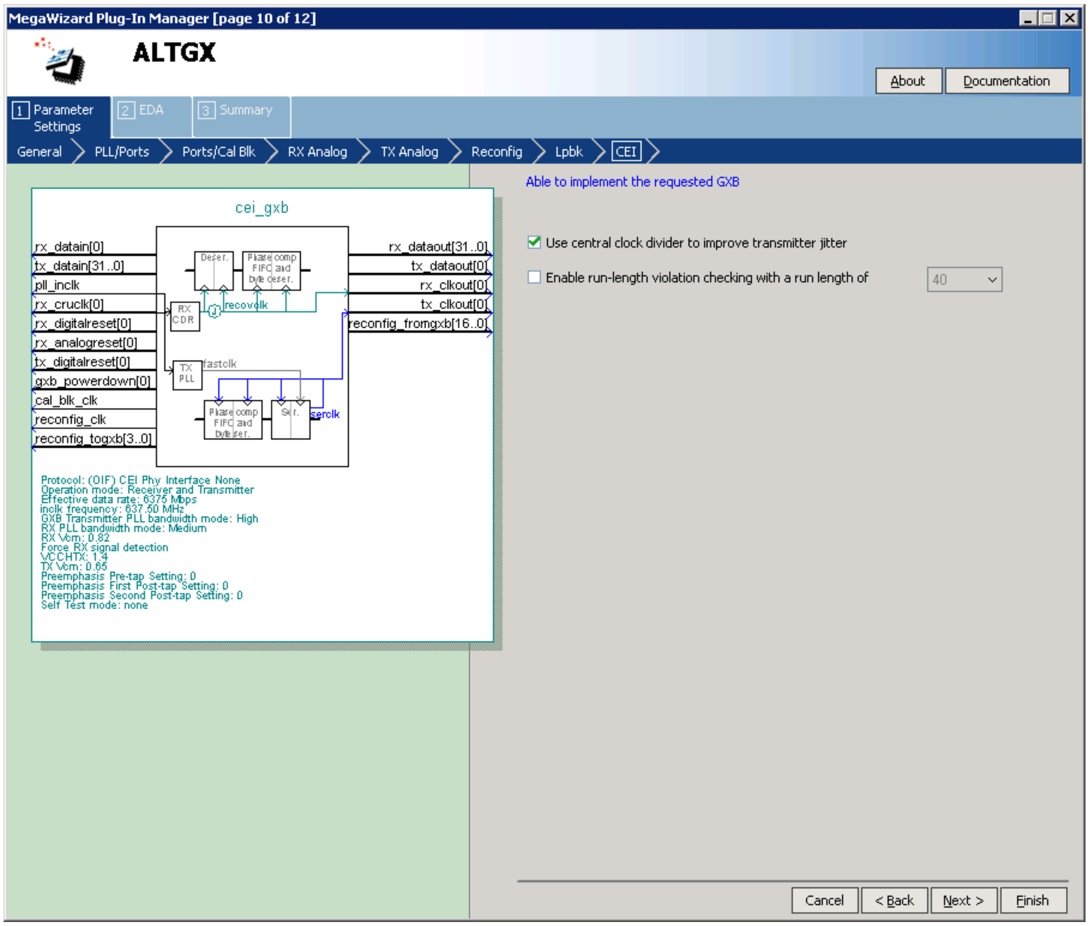
Table 1–29. MegaWizard Plug-In Manager Options (Lpbk Screen for (OIF) CEI PHY Interface Mode)

ALTGX Setting	Description	Reference
Which loopback option would you like?	There are two options available in (OIF) CEI PHY mode: ■ No loopback — this is the default mode. ■ Serial loopback — if you select serial loopback, the `rx_seriallpbken` port is available to control the serial loopback feature dynamically. 1'b1 — enables serial loopback 1'b0 — disables serial loopback This signal is asynchronous to the receiver data path.	*Serial Loopback* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Which reverse loopback option would you like?	There are three options available in Basic mode: ■ No reverse loopback — This is the default mode. ■ Reverse Serial loopback (pre-CDR) — This is the loopback before the receiver's CDR block to the transmitter buffer. The receiver path in the PCS is active but the transmitter side is not. ■ Reverse Serial loopback — This is a loopback after the receiver's CDR block to the transmitter buffer. The receiver path in the PCS is active but the transmitter side is not.	*Loopback Modes* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

CEI Screen

Figure 1–34 shows the **CEI** screen of the MegaWizard Plug-In Manager for the (OIF) CEI PHY Interface protocol selection. If the option **Enforce default settings for this protocol** is selected, this page does not appear in the MegaWizard Plug-In Manager.

Figure 1–34. MegaWizard Plug-In Manager - ALTGX (CEI Screen)



Protocol: (OIF) CEI Phy Interface None
Operation mode: Receiver and Transmitter
Effective data rate: 6375 Mbps
inclk frequency: 637.50 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none

Able to implement the requested GXB

☑ Use central clock divider to improve transmitter jitter

☐ Enable run-length violation checking with a run length of 40

Table 1–30 describes the available options on the **CEI** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–30. MegaWizard Plug-In Manager Options (CEI Screen for [OIF] CEI PHY Interface Mode)

ALTGX Setting	Description	Reference
Use central clock divider to improve Transmitter jitter.	If you select this option, the high-speed serial clock generated by the central clock divider clocks all four transceiver channels within the same transceiver block. Otherwise, the high-speed serial clock generated by the local clock divider in each channel clocks the respective channel. The transmitter PCS is not bonded in the (OIF) CEI PHY Interface with the low-jitter option selected.	*Transmitter Placement Limitations with Use Central Clock Divider to Improve Transmitter Jitter Option Enabled* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Enable run-length violation checking with a run length of.	This option creates the output signal `rx_rlv`. Enabling this option also activates the run-length violation circuit. If the number of continuous 1's and 0's exceeds the number that you set in this option, the run-length violation circuit asserts the `rx_rlv` signal. The `rx_rlv` signal is asynchronous to the receiver data path and is asserted for a minimum of two recovered clock cycles. For a 32-bit channel width, the run length limits are 8 to 512 in increments of 8.	*Programmable Run Length Violation Detection* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

EDA Screen for (OIF) CEI PHY Interface Mode

Figure 1–13 describes the **EDA** screen of the MegaWizard Plug-In Manager for (OIF) CEI PHY Interface mode. The **Generate Netlist** option generates a netlist for the third party EDA synthesis tool to estimate timing and resource utilization for the ALTGX instance.

Summary for (OIF) CEI PHY Interface Mode

Figure 1–14 describes the **Summary** screen of the MegaWizard Plug-In Manager for (OIF) CEI PHY Interface mode. You can select optional files on this page. After you make your selections, click **Finish** to generate the files.

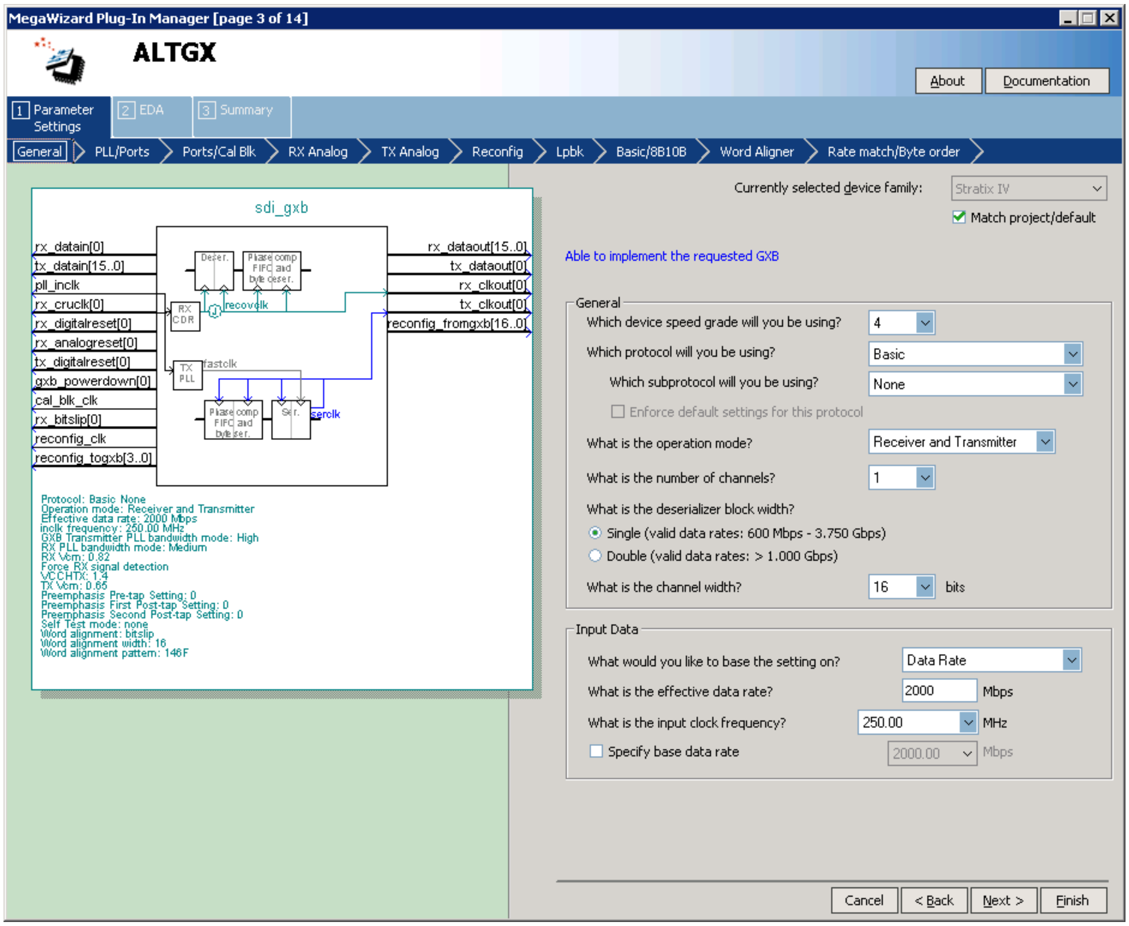
SDI Mode

This section provides descriptions of the options available on the individual pages of the ALTGX MegaWizard Plug-In Manager for SDI mode. The MegaWizard Plug-In Manager provides a warning if any of the settings you choose are illegal.

General Screen for SDI Mode

Figure 1–35 shows the **General** screen of the ALTGX MegaWizard Plug-In Manager for SDI mode.

Figure 1–35. MegaWizard Plug-In Manager - ALTGX (General Screen for SDI Mode)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip
Word alignment width: 16
Word alignment pattern: 146F

Currently selected device family: Stratix IV
☑ Match project/default

Able to implement the requested GXB

General

Which device speed grade will you be using? 4
Which protocol will you be using? Basic
Which subprotocol will you be using? None
☐ Enforce default settings for this protocol
What is the operation mode? Receiver and Transmitter
What is the number of channels? 1
What is the deserializer block width?
● Single (valid data rates: 600 Mbps - 3.750 Gbps)
○ Double (valid data rates: > 1.000 Gbps)
What is the channel width? 16 bits

Input Data

What would you like to base the setting on? Data Rate
What is the effective data rate? 2000 Mbps
What is the input clock frequency? 250.00 MHz
☐ Specify base data rate 2000.00 Mbps

Table 1–31 describes the available options on the **General** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–31. MegaWizard Plug-In Manager Options (General Screen for SDI Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Which device speed grade will you be using?	Select the speed grade of the device you will be using. The available speed grades are 2, 2x, 3, and 4.	—
Which protocol will you be using?	Selects the specific protocol or modes under which the transceiver operates. For SDI mode, you must select the **SDI** protocol.	*Table 1-18* in the *DC and Switching Characteristics of the Stratix IV Device Family* chapter in volume 5 of the *Stratix IV Device Handbook* and the *SDI Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

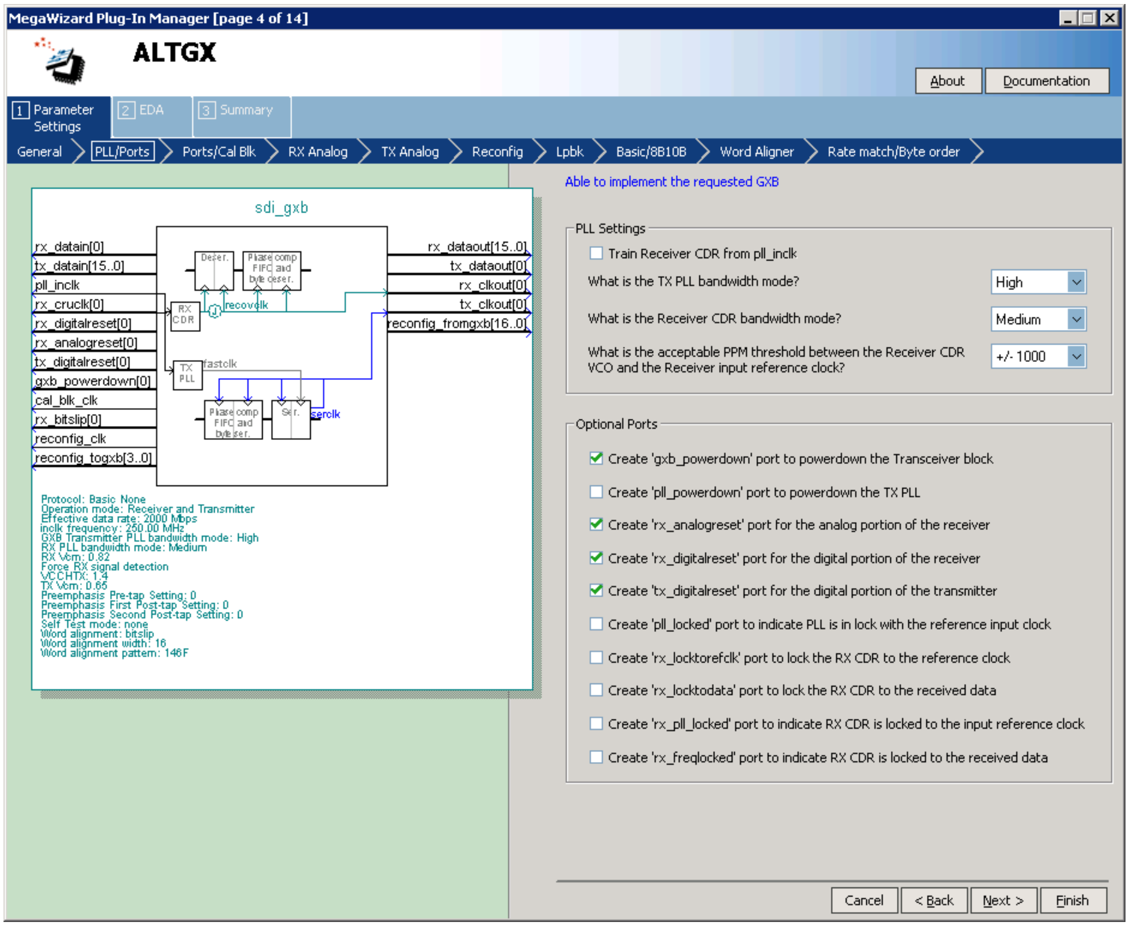
Table 1–31. MegaWizard Plug-In Manager Options (General Screen for SDI Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
Which subprotocol will you be using?	In SDI mode, the two available subprotocols are: ■ 3G: third-generation (3 Gbps) SDI at 2967 Mbps or 2970 Mbps ■ HD: high-definition SDI at 1483.5 Mbps or 1485 Mbps	—
Enforce default settings for this protocol.	This option is not available in SDI mode.	—
What is the operation mode?	The available operation modes are **Receiver only**, **Transmitter only**, and **Receiver and Transmitter**.	—
What is the number of channels?	It is the number of channels required with the same configuration. This option determines how many identical channels this ALTGX instance contains.	—
What is the deserializer block width?	SDI mode only operates in single-width mode. Double-width mode is not allowed.	—
What is the channel width?	This option determines the FPGA fabric-Transceiver Interface width: ■ In HD mode, 10-bit and 20-bit channel widths are allowed. ■ In 3G mode, only 20-bit channel width is allowed. ■ In 10-bit configuration, the byte serializer is not used. ■ In 20-bit configuration, the byte serializer is used.	*Byte Serializer* and *Byte Deserializer* sections in the Stratix IV Transceiver Architecture chapter in volume 2 of the *Stratix IV Device Handbook*.
What would you like to base the setting on?	This option is not available for selection in SDI mode.	—
What is the effective data rate?	The effective data rate values are fixed at: ■ 2967 Mbps /2970 Mbps in 3G mode. ■ 1483.5 Mbps/1485 Mbps in HD mode.	—
What is the input clock frequency?	Based on the effective data rate value you enter in the **What is the effective data rate?** field, the ALTGX MegaWizard Plug-In Manager determines the input reference clock frequencies depending on the available multiplier settings.	*CMU PLL and Receiver CDR Input Reference Clock* section in the Stratix IV Transceiver Clocking chapter in volume 2 of the *Stratix IV Device Handbook*.
Specify base data rate.	This option is not available for selection in SDI mode as the data rate is fixed in 3G and HD modes. The ALTGX MegaWizard Plug-In Manager provides you the base data rate options for the VCO in the CMU PLL and receiver CDR.	—

PLL/Ports Screen for SDI Mode

Figure 1–36 shows the **PLL/ports** screen of the ALTGX MegaWizard Plug-In Manager for SDI mode.

Figure 1–36. MegaWizard Plug-In Manager - ALTGX (PLL/Ports Screen for SDI Mode)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip
Word alignment width: 16
Word alignment pattern: 146F

Able to implement the requested GXB

PLL Settings

□ Train Receiver CDR from pll_inclk

What is the TX PLL bandwidth mode? High

What is the Receiver CDR bandwidth mode? Medium

What is the acceptable PPM threshold between the Receiver CDR VCO and the Receiver input reference clock? +/- 1000

Optional Ports

☑ Create 'gxb_powerdown' port to powerdown the Transceiver block

□ Create 'pll_powerdown' port to powerdown the TX PLL

☑ Create 'rx_analogreset' port for the analog portion of the receiver

☑ Create 'rx_digitalreset' port for the digital portion of the receiver

☑ Create 'tx_digitalreset' port for the digital portion of the transmitter

□ Create 'pll_locked' port to indicate PLL is in lock with the reference input clock

□ Create 'rx_locktorefclk' port to lock the RX CDR to the reference clock

□ Create 'rx_locktodata' port to lock the RX CDR to the received data

□ Create 'rx_pll_locked' port to indicate RX CDR is locked to the input reference clock

□ Create 'rx_freqlocked' port to indicate RX CDR is locked to the received data

Table 1–32 describes the available options on the **PLL/ports** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–32. MegaWizard Plug-In Manager Options (PLL/Ports Screen for SDI Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
Train receiver CDR from `pll_inclk`.	If you select this option, the input reference clock to the CMU PLL trains the receiver CDR.	*Table 1-2* in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the TX PLL bandwidth mode?	The appropriate bandwidth settings will be fixed after characterization.	*PLL Bandwidth Setting* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the receiver CDR bandwidth mode?	The appropriate bandwidth settings will be fixed after characterization.	—

Table 1–32. MegaWizard Plug-In Manager Options (PLL/Ports Screen for SDI Mode) (Part 2 of 3)

ALTGX Setting	Description	Reference
What is the acceptable PPM threshold between the receiver CDR VCO and the receiver input reference clock?	In automatic lock mode, the CDR remains in LTD mode as long as the PPM difference between the CDR VCO output clock and the input reference clock is less than the PPM value that you set in this option. If the PPM difference is greater than the PPM value that you set in this option, the CDR switches to LTR mode. The range of values available in this option is ±62.5 ppm to ±1000 ppm. (1)	*Automatic Lock Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `gxb_powerdown` port to power down the transceiver block.	This is an optional signal. When asserted, it powers down the entire transceiver block. If none of the channels are instantiated in a transceiver block, the Quartus II software automatically powers down the entire transceiver block.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `pll_powerdown` port to power down the TX PLL	Each transceiver block has two CMU PLLs. Each CMU PLL has a dedicated power down signal called `pll_powerdown`. This signal powers down the CMU PLL.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_analogreset` port for the analog portion of the receiver.	Receiver analog reset port available in **Receiver only** and **Receiver and Transmitter** operation modes. Resets part of the analog portion of the receiver CDR in the receiver channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_digitalreset` port for the digital portion of the receiver.	Receiver digital reset port available in **Receiver only** and **Receiver and Transmitter** operation modes. Resets the PCS portion of the receiver channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_digitalreset` port for the digital portion of the transmitter.	Transmitter digital reset port available in **Transmitter only** and **Receiver and Transmitter** operation modes. Resets the PCS portion of the transmitter channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `pll_locked` port to indicate PLL is in lock with the input reference clock.	Each CMU PLL has a dedicated `pll_locked` signal that is fed to the FPGA fabric to indicate when the PLL is locked to the input reference clock.	*Transceiver Reset Sequences* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_locktorefclk` port to lock the RX CDR to the input reference clock.	This is an optional port. When this signal is asserted high, the LTR/LTD controller forces the receiver CDR to lock to the phase and frequency of the input reference clock. (1), (2)	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–32. MegaWizard Plug-In Manager Options (PLL/Ports Screen for SDI Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
Create an `rx_locktodata` port to lock the RX CDR to the received data.	This is an optional port. When this signal is asserted high, the LTR/LTD controller forces the receiver CDR to lock to the received data. *(1), (2)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_pll_locked` port to indicate RX CDR is locked to the input reference clock.	In LTR mode, this signal is asserted high to indicate that the receiver CDR has locked to the phase and frequency of the input reference clock. In LTD mode, this signal has no significance. *(1)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_freqlocked` port to indicate RX CDR is locked to the received data.	This signal gets asserted high to indicate that the receiver CDR has switched from LTR to LTD mode. This signal has relevance only in automatic lock mode and may be required to control the transceiver resets, as discussed in the *User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook. (1)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Notes to Table 1–32:

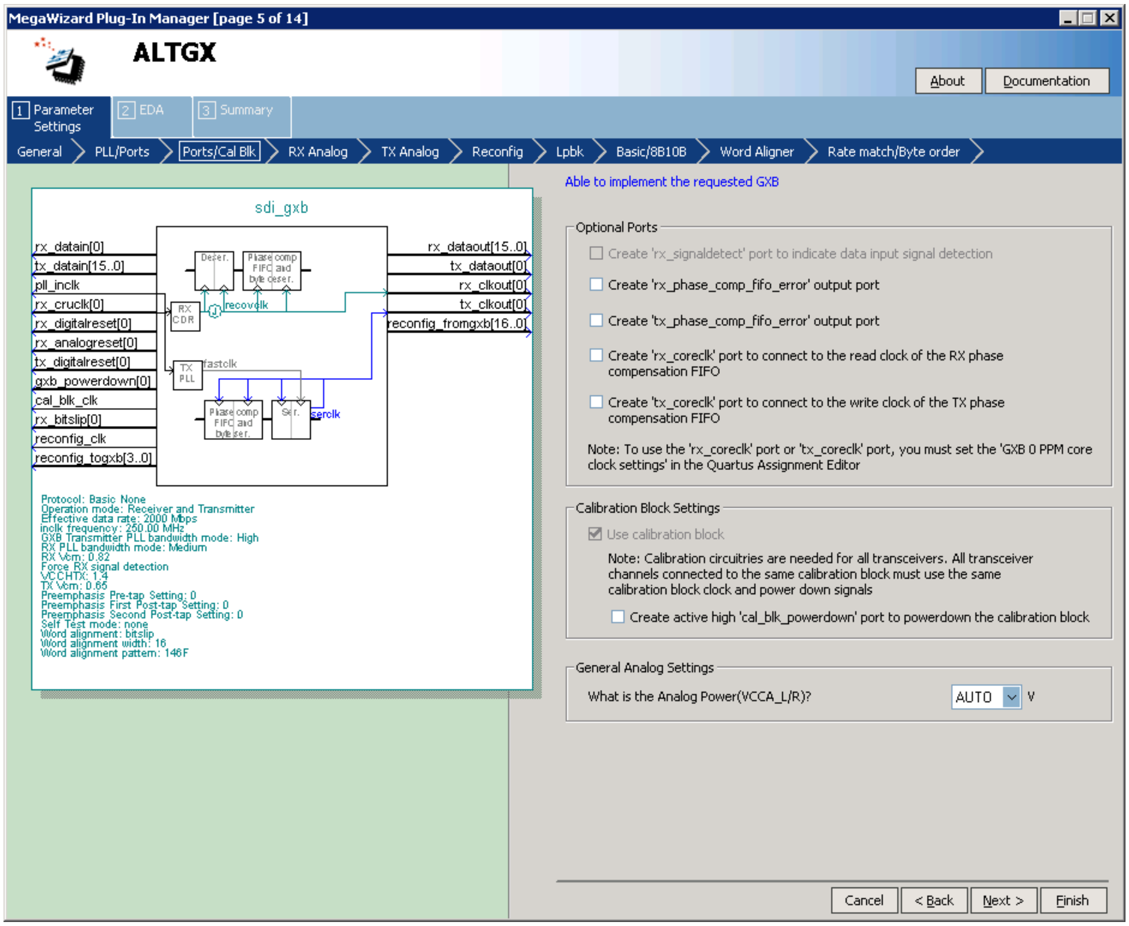
(1) LTR mode is lock-to-reference mode and LTD mode is lock-to-data mode.

(2) When `rx_locktorefclk` and `rx_locktodata` are both asserted high, `rx_locktodata` takes precedence over `rx_locktorefclk`, forcing the CDR to lock to the received data. When both these signals are de-asserted, the LTR/LTD controller is configured in automatic lock mode.

Ports/Cal Blk Screen for SDI Mode

Figure 1–37 shows the **Ports/Cal Blk** screen of the ALTGX MegaWizard Plug-In Manager for SDI mode.

Figure 1–37. MegaWizard Plug-In Manager - ALTGX (Ports/Cal Blk Screen for SDI Mode)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip
Word alignment width: 16
Word alignment pattern: 146F

Able to implement the requested GXB

Optional Ports

- ☐ Create 'rx_signaldetect' port to indicate data input signal detection
- ☐ Create 'rx_phase_comp_fifo_error' output port
- ☐ Create 'tx_phase_comp_fifo_error' output port
- ☐ Create 'rx_coreclk' port to connect to the read clock of the RX phase compensation FIFO
- ☐ Create 'tx_coreclk' port to connect to the write clock of the TX phase compensation FIFO

Note: To use the 'rx_coreclk' port or 'tx_coreclk' port, you must set the 'GXB 0 PPM core clock settings' in the Quartus Assignment Editor

Calibration Block Settings

- ☑ Use calibration block

Note: Calibration circuitries are needed for all transceivers. All transceiver channels connected to the same calibration block must use the same calibration block clock and power down signals

- ☐ Create active high 'cal_blk_powerdown' port to powerdown the calibration block

General Analog Settings

What is the Analog Power(VCCA_L/R)? AUTO V

Table 1–33 describes the available options on the **Ports/Cal Blk** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–33. MegaWizard Plug-In Manager Options (Ports/Cal Blk Screen for SDI Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Create an `rx_signaldetect` port to indicate data input signal detection.	This port is not available in SDI Mode.	*Signal Threshold Detection Circuitry* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create an `rx_phase_comp_fifo_error` output port.	This optional output port indicates receiver phase compensation FIFO overflow or underrun condition.	*Receiver Phase Compensation FIFO Error Flag* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create a `tx_phase_comp_fifo_error` output port.	This optional output port indicates transmitter phase compensation FIFO overflow or underrun condition.	*TX Phase Compensation FIFO Status Signal* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create an `rx_coreclk` port to connect to the read clock of the RX phase compensation FIFO.	The parallel output data from the receiver can be clocked using this optional input port. It allows you to clock the read side of the receiver phase compensation FIFO with a user-provided clock (FPGA fabric clock, FPGA fabric-transceiver interface clock, or input reference clock).	*FPGA Fabric-Transceiver Interface Clocking* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_coreclk` port to connect to the write clock of the TX phase compensation FIFO.	The parallel transmitter data generated in the FPGA fabric can be clocked using this optional input port. It allows you to clock the write side of the transmitter phase compensation FIFO with a user-provided clock (FPGA fabric clock, FPGA fabric-transceiver interface clock, or input reference clock).	*FPGA fabric-Transceiver Interface Clocking* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Use calibration block.	Calibration block is always enabled.	*Calibration Blocks* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

Table 1–33. MegaWizard Plug-In Manager Options (Ports/Cal Blk Screen for SDI Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
Create an active high `cal_blk_powerdown` to power down the calibration block.	Asserting this signal high powers down the calibration block. A high-to-low transition on this signal restarts calibration.	*Input Signals to the Calibration Block* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the analog power ($V_{CCA\_L/R}$)?	The options available for selection are 2.5V, 3.0V and AUTO: 3.0 V : up to 8.5 Gbps 2.5 V : up to 4.25 Gbps AUTO : The ALTGX MegaWizard Plug-In Manager will automatically set V_{CCA} to 2.5 V and V_{CCH} (transmitter buffer voltage) to 1.5 V for the VCO base data rates less than 4.25 Gbps. -OR- V_{CCA} to 3.0 V and V_{CCH} (transmitter buffer voltage) to 1.4 V for the VCO base data rates greater than 4.25 Gbps.	*General Requirements to Combine Channels* section in the *Configuring Multiple Protocols and Data Rates* chapter in volume 2 of the *Stratix IV Device Handbook.*

RX Analog Screen for SDI Mode

Figure 1–6 describes the **RX Analog** screen of the ALTGX MegaWizard Plug-In Manager for the SDI mode. Table 1–4 explains each of the parameter settings available for the **RX Analog** screen.

TX Analog Screen for SDI Mode

Figure 1–7 describes the **TX Analog** screen of the ALTGX MegaWizard Plug-In Manager for the SDI mode. Table 1–5 explains each of the parameter settings available for the **TX Analog** screen.

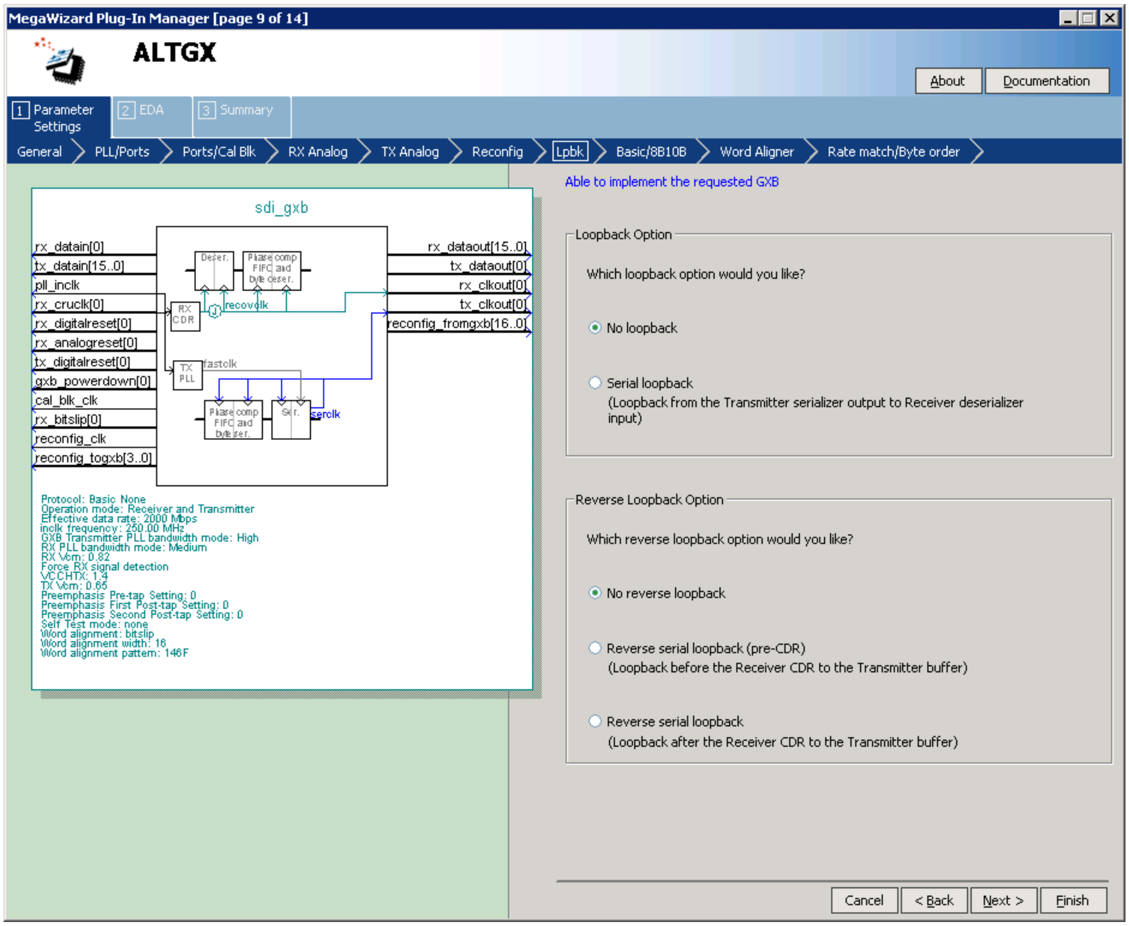
Reconfig Screen for SDI Mode

Figure 1–8 describes the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager for the SDI mode. Table 1–6 explains each of the parameter settings available for the **Reconfig** screen.

Lpbk Screen for SDI Mode

Figure 1–38 shows the **Lpbk** screen of the ALTGX MegaWizard Plug-In Manager for SDI mode.

Figure 1–38. MegaWizard Plug-In Manager - ALTGX (Lpbk Screen)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip
Word alignment width: 16
Word alignment pattern: 146F

Able to implement the requested GXB

Loopback Option

Which loopback option would you like?

● No loopback

○ Serial loopback
(Loopback from the Transmitter serializer output to Receiver deserializer input)

Reverse Loopback Option

Which reverse loopback option would you like?

● No reverse loopback

○ Reverse serial loopback (pre-CDR)
(Loopback before the Receiver CDR to the Transmitter buffer)

○ Reverse serial loopback
(Loopback after the Receiver CDR to the Transmitter buffer)

Table 1–34 describes the available options on the **Lpbk** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

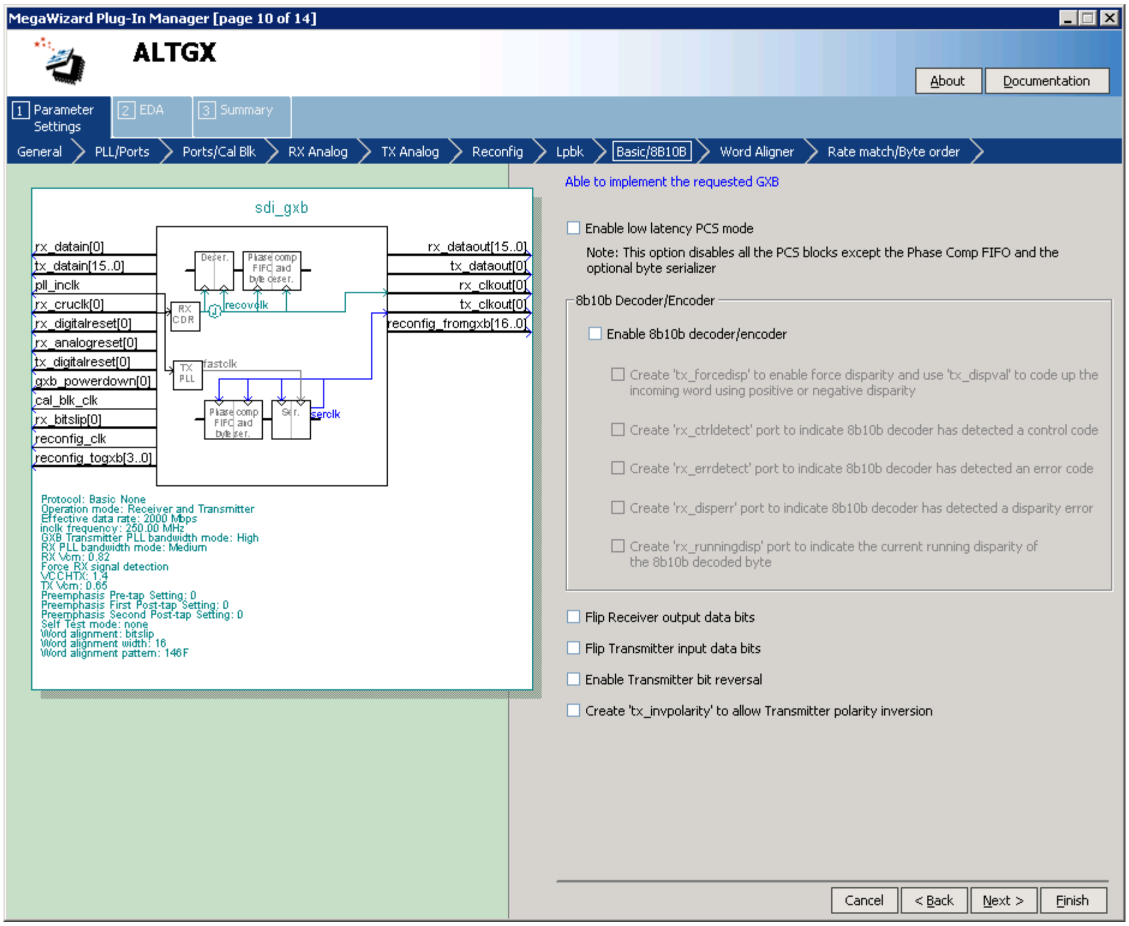
Table 1–34. MegaWizard Plug-In Manager Options (Lpbk Screen for SDI Mode)

ALTGX Setting	Description	Reference
Which loopback option would you like?	There are two options available in SDI mode: ■ No loopback — this is the default mode. ■ Serial loopback — if you select serial loopback, the `rx_seriallpbken` port is available to control the serial loopback feature dynamically. 1'b1 — enables serial loopback 1'b0 — disables serial loopback This signal is asynchronous to the receiver data path.	*Serial Loopback* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Which reverse loopback option would you like?	There are three options available in Basic mode: ■ No reverse loopback — This is the default mode. ■ Reverse Serial loopback (pre-CDR) — This is the loopback before the receiver's CDR block to the transmitter buffer. The receiver path in the PCS is active but the transmitter side is not. ■ Reverse Serial loopback — This is a loopback after the receiver's CDR block to the transmitter buffer. The receiver path in the PCS is active but the transmitter side is not.	*Loopback Modes* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

SDI/8B10B Screen for SDI Mode

Figure 1–39 shows the **SDI/8B10B** screen of the ALTGX MegaWizard Plug-In Manager for SDI mode. If the **Enforce default settings for this protocol** option is selected, this page does not appear in the MegaWizard Plug-In Manager.

Figure 1–39. MegaWizard Plug-In Manager - ALTGX (SDI/8B10B Screen)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip
Word alignment width: 16
Word alignment pattern: 146F

Able to implement the requested GXB

☐ Enable low latency PCS mode

Note: This option disables all the PCS blocks except the Phase Comp FIFO and the optional byte serializer

8b10b Decoder/Encoder

☐ Enable 8b10b decoder/encoder

☐ Create 'tx_forcedisp' to enable force disparity and use 'tx_dispval' to code up the incoming word using positive or negative disparity

☐ Create 'rx_ctrldetect' port to indicate 8b10b decoder has detected a control code

☐ Create 'rx_errdetect' port to indicate 8b10b decoder has detected an error code

☐ Create 'rx_disperr' port to indicate 8b10b decoder has detected a disparity error

☐ Create 'rx_runningdisp' port to indicate the current running disparity of the 8b10b decoded byte

☐ Flip Receiver output data bits

☐ Flip Transmitter input data bits

☐ Enable Transmitter bit reversal

☐ Create 'tx_invpolarity' to allow Transmitter polarity inversion

Table 1–35 describes the available options on the **SDI/8B10B** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–35. MegaWizard Plug-In Manager Options (SDI/8B10B Screen for SDI Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Enable 8B/10B decoder/encoder.	This option is not available in SDI mode.	—
Create a `tx_forcedisp` to enable Force disparity and use `tx_dispval` to code up the incoming word using positive or negative disparity.	This option is not available in SDI mode.	—
Create an `rx_ctrldetect` port to indicate 8B/10B decoder has detected a control code.	This option is not available in SDI mode.	—
Create an `rx_errdetect` port to indicate 8B/10B decoder has detected an error code.	This option is not available in SDI mode.	—

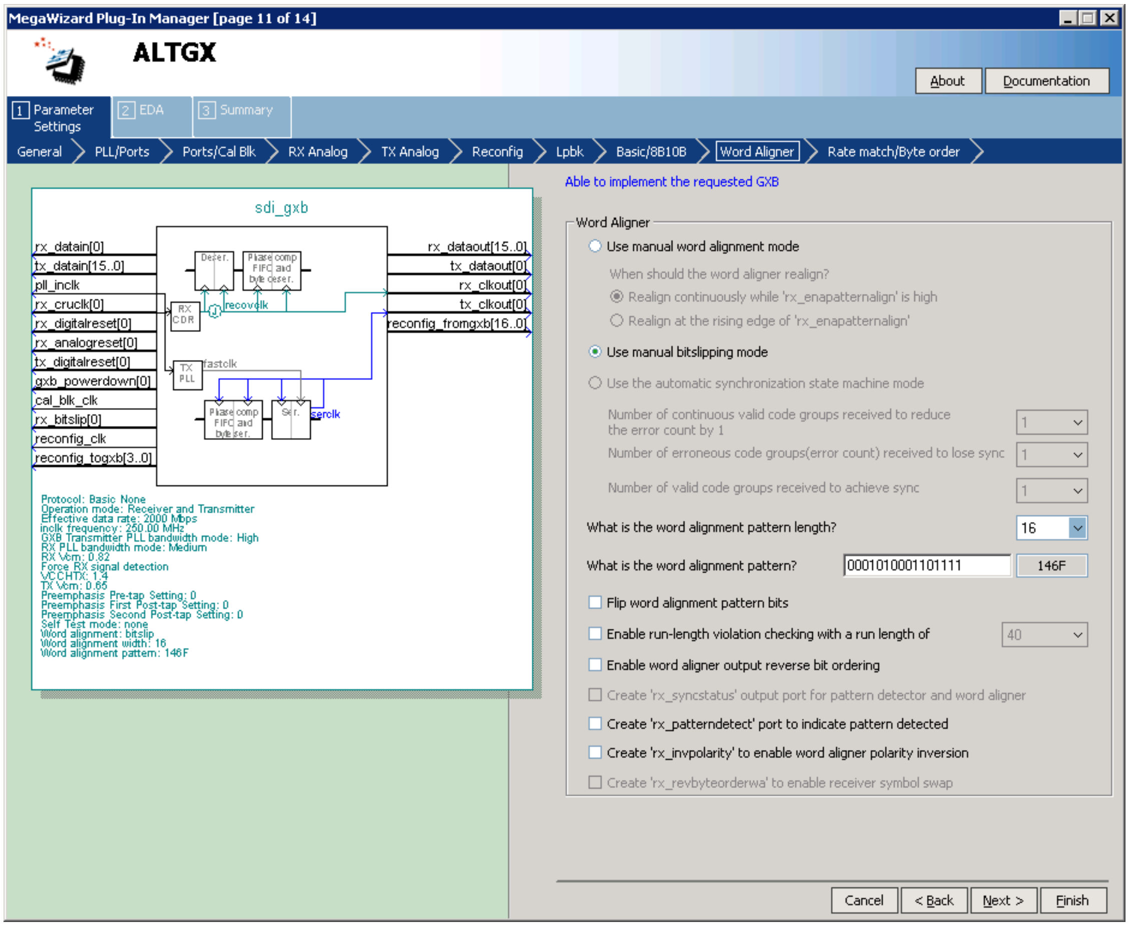
Table 1–35. MegaWizard Plug-In Manager Options (SDI/8B10B Screen for SDI Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
Create an `rx_disperr` port to indicate 8B/10B decoder has detected a disparity code.	This option is not available in SDI mode.	—
Create an `rx_running_disp` port to indicate the current running disparity of the 8B10B decoded byte.	This option is not available in SDI mode.	—
Flip receiver output data bits.	This option reverses the bit order of the parallel receiver data at a byte level at the output of the receiver phase compensation FIFO. For example, if the 20-bit parallel receiver data at the output of the receiver phase compensation FIFO is '1011110000 1010110100' (20'hBC2B4), enabling this option reverses the data on `rx_dataout` port to '0000111101 0010110101' (20'h0F4B5).	—
Flip transmitter input data bits.	This option reverses the bit order of the parallel transmitter data at a byte level at the input of the transmitter phase compensation FIFO. For example, if the 20-bit parallel transmitter data at the `tx_datain` port is '1011110000 1010110100' (20'hBC2B4), enabling this option reverses the input data to the transmitter phase compensation FIFO to '0000111101 0010110101' (20'h0F4B5).	—
Enable transmitter bit reversal.	Enabling this option reverses every bit of the 10-bit parallel data at the input of the serializer. The 10-bit `D[9:0]` gets rewired to `D[0:9]`.	*Transmitter Bit Reversal* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_invpolarity` port to allow Transmitter polarity inversion.	This optional port allows you to dynamically reverse the polarity of every bit of the data word fed to the serializer in the transmitter data path. Use this option when the positive and negative signals of the differential output from the transmitter (`tx_dataout`) are erroneously swapped on the board.	*Transmitter Polarity Inversion* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Word Aligner Screen for SDI Mode

Figure 1–40 shows the **Word Aligner** screen of the ALTGX MegaWizard Plug-In Manager for SDI mode.

Figure 1–40. MegaWizard Plug-In Manager - ALTGX (Word Aligner Screen for SDI Mode)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 2000 Mbps
inclk frequency: 250.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip
Word alignment width: 16
Word alignment pattern: 146F

Able to implement the requested GXB

Word Aligner

○ Use manual word alignment mode

When should the word aligner realign?

● Realign continuously while 'rx_enapatternalign' is high

○ Realign at the rising edge of 'rx_enapatternalign'

● Use manual bitslipping mode

○ Use the automatic synchronization state machine mode

Number of continuous valid code groups received to reduce the error count by 1: 1

Number of erroneous code groups(error count) received to lose sync: 1

Number of valid code groups received to achieve sync: 1

What is the word alignment pattern length? 16

What is the word alignment pattern? 0001010001101111 146F

□ Flip word alignment pattern bits

□ Enable run-length violation checking with a run length of 40

□ Enable word aligner output reverse bit ordering

□ Create 'rx_syncstatus' output port for pattern detector and word aligner

□ Create 'rx_patterndetect' port to indicate pattern detected

□ Create 'rx_invpolarity' to enable word aligner polarity inversion

□ Create 'rx_revbyteorderwa' to enable receiver symbol swap

Table 1–36 describes the available options on the **Word Aligner** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–36. MegaWizard Plug-In Manager Options (Word Aligner Screen for SDI Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
Use manual word alignment mode.	This option is not available in SDI mode.	—
Use manual bitslipping mode.	This option is available in SDI mode. Enabling this option creates an input control signal `rx_bitslip`. In SDI systems because the word alignment and framing happens after de-scrambling, the word aligner in the receiver data path is not useful. Altera recommends driving the ALTGX `rx_bitslip` signal low to prevent the word aligner from inserting bits in the received data stream.	*Receiver Word Alignment and Framing* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–36. MegaWizard Plug-In Manager Options (Word Aligner Screen for SDI Mode) (Part 2 of 3)

ALTGX Setting	Description	Reference
Use the Automatic synchronization state machine mode?	This option is not available in SDI mode.	—
Number of erroneous code groups (error count) received to lose synch.	This option is not available in SDI mode.	—
Number of continuous valid code groups received to reduce the error count by 1?	This option is not available in SDI mode.	—
Number of valid code groups received to achieve synch?	This option is not available in SDI mode.	—
What is the word alignment pattern length?	This option sets the word alignment length. The available choices are 7 and 10 bits. In SDI systems, because word alignment and framing happens after de-scrambling, the word aligner in the receiver data path is not useful.	*Word Aligner* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the word alignment pattern?	The ALTGX MegaWizard Plug-In Manager defaults the word alignment pattern to K28.5-(10'b0101111100) in SDI mode. In SDI systems because word alignment and framing happens after de-scrambling, the word aligner in the receiver data path is not useful.	*Word Aligner* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Flip word alignment pattern bits.	When this option is enabled, the ALTGX MegaWizard Plug-In Manager flips the bit order of the default word alignment pattern in the **What is the word alignment pattern?** option and uses the flipped version as the word alignment pattern. In SDI systems, because word alignment and framing happens after de-scrambling, the word aligner in the receiver data path is not useful.	—
Enable run-length violation checking with a run length of.	This option creates the output signal `rx_rlv`. Enabling this option also activates the run-length violation circuit. If the number of continuous 1's and 0's exceeds the number that you set in this option, the run-length violation circuit asserts the `rx_rlv` signal. The `rx_rlv` signal is asynchronous to the receiver data path and is asserted for a minimum of two recovered clock cycles. For a 10-bit channel width in SDI-3G mode and a 10-bit or 20-bit channel width in SDI-HD mode, the run length limits are 5 to 160 in increments of 5.	*Programmable Run Length Violation Detection* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable word aligner output reverse bit ordering.	In manual bit-slip mode, this option creates an input port `rx_revbitorderwa` to dynamically reverse the bit order at the output of the receiver word aligner. In SDI systems, because word alignment and framing happens after de-scrambling, the word aligner in the receiver data path is not useful.	*Receiver Bit Reversal* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–36. MegaWizard Plug-In Manager Options (Word Aligner Screen for SDI Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
Create an `rx_syncstatus` output port for pattern detector and word aligner.	This option is not available in SDI mode.	—
Create an `rx_patterndetect` port to indicate pattern detected.	This is an output status signal that the word aligner forwards to the FPGA fabric to indicate that the word alignment pattern programmed has been detected in the current word boundary.	*Table 1-24 and Word Aligner* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create an `rx_invpolarity` port to enable word aligner polarity inversion.	This optional port allows you to dynamically reverse the polarity of every bit of the received data at the input of the word aligner. Use this option when the positive and negative signals of the differential input to the receiver (`rx_datain`) are erroneously swapped on the board.	*Receiver Polarity Inversion* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

EDA Screen for SDI Mode

Figure 1–13 describes the **EDA** screen of the MegaWizard Plug-In Manager for SDI mode. The **Generate Netlist** option generates a netlist for the third party EDA synthesis tool to estimate timing and resource utilization for the ALTGX instance.

Summary for SDI Mode

Figure 1–14 describes the **Summary** screen of the MegaWizard Plug-In Manager for SDI mode. You can select optional files on this page. After you make your selections, click **Finish** to generate the files.

XAUI Mode

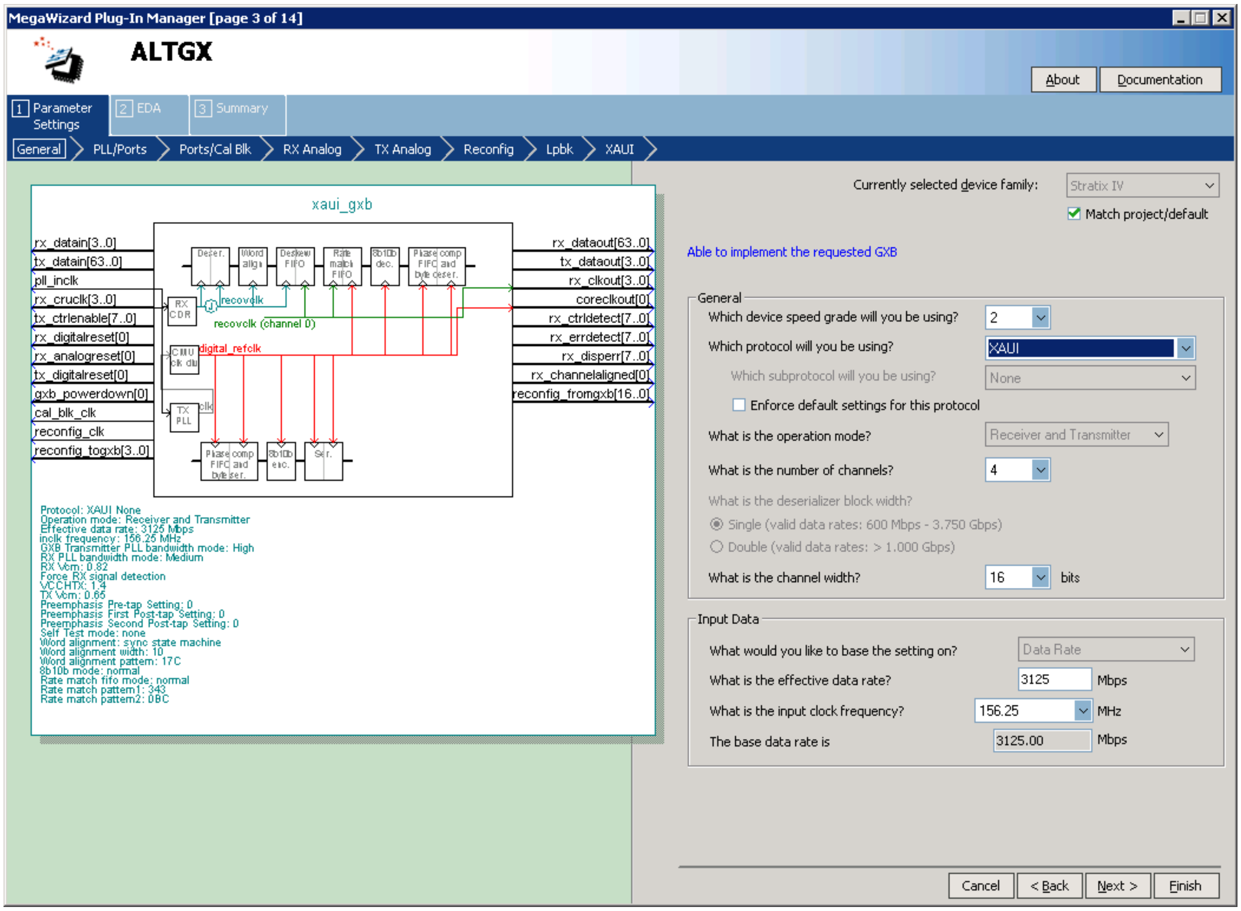
This section provides descriptions of the options available on the individual pages of the ALTGX MegaWizard Plug-In Manager for XAUI mode. The MegaWizard Plug-In Manager provides a warning if any of the settings you choose are illegal.

☞ The word aligner and rate matcher operations and patterns are pre-configured for XAUI mode and cannot be altered.

General Screen for XAUI Mode

Figure 1–41 shows the **General** screen of the ALTGX MegaWizard Plug-In Manager for XAUI mode.

Figure 1–41. MegaWizard Plug-In Manager - ALTGX (General Screen)



Protocol: XAUI None
Operation mode: Receiver and Transmitter
Effective data rate: 3125 Mbps
inclk frequency: 156.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: 343
Rate match pattern2: 0BC

Currently selected device family: Stratix IV

☑ Match project/default

Able to implement the requested GXB

General

Which device speed grade will you be using? 2

Which protocol will you be using? XAUI

Which subprotocol will you be using? None

☐ Enforce default settings for this protocol

What is the operation mode? Receiver and Transmitter

What is the number of channels? 4

What is the deserializer block width?

● Single (valid data rates: 600 Mbps - 3.750 Gbps)

○ Double (valid data rates: > 1.000 Gbps)

What is the channel width? 16 bits

Input Data

What would you like to base the setting on? Data Rate

What is the effective data rate? 3125 Mbps

What is the input clock frequency? 156.25 MHz

The base data rate is 3125.00 Mbps

Table 1–37 describes the available options on the **General** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

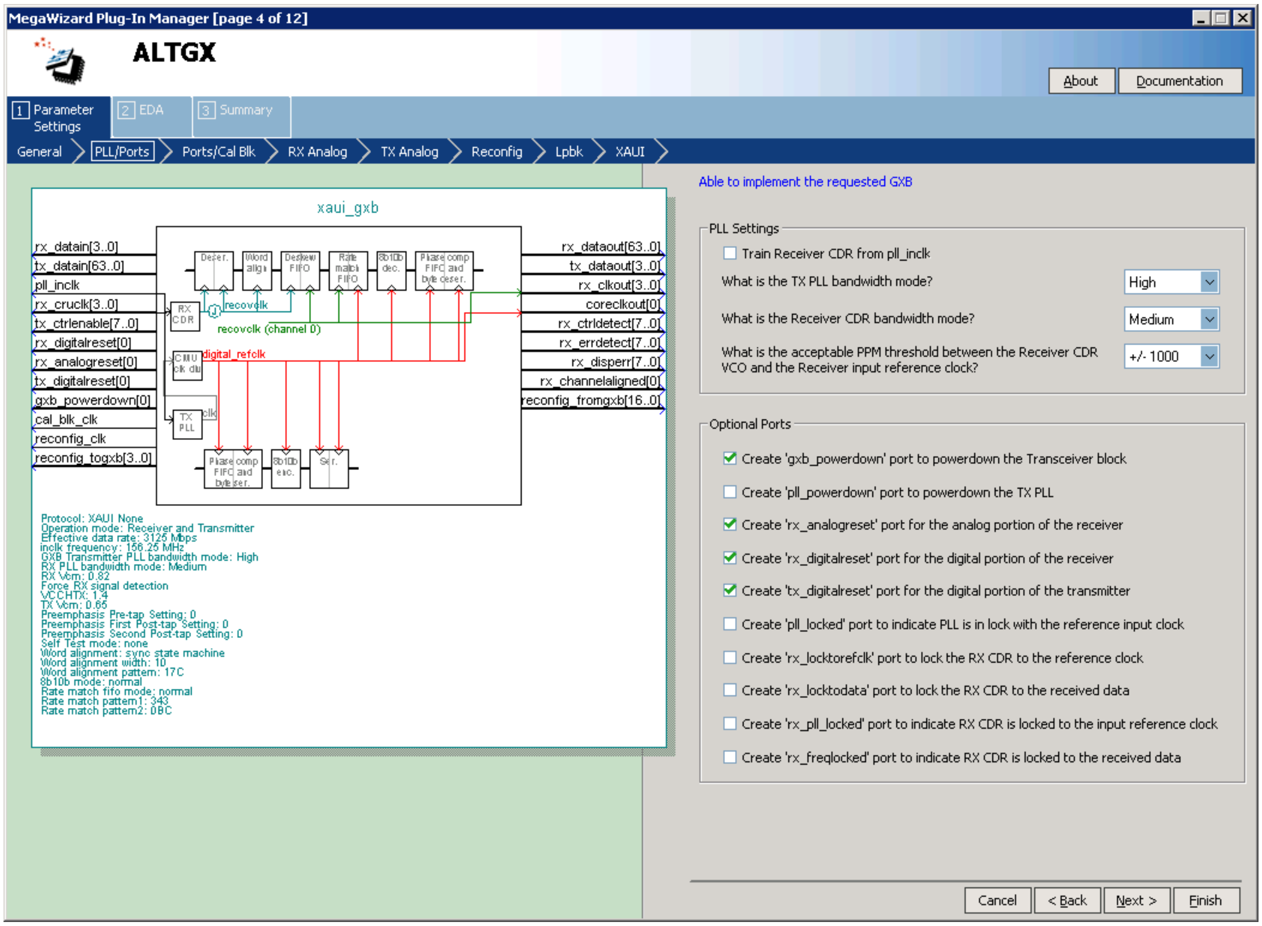
Table 1–37. MegaWizard Plug-In Manager Options (General Screen for XAUI Mode)

ALTGX Setting	Description	Reference
Which device speed grade will you be using?	Select the speed grade of the device you will be using. The available speed grades are 2, 2x, 3, and 4.	—
Which protocol will you be using?	Selects the specific protocol or modes under which the transceiver operates. For XAUI or HiGig, you must select the **XAUI** protocol.	*Table 1-18* in the *DC and Switching Characteristics of the Stratix IV Device Family* chapter in volume 5 of the *Stratix IV Device Handbook* and the *XAUI Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Which subprotocol will you be using?	Not applicable to XAUI mode.	—
Enforce default settings for this protocol.	Selecting this option skips the **XAUI** screen of the XAUI MegaWizard Plug-In Manager. The **XAUI** screen allows you to select the XAUI-specific ports for your design. If you select this option, all XAUI-specific ports are used.	—
What is the operation mode?	Only **Receiver and Transmitter** is allowed in XAUI mode. **Receiver only** and **Transmitter only** modes are not allowed.	—
What is the number of channels?	This selects how many duplicate channels this ALTGX instance contains. In XAUI mode, the number of channels increments by 4.	—
What is the deserializer block width?	XAUI mode only operates in single-width mode. Double-width mode is not allowed.	—
What is the channel width?	This option determines the FPGA fabric-transceiver Interface width. Only 16-bit channel width is allowed in XAUI mode.	*Byte Serializer* and *Byte Deserializer* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What would you like to base the setting on?	This option is not available for selection in XAUI mode.	—
What is the effective data rate?	This option is not available for selection in XAUI mode.	—
What is the input clock frequency?	Based on the effective data rate value in the **What is the effective data rate?** field, the ALTGX MegaWizard Plug-In Manager determines the input reference clock frequencies depending on the available multiplier settings. Enter 3125 Mbps as the transceiver channel serial data rate for XAUI, and 3750 Mbps for HiGig.	*CMU PLL and Receiver CDR Input Reference Clock* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Specify base data rate.	This option is not available for this selection. Based on whether the transceiver is configured in XAUI or HiGig, the ALTGX MegaWizard Plug-In Manager provides you the base data rates options for the VCO in the CMU PLL and receiver CDR.	—

PLL/Ports Screen for XAUI Mode

Figure 1–42 shows the **PLL/Ports** screen of the ALTGX MegaWizard Plug-In Manager for XAUI mode.

Figure 1–42. MegaWizard Plug-In Manager - ALTGX (PLL/Ports Screen)



MegaWizard Plug-In Manager [page 4 of 12]

ALTGX

Able to implement the requested GXB

PLL Settings

□ Train Receiver CDR from pll_inclk

What is the TX PLL bandwidth mode? High

What is the Receiver CDR bandwidth mode? Medium

What is the acceptable PPM threshold between the Receiver CDR VCO and the Receiver input reference clock? +/- 1000

Optional Ports

☑ Create 'gxb_powerdown' port to powerdown the Transceiver block

□ Create 'pll_powerdown' port to powerdown the TX PLL

☑ Create 'rx_analogreset' port for the analog portion of the receiver

☑ Create 'rx_digitalreset' port for the digital portion of the receiver

☑ Create 'tx_digitalreset' port for the digital portion of the transmitter

□ Create 'pll_locked' port to indicate PLL is in lock with the reference input clock

□ Create 'rx_locktorefclk' port to lock the RX CDR to the reference clock

□ Create 'rx_locktodata' port to lock the RX CDR to the received data

□ Create 'rx_pll_locked' port to indicate RX CDR is locked to the input reference clock

□ Create 'rx_freqlocked' port to indicate RX CDR is locked to the received data

Protocol: XAUI None
Operation mode: Receiver and Transmitter
Effective data rate: 3125 Mbps
inclk frequency: 156.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: 343
Rate match pattern2: 0BC

Table 1–38 describes the available options on the **PLL/ports** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–38. MegaWizard Plug-In Manager Options (PLL/Ports Screen for XAUI Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
Train receiver CDR from `pll_inclk`.	If you select this option, the input reference clock to the CMU PLL trains the receiver CDR.	*Table 1-2* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
What is the TX PLL bandwidth mode?	The appropriate bandwidth settings will be fixed after characterization.	*PLL Bandwith Setting* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*
What is the receiver CDR bandwidth mode?	The appropriate bandwidth settings will be fixed after characterization.	—

Table 1–38. MegaWizard Plug-In Manager Options (PLL/Ports Screen for XAUI Mode) (Part 2 of 3)

ALTGX Setting	Description	Reference
What is the acceptable PPM threshold between the receiver CDR VCO and the receiver input reference clock?	In automatic lock mode, the CDR remains in LTD mode as long as the PPM difference between the CDR VCO output clock and the input reference clock is less than the PPM value that you set in this option. If the PPM difference is greater than the PPM value that you set in this option, the CDR switches to LTR mode. The range of values available in this option is ±62.5 ppm to ±1000 ppm. *(1)*	*Automatic Lock Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `gxb_powerdown` port to power down the Transceiver block.	This is an optional signal. When asserted, it powers down the entire transceiver block. If none of the channels are instantiated in a transceiver block, the Quartus II software automatically powers down the entire transceiver block.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `pll_powerdown` port to power down the TX PLL.	Each transceiver block has two CMU PLLs. Each CMU PLL has a dedicated power down signal called `pll_powerdown`. This signal powers down the CMU PLL.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_analogreset` port for the analog portion of the receiver.	Receiver analog reset port available in **Receiver only** and **Receiver and Transmitter** operation modes. Resets part of the analog portion of the receiver CDR in the receiver channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles. The width of this signal depends on the number of channels configured. For example, if the number of channels is 4, this signal is 1-bit wide; if the number channels is 8, the signal is 2-bits wide.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_digitalreset` port for the digital portion of the receiver.	Receiver digital reset port available in **Receiver only** and **Receiver and Transmitter** operation modes. Resets the PCS portion of the receiver channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles. The width of this signal depends on the number of channels configured. For example, if the number of channels is 4, this signal is 1-bit wide; if the number channels is 8, the signal is 2-bits wide.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–38. MegaWizard Plug-In Manager Options (PLL/Ports Screen for XAUI Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
Create a `tx_digitalreset` port for the digital portion of the transmitter.	Transmitter digital reset port available in **Transmitter only** and **Receiver and Transmitter** operation modes. Resets the PCS portion of the transmitter channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles. The width of this signal depends on the number of channels configured. For example, if the number of channels is 4, this signal is 1-bit wide; if the number channels is 8, the signal is 2-bits wide.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `pll_locked` port to indicate PLL is locked to the input reference clock.	Each CMU PLL has a dedicated `pll_locked` signal that is fed to the FPGA fabric to indicate when the PLL is locked to the input reference clock.	*Transceiver Reset Sequences* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_locktorefclk` port to lock the RX CDR to the input reference clock.	This is an optional port. When this signal is asserted high, the LTR/LTD controller forces the receiver CDR to lock to the phase and frequency of the input reference clock. *(1), (2)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_locktodata` port to lock the RX CDR to the received data.	This is an optional port. When this signal is asserted high, the LTR/LTD controller forces the receiver CDR to lock to the received data. *(1), (2)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_pll_locked` port to indicate RX CDR is locked to the input reference clock.	In LTR mode, this signal is asserted high to indicate that the receiver CDR has locked to the phase and frequency of the input reference clock. In LTD mode, this signal has no significance. *(1)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_freqlocked` port to indicate RX CDR is locked to the received data.	This signal gets asserted high to indicate that the receiver CDR has switched from LTR to LTD mode. This signal has relevance only in automatic lock mode and may be required to control the transceiver resets, as discussed in the *User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook. (1)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Notes to Table 1–38:

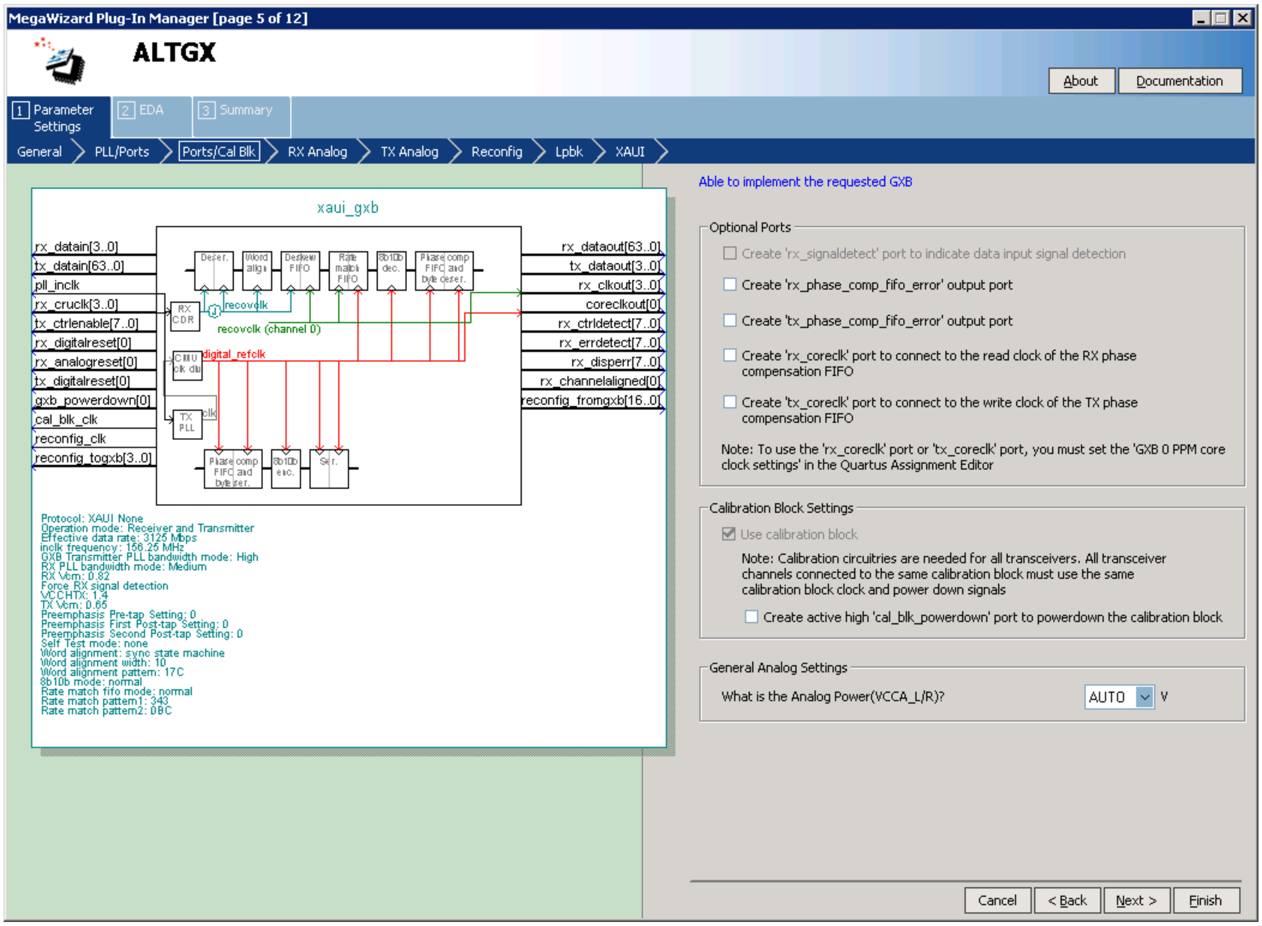
(1) LTR mode is lock-to-reference mode and LTD mode is lock-to-data mode.

(2) When `rx_locktorefclk` and `rx_locktodata` are both asserted high, `rx_locktodata` takes precedence over `rx_locktorefclk`, forcing the CDR to lock to the received data. When both these signals are de-asserted, the LTR/LTD controller is configured in automatic lock mode.

Ports/Cal Blk Screen for XAUI Mode

Figure 1–43 shows the **Ports/Cal Blk** screen of the ALTGX MegaWizard Plug-In Manager for XAUI mode.

Figure 1–43. MegaWizard Plug-In Manager - ALTGX (Ports/Cal Blk Screen for XAUI Mode)



Protocol: XAUI None
Operation mode: Receiver and Transmitter
Effective data rate: 3125 Mbps
inclk frequency: 156.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: 343
Rate match pattern2: 0BC

Able to implement the requested GXB

Optional Ports

- □ Create 'rx_signaldetect' port to indicate data input signal detection
- □ Create 'rx_phase_comp_fifo_error' output port
- □ Create 'tx_phase_comp_fifo_error' output port
- □ Create 'rx_coreclk' port to connect to the read clock of the RX phase compensation FIFO
- □ Create 'tx_coreclk' port to connect to the write clock of the TX phase compensation FIFO

Note: To use the 'rx_coreclk' port or 'tx_coreclk' port, you must set the 'GXB 0 PPM core clock settings' in the Quartus Assignment Editor

Calibration Block Settings

- ☑ Use calibration block

Note: Calibration circuitries are needed for all transceivers. All transceiver channels connected to the same calibration block must use the same calibration block clock and power down signals

- □ Create active high 'cal_blk_powerdown' port to powerdown the calibration block

General Analog Settings

What is the Analog Power(VCCA_L/R)? AUTO V

Table 1–39 describes the available options on the **Ports/Cal Blk** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–39. MegaWizard Plug-In Manager Options (Ports/Cal Blk Screen for XAUI Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Create an `rx_signaldetect` port to indicate data input signal detection.	This port is not available in XAUI mode.	*Signal Threshold Detection Circuitry* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_phase_comp_fifo_error` output port.	This optional output port indicates receiver phase compensation FIFO overflow or underrun condition.	*Receiver Phase Compensation FIFO Error Flag* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–39. MegaWizard Plug-In Manager Options (Ports/Cal Blk Screen for XAUI Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
Create an `tx_phase_comp_fifo_error` output port.	This optional output port indicates transmitter phase compensation FIFO overflow or underrun condition.	*TX Phase Compensation FIFO Status Signal* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_coreclk` port to connect to the read clock of the RX phase compensation FIFO.	The parallel output data from the receiver can be clocked using this optional input port. It allows you to clock the read side of the receiver phase compensation FIFO with a user-provided clock (FPGA fabric clock, FPGA fabric-transceiver interface clock, or input reference clock).	*FPGA Fabric-Transceiver Interface Clocking* section in the *Stratix IV Transceiver Clocking* in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_coreclk` port to connect to the write clock of the TX phase compensation FIFO.	The parallel transmitter data generated in the FPGA fabric can be clocked using this optional input port. It allows you to clock the write side of the transmitter phase compensation FIFO with a user-provided clock (FPGA fabric clock, FPGA fabric-Transceiver interface clock, or input reference clock).	FPGA fabric-Transceiver Interface Clocking section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Use calibration block.	Calibration block is always enabled.	*Calibration Blocks* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an active high `cal_blk_powerdown` to power down the calibration block.	Asserting this signal high powers down the calibration block. A high-to-low transition on this signal restarts calibration.	*Input Signals to the Calibration Block* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the analog power ($V_{CCA\_L/R}$)?	The options available for selection are 2.5V, 3.0V and AUTO: 3.0 V : upto 8.5 Gbps 2.5 V : upto 4.25 Gbps AUTO : The ALTGX MegaWizard Plug-In Manager will automatically set V_{CCA} to 2.5 V and V_{CCH} (transmitter buffer voltage) to 1.5 V for the VCO base data rates less than 4.25 Gbps. -OR- V_{CCA} to 3.0 V and V_{CCH} (transmitter buffer voltage) to 1.4 V for the VCO base data rates greater than 4.25 Gbps.	*General Requirements to Combine Channels* section in the *Configuring Multiple Protocols and Data Rates* chapter in volume 2 of the *Stratix IV Device Handbook.*

RX Analog Screen for XAUI Mode

Figure 1–6 describes the **RX Analog** screen of the ALTGX MegaWizard Plug-In Manager for XAUI mode. Table 1–4 explains each of the parameter settings available for the **RX Analog** screen.

TX Analog Screen for XAUI Mode

Figure 1–7 describes the **TX Analog** screen of the ALTGX MegaWizard Plug-In Manager for XAUI mode. Table 1–5 explains each of the parameter settings available for the **TX Analog** screen.

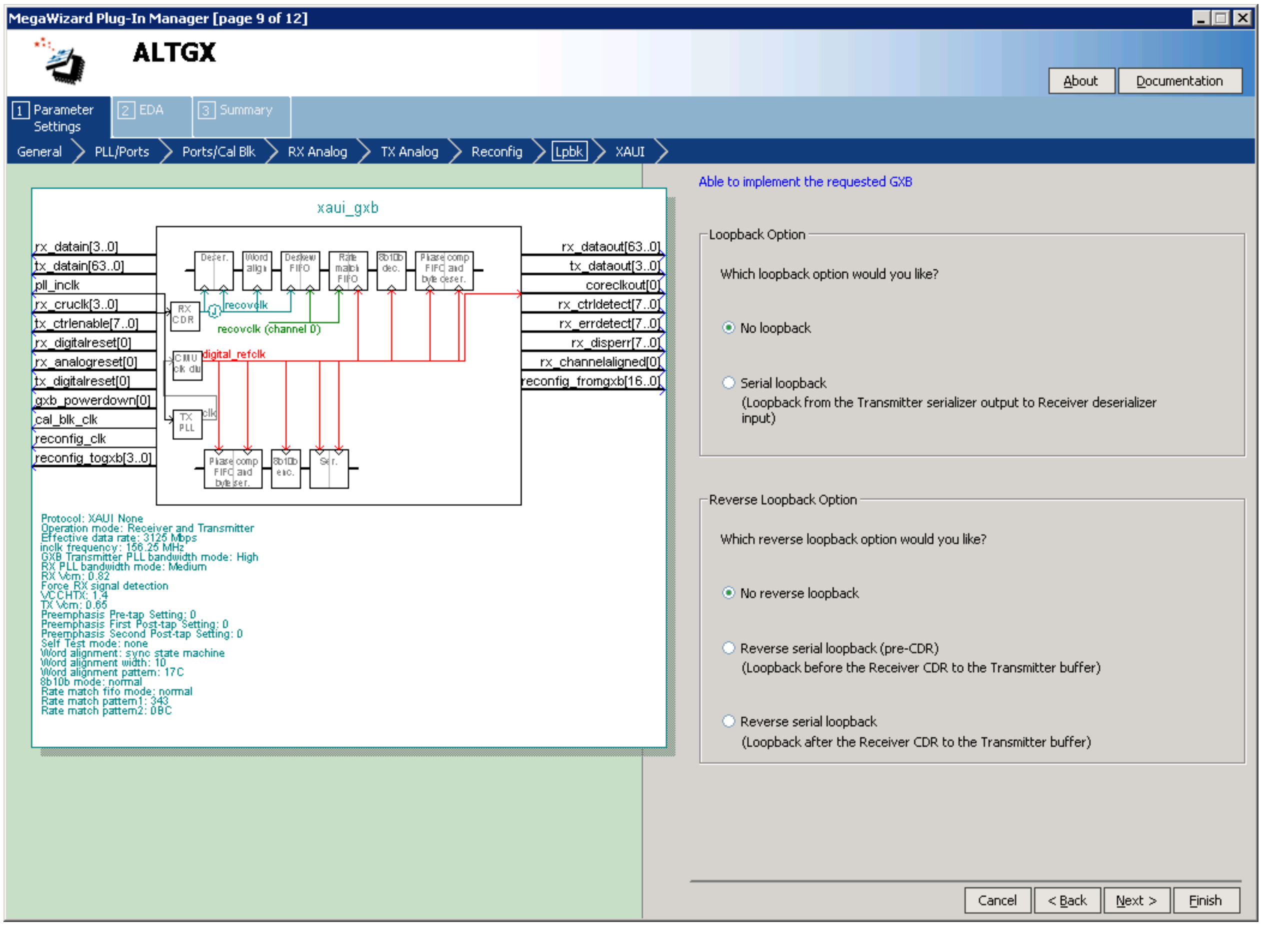
Reconfig Screen for XAUI Mode

Figure 1–8 describes the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager for XAUI mode. Table 1–6 explains each of the parameter settings available for the **Reconfig** screen.

Lpbk Screen for XAUI Mode

Figure 1–44 shows the **Lpbk** screen of the ALTGX MegaWizard Plug-In Manager for XAUI mode.

Figure 1–44. MegaWizard Plug-In Manager - ALTGX (Lpbk Screen for XAUI Mode)



Protocol: XAUI None
Operation mode: Receiver and Transmitter
Effective data rate: 3125 Mbps
inclk frequency: 156.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: 343
Rate match pattern2: 0BC

Able to implement the requested GXB

Loopback Option

Which loopback option would you like?

● No loopback

○ Serial loopback
(Loopback from the Transmitter serializer output to Receiver deserializer input)

Reverse Loopback Option

Which reverse loopback option would you like?

● No reverse loopback

○ Reverse serial loopback (pre-CDR)
(Loopback before the Receiver CDR to the Transmitter buffer)

○ Reverse serial loopback
(Loopback after the Receiver CDR to the Transmitter buffer)

Table 1–40 describes the available options on the **Lpbk** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

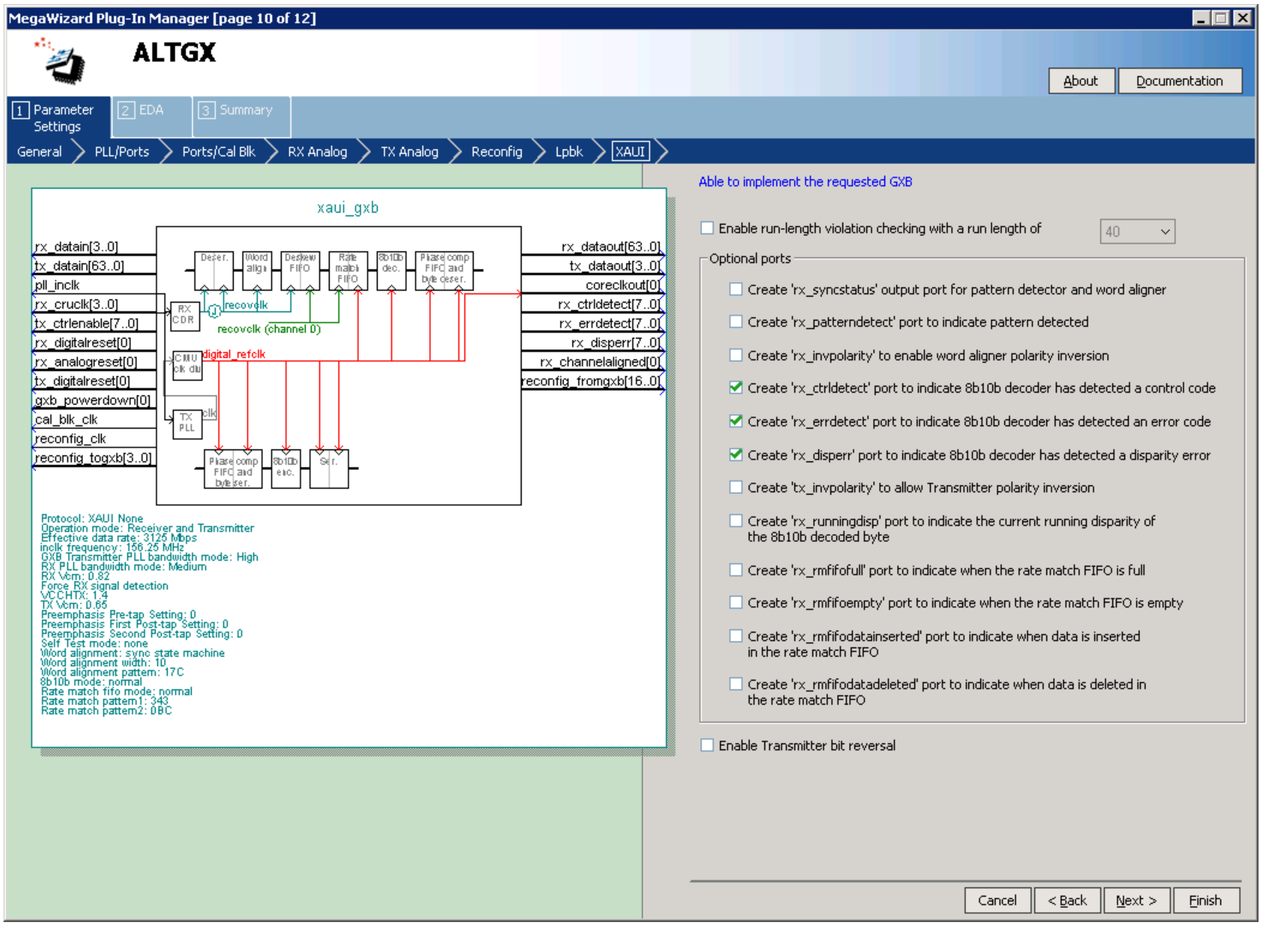
Table 1–40. MegaWizard Plug-In Manager Options (Lpbk Screen for XAUI Mode)

ALTGX Setting	Description	Reference
Which loopback option would you like?	There are two options available in XAUI mode: ■ No loopback — this is the default mode. ■ Serial loopback — if you select serial loopback, the `rx_seriallpbken` port is available to control the serial loopback feature dynamically. 1'b1 — enables serial loopback 1'b0 — disables serial loopback This signal is asynchronous to the receiver data path.	*Serial Loopback* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Which reverse loopback option would you like?	There are three options available in Basic mode: ■ No reverse loopback — This is the default mode. ■ Reverse Serial loopback (pre-CDR) — This is the loopback before the receiver's CDR block to the transmitter buffer. The receiver path in the PCS is active but the transmitter side is not. ■ Reverse Serial loopback — This is a loopback after the receiver's CDR block to the transmitter buffer. The receiver path in the PCS is active but the transmitter side is not.	*Loopback Modes* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

XAUI Screen

Figure 1–45 shows the **XAUI** screen of the ALTGX MegaWizard Plug-In Manager for XAUI mode. If the **Enforce default settings for this protocol** option is selected, this page does not appear in the MegaWizard Plug-In Manager.

Figure 1–45. MegaWizard Plug-In Manager - ALTGX (XAUI Screen)



Protocol: XAUI None
Operation mode: Receiver and Transmitter
Effective data rate: 3125 Mbps
inclk frequency: 156.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: 343
Rate match pattern2: 0BC

Able to implement the requested GXB

□ Enable run-length violation checking with a run length of 40

Optional ports

□ Create 'rx_syncstatus' output port for pattern detector and word aligner

□ Create 'rx_patterndetect' port to indicate pattern detected

□ Create 'rx_invpolarity' to enable word aligner polarity inversion

☑ Create 'rx_ctrldetect' port to indicate 8b10b decoder has detected a control code

☑ Create 'rx_errdetect' port to indicate 8b10b decoder has detected an error code

☑ Create 'rx_disperr' port to indicate 8b10b decoder has detected a disparity error

□ Create 'tx_invpolarity' to allow Transmitter polarity inversion

□ Create 'rx_runningdisp' port to indicate the current running disparity of the 8b10b decoded byte

□ Create 'rx_rmfifofull' port to indicate when the rate match FIFO is full

□ Create 'rx_rmfifoempty' port to indicate when the rate match FIFO is empty

□ Create 'rx_rmfifodatainserted' port to indicate when data is inserted in the rate match FIFO

□ Create 'rx_rmfifodatadeleted' port to indicate when data is deleted in the rate match FIFO

□ Enable Transmitter bit reversal

Table 1–41 describes the available options on the **XAUI** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–41. MegaWizard Plug-In Manager Options (XAUI Screen for XAUI Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
Enable run-length violation checking with a run length of.	This option creates the output signal `rx_rlv`. Enabling this option also activates the run-length violation circuit. If the number of continuous 1's and 0's exceeds the number that you set in this option, the run-length violation circuit asserts the `rx_rlv` signal. The `rx_rlv` signal is asynchronous to the receiver data path and is asserted for a minimum of two recovered clock cycles. For a 16-bit channel width, the run length limits are 5 to 160 in increments of 5.	*Programmable Run Length Violation Detection* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `rx_syncstatus` output port for pattern detector and word aligner.	This is an output status signal that the word aligner forwards to the FPGA fabric to indicate that synchronization has been achieved. This signal is synchronous with the parallel receiver data on the `rx_dataout` port. Receiver synchronization is indicated on the `rx_syncstatus` port of each channel. The `rx_syncstatus` signal is 2-bits wide per channel (8-bits wide per XAUI link).	*Table 1-24 and Automatic Synchronization State Machine Mode Word Aligner in 10-bit PMA-PCS Interface Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `rx_patterndetect` port to indicate pattern detected.	This is an output status signal that the word aligner forwards to the FPGA fabric to indicate that the word alignment pattern programmed has been detected in the current word boundary. The `rx_patterndetect` signal is 2-bits wide per channel (8-bits wide per XAUI link).	*Table 1-24 and Automatic Synchronization State Machine Mode Word Aligner in 10-bit PMA-PCS Interface Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `rx_invpolarity` port to enable word aligner polarity inversion.	This optional port allows you to dynamically reverse the polarity of every bit of the received data at the input of the word aligner. Use this option when the positive and negative signals of the differential input to the receiver (`rx_datain`) are erroneously swapped on the board.	*Receiver Polarity Inversion* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `rx_ctrldetect` port to indicate 8B/10B decoder has detected a control code.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric. This signal indicates whether the decoded 8-bit code group is a data or control code group on this port. If the received 10-bit code group is one of the 12 control code groups (/Kx.y/) specified in IEEE802.3 specification, this signal is driven high. If the received 10-bit code group is a data code group (/Dx.y/), this signal is driven low. The `rx_ctrldetect`signal is 2-bits wide per channel (8-bits wide per XAUI link).	*8B/10B Decoder* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–41. MegaWizard Plug-In Manager Options (XAUI Screen for XAUI Mode) (Part 2 of 3)

ALTGX Setting	Description	Reference
Create an `rx_errdetect` port to indicate 8B/10B decoder has detected an error code.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric. This signal indicates an 8B/10B code group violation. It is asserted high if the received 10-bit code group has a code violation or disparity error. It is used along with the `rx_disperr` signal to differentiate between a code violation error and/or a disparity error.The `rx_errdetect` signal is 2-bits wide per channel (8-bits wide per XAUI link).	*8B/10B Decoder* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_disperr` port to indicate 8B/10B decoder has detected a disparity code.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric.This signal is asserted high if the received 10-bit code or data group has a disparity error. When this signal goes high, `rx_errdetect` also gets asserted high. The `rx_disperr` signal is 2-bits wide per channel (8-bits wide per XAUI link).	*8B/10B Decoder* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_invpolarity` port to allow Transmitter polarity inversion.	This optional port allows you to dynamically reverse the polarity of every bit of the data word fed to the serializer in the transmitter data path. Use this option when the positive and negative signals of the differential output from the transmitter (`tx_dataout`) are erroneously swapped on the board.	*Transmitter Polarity Inversion* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_runningdisp` port to indicate the current running disparity of the 8B/10B decoded byte.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric. This signal is asserted high when the current running disparity of the 8B/10B decoded byte is negative. This signal is low when the current running disparity of the 8B/10B decoded byte is positive.	—
Enable `rx_rmfifofull` flag.	This option creates the output port `rx_rmfifofull`. It is a status flag that the rate match block forwards to the FPGA fabric. It indicates when the rate match FIFO block is full (20 words). This signal remains high as long as the FIFO is full and is asynchronous to the receiver data path. *(2)*	*Rate Match FIFO in XAUI Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable `rx_rmfifoempty` flag.	This option creates the output port `rx_rmfifoempty`. It is a status flag that the rate match block forwards to the FPGA fabric. It indicates when the rate match FIFO block is empty (5 words). This signal remains high as long as the FIFO is empty and is asynchronous to the receiver data path. *(2)*	*Rate Match FIFO in XAUI Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable `rx_rmfifodatainserted` flag.	This option creates the output port `rx_rmfifodatainserted` flag. It is a status flag that the rate match block forwards to the FPGA fabric. If an ||R|| column is inserted, the `rx_rmfifdataoinserted` flag from each of the four channels goes high for one clock cycle per inserted ||R|| column. *(1)*, *(2)*	*Rate Match FIFO in XAUI Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–41. MegaWizard Plug-In Manager Options (XAUI Screen for XAUI Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
Enable `rx_rmfifodatadeleted` flag.	This option creates the output port `rx_rmfifodatadeleted`. It is a status flag that the rate match block forwards to the FPGA fabric. If a ‖R‖ column is deleted, the `rx_rmfifodatadeleted` flag from each of the four channels goes high for one clock cycle per deleted ‖R‖ column. *(1)*, *(2)*	*Rate Match FIFO in XAUI Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable transmitter bit reversal.	Enabling this option reverses every bit of the 10-bit parallel data at the input of the serializer. The 10-bit `D[9:0]` gets reversed to `D[0:9]`.	*8B/10B Encoder* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
What is the word alignment pattern length?	This option sets the word alignment pattern length. The available choices are 7 and 10 for XAUI mode. The default setting for this option is 10.	*XAUI Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Notes for *Table 1–41:*

(1) ‖R‖ column is simultaneous /R/ (/K28.0/) code group on all four channels.

(2) The XAUI protocol requires the transmitter to send an ‖R‖ column during inter-packet gaps (IPG), adhering to rules listed in the IEEE P802.3ae specification. The rate match block looks for the ‖R‖ column and deletes or inserts an ‖R‖ column to prevent the rate match FIFO from overflowing or under running. It can insert or delete as many ‖R‖ columns as necessary to perform the rate match operation.

EDA Screen for XAUI Mode

Figure 1–13 describes the **EDA** screen of the MegaWizard Plug-In Manager for XAUI mode. The **Generate Netlist** option generates a netlist for the third party EDA synthesis tool to estimate timing and resource utilization for the ALTGX instance.

Summary for XAUI Mode

Figure 1–14 describes the **Summary** screen of the MegaWizard Plug-In Manager for XAUI mode. You can select optional files on this page. After you make your selections, click **Finish** to generate the files.

GIGE Mode

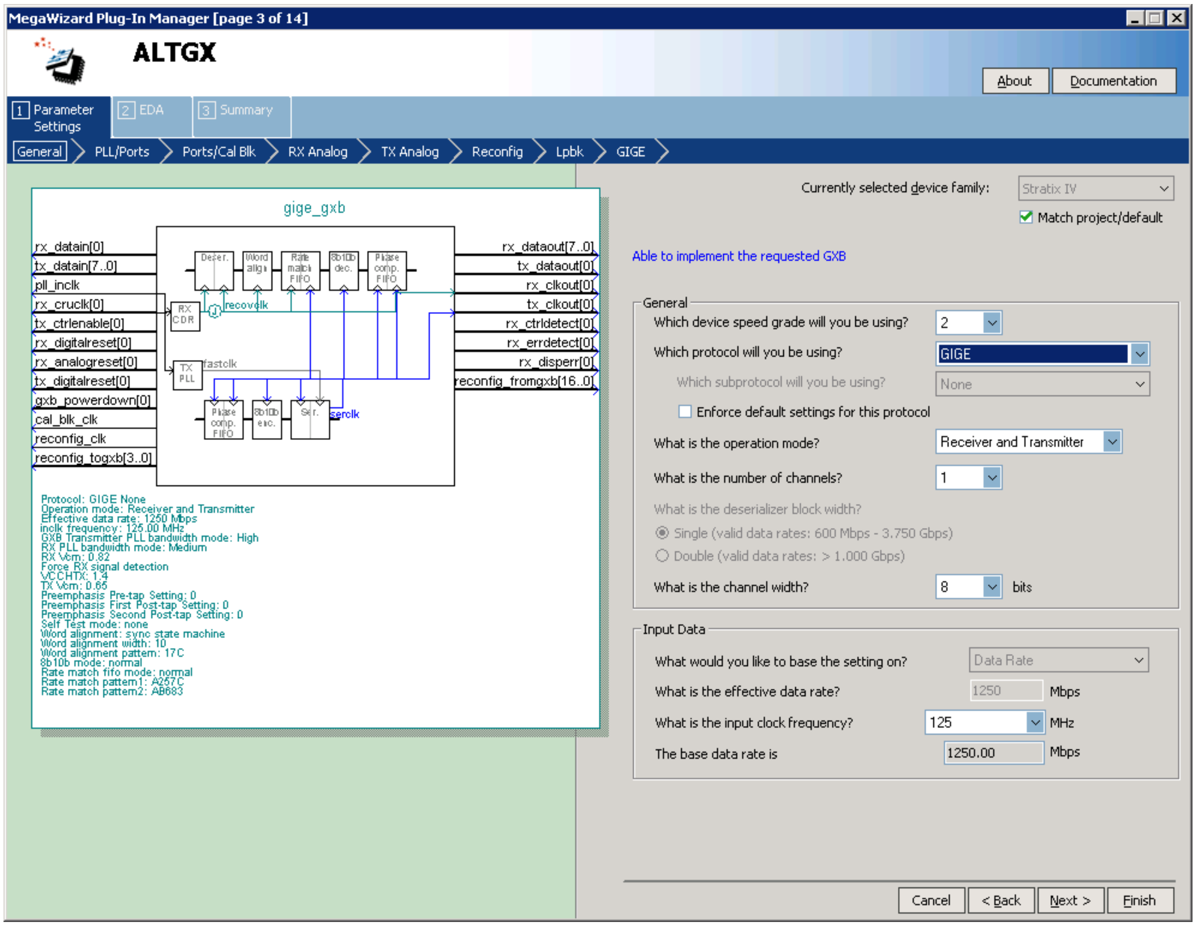
This section provides descriptions of the options available on the individual pages of the ALTGX MegaWizard Plug-In Manager for GIGE mode. The MegaWizard Plug-In Manager provides a warning if any of the settings you choose are illegal.

☞ Word aligner and rate matcher operations and patterns are pre-configured for GIGE mode and cannot be altered.

General Screen for GIGE Mode

Figure 1–46 shows the **General** screen of the ALTGX MegaWizard Plug-In Manager for GIGE mode.

Figure 1–46. MegaWizard Plug-In Manager - ALTGX (General Screen for GIGE Mode)



Protocol: GIGE None
Operation mode: Receiver and Transmitter
Effective data rate: 1250 Mbps
inclk frequency: 125.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: A257C
Rate match pattern2: AB683

Currently selected device family: Stratix IV

☑ Match project/default

Able to implement the requested GXB

General

Which device speed grade will you be using? 2

Which protocol will you be using? GIGE

Which subprotocol will you be using? None

☐ Enforce default settings for this protocol

What is the operation mode? Receiver and Transmitter

What is the number of channels? 1

What is the deserializer block width?

● Single (valid data rates: 600 Mbps - 3.750 Gbps)

○ Double (valid data rates: > 1.000 Gbps)

What is the channel width? 8 bits

Input Data

What would you like to base the setting on? Data Rate

What is the effective data rate? 1250 Mbps

What is the input clock frequency? 125 MHz

The base data rate is 1250.00 Mbps

Table 1–42 describes the available options on the **General** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–42. MegaWizard Plug-In Manager Options (General Screen for GIGE Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Which device speed grade will you be using?	Select the speed grade of the device you will be using. The available speed grades are 2, 2x, 3, and 4.	*Table 1-18* in the *DC and Switching Characteristics of the Stratix IV Device Family* chapter in volume 5 of the *Stratix IV Device Handbook.*
Which protocol will you be using?	Selects the specific protocol or modes under which the transceiver operates. For GIGE mode, you must select the **GIGE** protocol.	*GIGE Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Which subprotocol will you be using?	Not applicable to GIGE mode.	—

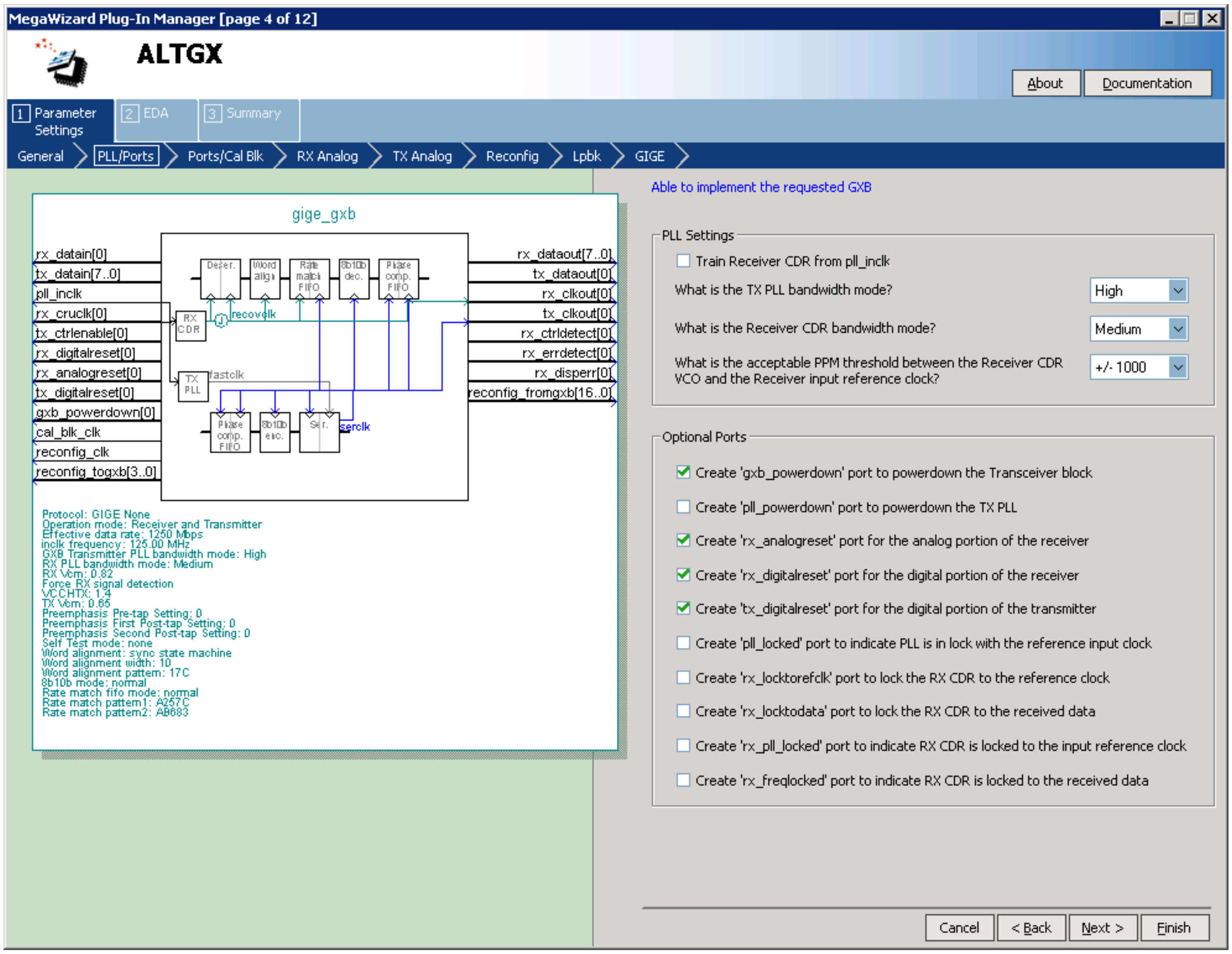
Table 1–42. MegaWizard Plug-In Manager Options (General Screen for GIGE Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
Enforce default settings for this protocol.	Selecting this option skips the **GIGE** screen of the GIGE MegaWizard Plug-In Manager. The **GIGE** screen allows you to select the GIGE-specific ports for your design. If you select this option, all GIGE-specific ports are used.	—
What is the operation mode?	The **Transmitter only** and **Receiver and Transmitter** modes are allowed in GIGE protocol. The **Receiver only** mode is not available.	—
What is the number of channels?	This selects how many duplicate channels this ALTGX instance contains. In GIGE mode, the number of channels increments by one.	—
What is the deserializer block width?	GIGE mode only operates in single-width mode. Double-width mode is not allowed.	—
What is the channel width?	This option determines the FPGA fabric-Transceiver Interface width. In GIGE mode, only 8 bits are allowed.	*Byte Serializer* and *Byte Deserializer* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What would you like to base the setting on?	This option is not available for selection in GIGE mode.	—
What is the effective data rate?	This option is not available for selection in GIGE mode. The transceiver channel serial data rate is fixed to 1250 Mbps in this mode.	—
What is the input clock frequency?	Based on the effective data rate value in the **What is the effective data rate?** field, the ALTGX MegaWizard Plug-In Manager determines the input reference clock frequencies depending on the available multiplier settings.	*CMU PLL and Receiver CDR Input Reference Clock* section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Specify base data rate.	This option is not available for selection because the data rate is fixed in GIGE mode. The ALTGX MegaWizard Plug-In Manager provides you the base data rate options for the VCO in the CMU PLL and receiver CDR in this option.	—

PLL/Ports Screen for GIGE Mode

Figure 1–47 shows the **PLL/ports** screen of the ALTGX MegaWizard Plug-In Manager for GIGE mode.

Figure 1–47. MegaWizard Plug-In Manager - ALTGX (PLL/Ports Screen for GIGE Mode)



Protocol: GIGE None
Operation mode: Receiver and Transmitter
Effective data rate: 1250 Mbps
inclk frequency: 125.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: A257C
Rate match pattern2: AB683

Able to implement the requested GXB

PLL Settings

☐ Train Receiver CDR from pll_inclk

What is the TX PLL bandwidth mode? High

What is the Receiver CDR bandwidth mode? Medium

What is the acceptable PPM threshold between the Receiver CDR VCO and the Receiver input reference clock? +/- 1000

Optional Ports

☑ Create 'gxb_powerdown' port to powerdown the Transceiver block

☐ Create 'pll_powerdown' port to powerdown the TX PLL

☑ Create 'rx_analogreset' port for the analog portion of the receiver

☑ Create 'rx_digitalreset' port for the digital portion of the receiver

☑ Create 'tx_digitalreset' port for the digital portion of the transmitter

☐ Create 'pll_locked' port to indicate PLL is in lock with the reference input clock

☐ Create 'rx_locktorefclk' port to lock the RX CDR to the reference clock

☐ Create 'rx_locktodata' port to lock the RX CDR to the received data

☐ Create 'rx_pll_locked' port to indicate RX CDR is locked to the input reference clock

☐ Create 'rx_freqlocked' port to indicate RX CDR is locked to the received data

Table 1–43 describes the available options on the **PLL/ports** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–43. MegaWizard Plug-In Manager Options (PLL/Ports Screen for GIGE Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
Train receiver CDR from `pll_inclk`.	If you select this option, the input reference clock to the CMU PLL trains the receiver CDR.	*Table 1-2* in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the TX PLL bandwidth mode?	The appropriate bandwidth settings will be fixed after characterization.	*Clock Multiplier Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–43. MegaWizard Plug-In Manager Options (PLL/Ports Screen for GIGE Mode) (Part 2 of 3)

ALTGX Setting	**Description**	**Reference**
What is the receiver CDR bandwidth mode?	The appropriate bandwidth settings will be fixed after characterization.	—
What is the acceptable PPM threshold between the Receiver CDR VCO and the Receiver input reference clock?	In automatic lock mode, the CDR remains in LTD mode as long as the PPM difference between the CDR VCO output clock and the input reference clock is less than the PPM value that you set in this option. If the PPM difference is greater than the PPM value that you set in this option, the CDR switches to LTR mode. The range of values available in this option is ±62.5 ppm to ±1000 ppm. *(1)*	*Automatic Lock Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `gxb_powerdown` port to power down the Transceiver block.	This is an optional signal. When asserted, it powers down the entire transceiver block. If none of the channels are instantiated in a transceiver block, the Quartus II software automatically powers down the entire transceiver block.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `pll_powerdown` port to power down the TX PLL.	Each transceiver block has two CMU PLLs. Each CMU PLL has a dedicated power down signal called `pll_powerdown`. This signal powers down the CMU PLL.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_analogreset` port for the analog portion of the receiver.	Receiver analog reset port available in **Receiver only** and **Receiver and Transmitter** operation modes. Resets part of the analog portion of the receiver CDR in the receiver channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_digitalreset` port for the digital portion of the receiver.	Receiver digital reset port available in **Receiver only** and **Receiver and Transmitter** operation modes. Resets the PCS portion of the receiver channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_digitalreset` port for the digital portion of the transmitter.	Transmitter digital reset port available in **Transmitter only** and **Receiver and Transmitter** operation modes. Resets the PCS portion of the transmitter channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `pll_locked` port to indicate PLL locked to the input reference clock.	Each CMU PLL has a dedicated `pll_locked` signal that is fed to the FPGA fabric to indicate when the PLL is locked to the input reference clock.	*Transceiver Reset Sequences* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–43. MegaWizard Plug-In Manager Options (PLL/Ports Screen for GIGE Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
Create an `rx_locktorefclk` port to lock the RX CDR to the input reference clock.	This is an optional port. When this signal is asserted high, the LTR/LTD controller forces the receiver CDR to lock to the phase and frequency of the input reference clock. *(1)*, *(2)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create an `rx_locktodata` port to lock the RX CDR to the received data.	This is an optional port. When this signal is asserted high, the LTR/LTD controller forces the receiver CDR to lock to the received data. *(1)*, *(2)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create an `rx_pll_locked` port to indicate RX CDR is locked to the input reference clock.	In LTR mode, this signal is asserted high to indicate that the receiver CDR has locked to the phase and frequency of the input reference clock. In LTD mode, this signal has no significance. *(1)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create an `rx_freqlocked` port to indicate RX CDR is locked to the received data.	This signal gets asserted high to indicate that the receiver CDR has switched from LTR to LTD mode. This signal has relevance only in automatic lock mode and may be required to control the transceiver resets, as discussed in the *User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook*. *(1)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

Notes to Table 1–43:

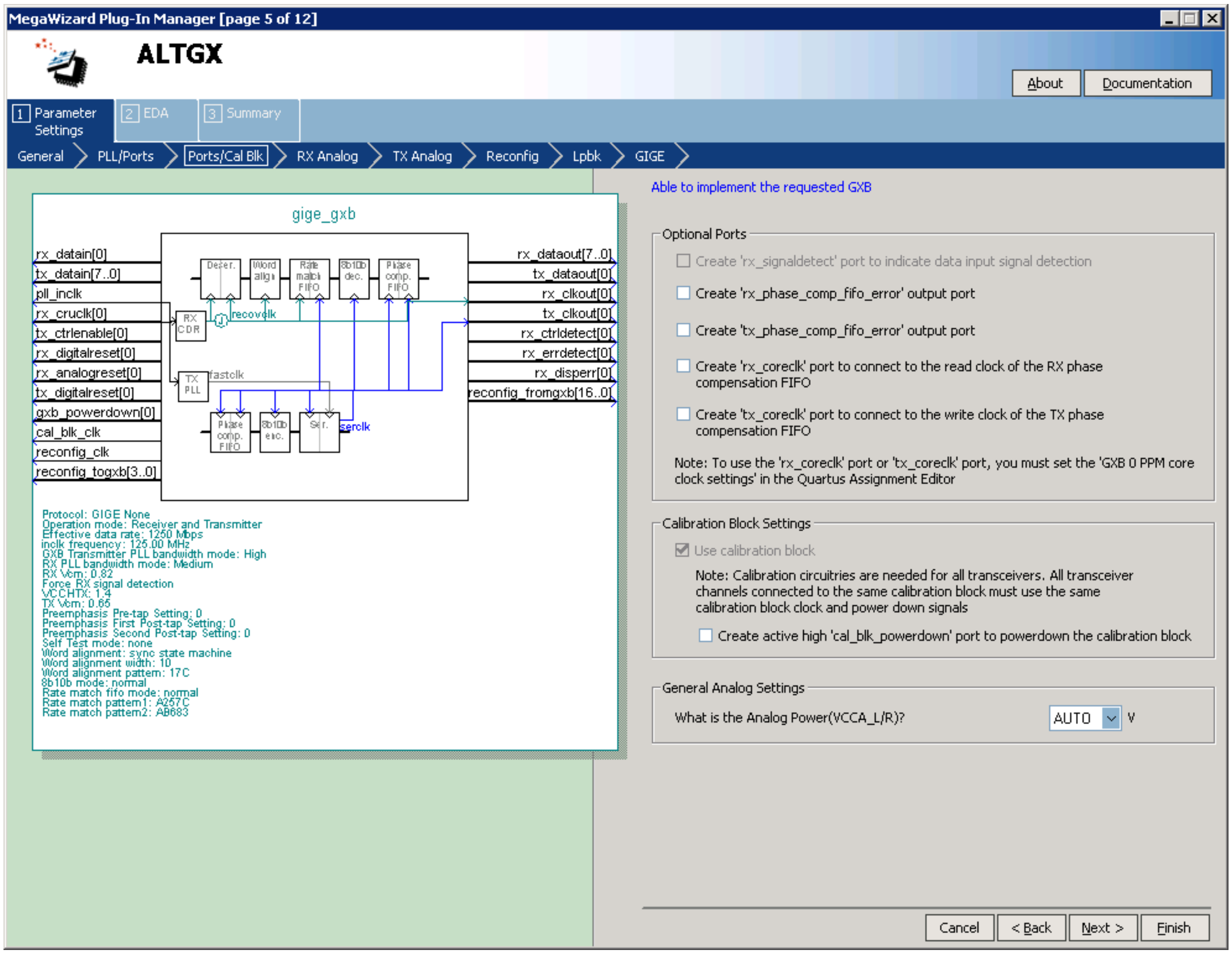
(1) LTR mode is lock-to-reference mode and LTD mode is lock-to-data mode.

(2) When `rx_locktorefclk` and `rx_locktodata` are both asserted high, `rx_locktodata` takes precedence over `rx_locktorefclk`, forcing the CDR to lock to the received data. When both these signals are de-asserted, the LTR/LTD controller is configured in automatic lock mode.

Ports/Cal Blk Screen for GIGE Mode

Figure 1–48 shows the **Ports/Cal Blk** screen of the ALTGX MegaWizard Plug-In Manager for GIGE mode.

Figure 1–48. MegaWizard Plug-In Manager - ALTGX (Ports/Cal Blk Screen for GIGE Mode)



Protocol: GIGE None
Operation mode: Receiver and Transmitter
Effective data rate: 1250 Mbps
inclk frequency: 125.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: A257C
Rate match pattern2: AB683

Able to implement the requested GXB

Optional Ports

☐ Create 'rx_signaldetect' port to indicate data input signal detection

☐ Create 'rx_phase_comp_fifo_error' output port

☐ Create 'tx_phase_comp_fifo_error' output port

☐ Create 'rx_coreclk' port to connect to the read clock of the RX phase compensation FIFO

☐ Create 'tx_coreclk' port to connect to the write clock of the TX phase compensation FIFO

Note: To use the 'rx_coreclk' port or 'tx_coreclk' port, you must set the 'GXB 0 PPM core clock settings' in the Quartus Assignment Editor

Calibration Block Settings

☑ Use calibration block

Note: Calibration circuitries are needed for all transceivers. All transceiver channels connected to the same calibration block must use the same calibration block clock and power down signals

☐ Create active high 'cal_blk_powerdown' port to powerdown the calibration block

General Analog Settings

What is the Analog Power(VCCA_L/R)? AUTO V

Table 1–44 describes the available options on the **Ports/Cal Blk** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–44. MegaWizard Plug-In Manager Options (Ports/Cal Blk Screen for GIGE Mode)

ALTGX Setting	Description	Reference
Create a `rx_signaldetect` port to indicate data input signal detection.	This port is not available in GIGE mode.	*Signal Threshold Detection Circuitry* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create a `rx_phase_comp_fifo_error` output port.	This optional output port indicates Receiver Phase Compensation FIFO overflow or underrun condition.	*Receiver Phase Compensation FIFO Error Flag* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

Table 1–44. MegaWizard Plug-In Manager Options (Ports/Cal Blk Screen for GIGE Mode)

ALTGX Setting	Description	Reference
Create a `tx_phase_comp_fifo_error` output port.	This optional output port indicates Transmitter Phase Compensation FIFO overflow or underrun condition.	*TX Phase Compensation FIFO Status Signal* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `rx_coreclk` port to connect to the read clock of the RX phase compensation FIFO.	The parallel output data from the receiver can be clocked using this optional input port. It allows you to clock the read side of the Receiver Phase Compensation FIFO with a user-provided clock (FPGA fabric clock, FPGA fabric-Transceiver interface clock, or input reference clock).	FPGA fabric-Transceiver Interface Clocking section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_coreclk` port to connect to the write clock of the TX phase compensation FIFO.	The parallel transmitter data generated in the FPGA fabric can be clocked using this optional input port. It allows you to clock the write side of the Transmitter Phase Compensation FIFO with a user-provided clock (FPGA fabric clock, FPGA fabric-Transceiver interface clock, or input reference clock).	FPGA fabric-Transceiver Interface Clocking section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Use calibration block.	Calibration block is always enabled.	*Calibration Blocks* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an active high `cal_blk_powerdown` to power down the calibration block.	Asserting this signal high powers down the calibration block. A high-to-low transition on this signal restarts calibration.	*Input Signals to the Calibration Block* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the analog power ($V_{CCA\_L/R}$)?	The options available for selection are 2.5V, 3.0V and AUTO: 3.0 V : up to 8.5 Gbps 2.5 V : up to 4.25 Gbps AUTO : The ALTGX MegaWizard Plug-In Manager will automatically set V_{CCA} to 2.5 V and V_{CCH} (transmitter buffer voltage) to 1.5 V for the VCO base data rates less than 4.25 Gbps. -OR- V_{CCA} to 3.0 V and V_{CCH} (transmitter buffer voltage) to 1.4 V for the VCO base data rates greater than 4.25 Gbps.	*General Requirements to Combine Channels* section in the *Configuring Multiple Protocols and Data Rates* chapter in volume 2 of the *Stratix IV Device Handbook.*

RX Analog Screen for GIGE Mode

Refer to Figure 1–6 which describes the **RX Analog** screen of the ALTGX MegaWizard Plug-In Manager for GIGE mode. Table 1–4 explains each of the parameter settings available for the **RX Analog** screen.

TX Analog Screen for GIGE Mode

Refer to Figure 1–7 which describes the **TX Analog** screen of the ALTGX MegaWizard Plug-In Manager for GIGE mode. Table 1–5 explains each of the parameter settings available for the **TX Analog** screen.

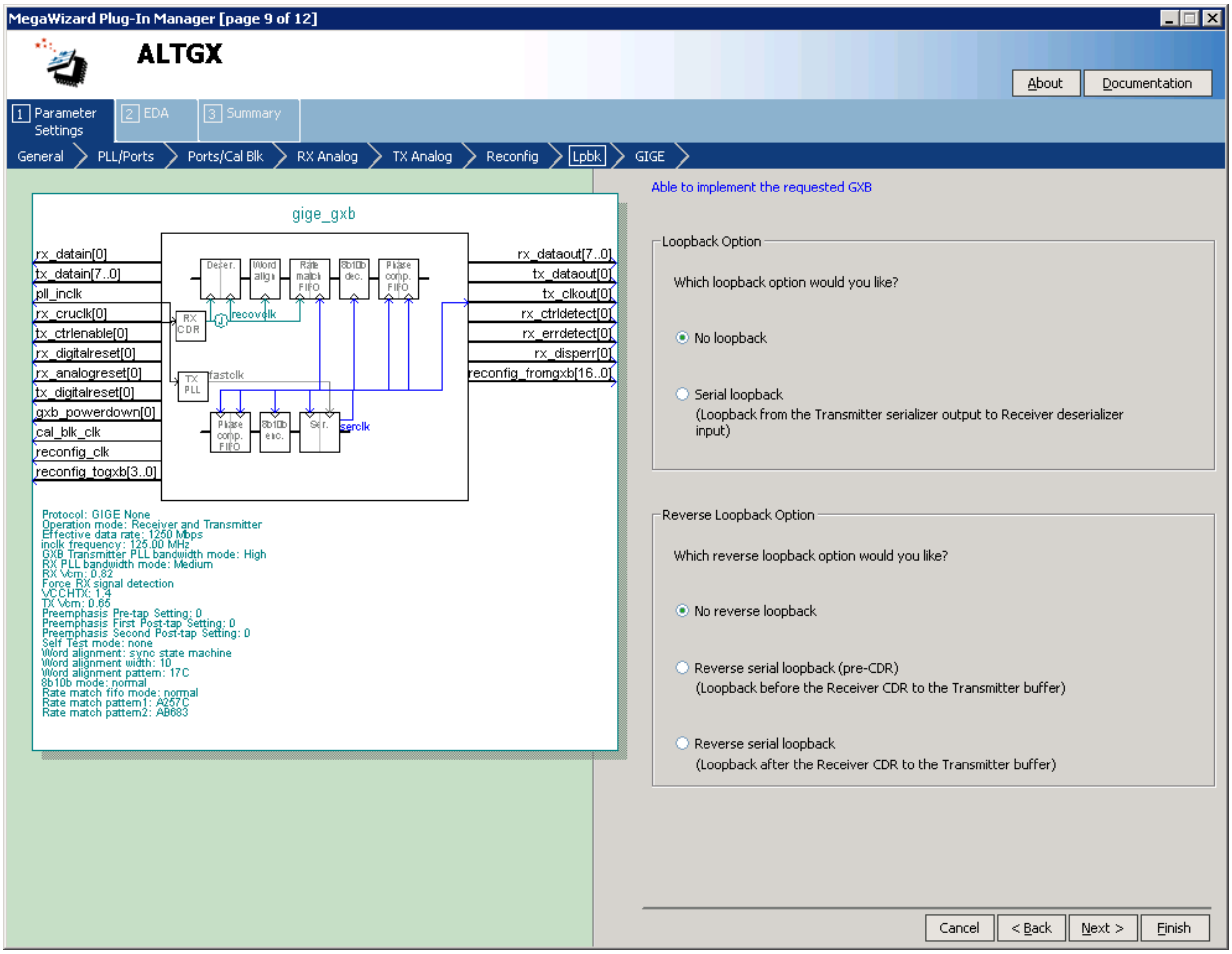
Reconfig Screen for GIGE Mode

Figure 1–8 describes the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager for GIGE mode. Table 1–6 explains each of the parameter settings available for the **Reconfig** screen.

Lpbk Screen for GIGE Mode

Figure 1–49 shows the **Lpbk** screen of the ALTGX MegaWizard Plug-In Manager for GIGE mode.

Figure 1–49. MegaWizard Plug-In Manager - ALTGX (Lpbk Screen for GIGE Mode)



Protocol: GIGE None
Operation mode: Receiver and Transmitter
Effective data rate: 1250 Mbps
inclk frequency: 125.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: A257C
Rate match pattern2: AB683

Able to implement the requested GXB

Loopback Option

Which loopback option would you like?

● No loopback

○ Serial loopback
(Loopback from the Transmitter serializer output to Receiver deserializer input)

Reverse Loopback Option

Which reverse loopback option would you like?

● No reverse loopback

○ Reverse serial loopback (pre-CDR)
(Loopback before the Receiver CDR to the Transmitter buffer)

○ Reverse serial loopback
(Loopback after the Receiver CDR to the Transmitter buffer)

Table 1–45 describes the available options on the **Lpbk** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

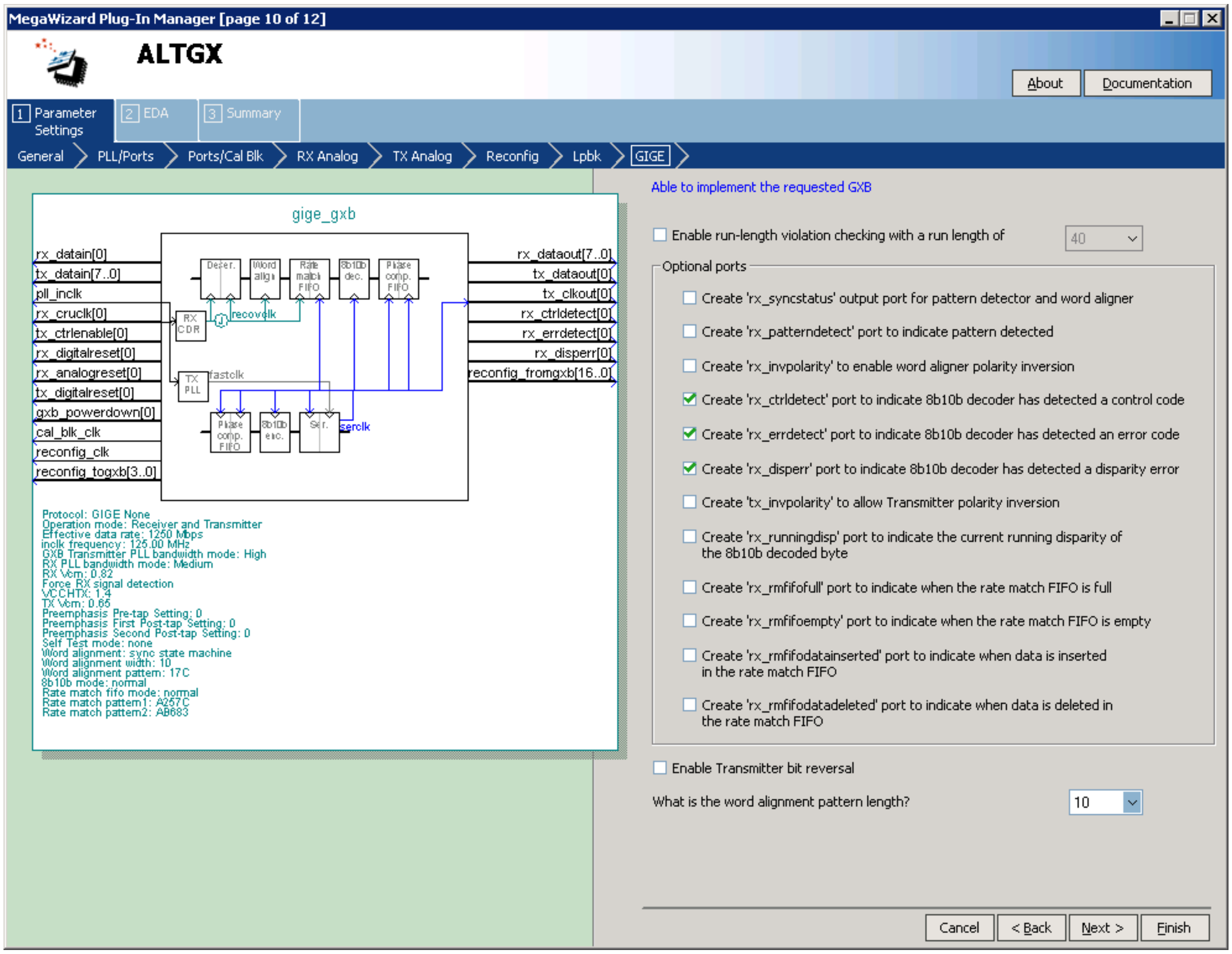
Table 1–45. MegaWizard Plug-In Manager Options (Lpbk Screen for GIGE Mode)

ALTGX Setting	Description	Reference
Which loopback option would you like?	There are two options available in GIGE mode: ■ No loopback — this is the default mode. ■ Serial loopback — if you select serial loopback, the `rx_seriallpbken` port is available to control the serial loopback feature dynamically. 1'b1 — enables serial loopback 1'b0 — disables serial loopback This signal is asynchronous to the receiver data path.	*Serial Loopback* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Which reverse loopback option would you like?	There are three options available in Basic mode: ■ No reverse loopback — This is the default mode. ■ Reverse Serial loopback (pre-CDR) — This is the loopback before the receiver's CDR block to the transmitter buffer. The receiver path in the PCS is active but the transmitter side is not. ■ Reverse Serial loopback — This is a loopback after the receiver's CDR block to the transmitter buffer. The receiver path in the PCS is active but the transmitter side is not.	*Loopback Modes* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

GIGE Screen

Figure 1–50 shows the **GIGE** screen of the ALTGX MegaWizard Plug-In Manager for GIGE mode. If the **Enforce default settings for this protocol** option is selected, this page does not appear in the MegaWizard Plug-In Manager.

Figure 1–50. MegaWizard Plug-In Manager - ALTGX (GIGE Screen)



Protocol: GIGE None
Operation mode: Receiver and Transmitter
Effective data rate: 1250 Mbps
inclk frequency: 125.00 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: sync state machine
Word alignment width: 10
Word alignment pattern: 17C
8b10b mode: normal
Rate match fifo mode: normal
Rate match pattern1: A257C
Rate match pattern2: AB683

Able to implement the requested GXB

□ Enable run-length violation checking with a run length of 40

Optional ports

- □ Create 'rx_syncstatus' output port for pattern detector and word aligner
- □ Create 'rx_patterndetect' port to indicate pattern detected
- □ Create 'rx_invpolarity' to enable word aligner polarity inversion
- ☑ Create 'rx_ctrldetect' port to indicate 8b10b decoder has detected a control code
- ☑ Create 'rx_errdetect' port to indicate 8b10b decoder has detected an error code
- ☑ Create 'rx_disperr' port to indicate 8b10b decoder has detected a disparity error
- □ Create 'tx_invpolarity' to allow Transmitter polarity inversion
- □ Create 'rx_runningdisp' port to indicate the current running disparity of the 8b10b decoded byte
- □ Create 'rx_rmfifofull' port to indicate when the rate match FIFO is full
- □ Create 'rx_rmfifoempty' port to indicate when the rate match FIFO is empty
- □ Create 'rx_rmfifodatainserted' port to indicate when data is inserted in the rate match FIFO
- □ Create 'rx_rmfifodatadeleted' port to indicate when data is deleted in the rate match FIFO

□ Enable Transmitter bit reversal

What is the word alignment pattern length? 10

Table 1–46 describes the available options on the **GIGE** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–46. MegaWizard Plug-In Manager Options (GIGE Screen for GIGE Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
Enable run-length violation checking with a run length of.	This option creates the output signal `rx_rlv`. Enabling this option also activates the run-length violation circuit. If the number of continuous 1's and 0's exceeds the number that you set in this option, the run-length violation circuit asserts the `rx_rlv` signal. The `rx_rlv` signal is asynchronous to the receiver data path and is asserted for a minimum of two recovered clock cycles. For an 8-bit channel width, the run length limits are 5 to 160 in increments of 5.	*Programmable Run Length Violation Detection* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_syncstatus` output port for pattern detector and word aligner.	This is an output status signal that the word aligner forwards to the FPGA fabric to indicate that synchronization has been achieved. This signal is synchronous with the parallel receiver data on the `rx_dataout` port. Receiver synchronization is indicated on the `rx_syncstatus` port of each channel.	*Table 1-24 and Automatic Synchronization State Machine Mode Word Aligner in 10-bit PMA-PCS Interface Mode* in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_patterndetect` port to indicate pattern detected.	This is an output status signal that the word aligner forwards to the FPGA fabric to indicate that the word alignment pattern programmed has been detected in the current word boundary.	*Table 1-24 and Automatic Synchronization State Machine Mode Word Aligner in 10-bit PMA-PCS Interface Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_invpolarity` port to enable word aligner polarity inversion.	This optional port allows you to dynamically reverse the polarity of every bit of the received data at the input of the word aligner. Use this option when the positive and negative signals of the differential input to the receiver (`rx_datain`) are erroneously swapped on the board.	*Receiver Polarity Inversion* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_ctrldetect` port to indicate 8B/10B decoder has detected a control code.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric. This signal indicates whether the decoded 8-bit code group is a data or control code group on this port. If the received 10-bit code group is one of the 12 control code groups (/Kx.y/) specified in IEEE802.3 specification, this signal is driven high. If the received 10-bit code group is a data code group (/Dx.y/), this signal is driven low.	*8B/10B Decoder* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–46. MegaWizard Plug-In Manager Options (GIGE Screen for GIGE Mode) (Part 2 of 3)

ALTGX Setting	Description	Reference
Create an `rx_errdetect` port to indicate 8B/10B decoder has detected an error code.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric. This signal indicates an 8B/10B code group violation. It is asserted high if the received 10-bit code group has a code violation or disparity error. It is used along with the `rx_disperr` signal to differentiate between a code violation error and/or a disparity error.	*8B/10B Decoder* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_disperr` port to indicate 8B/10B decoder has detected a disparity error.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric.This signal is asserted high if the received 10-bit code or data group has a disparity error. When this signal goes high, `rx_errdetect` also gets asserted high.	*8B/10B Decoder* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_invpolarity` port to allow Transmitter polarity inversion.	This optional port allows you to dynamically reverse the polarity of every bit of the data word fed to the serializer in the transmitter data path. Use this option when the positive and negative signals of the differential output from the transmitter (`tx_dataout`) are erroneously swapped on the board.	*Transmitter Polarity Inversion* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_runningdisp` port to indicate the current running disparity of the 8B/10B decoded byte.	This is an output status signal that the 8B/10B decoder forwards to the FPGA fabric. This signal is asserted high when the current running disparity of the 8B/10B decoded byte is negative. This signal is low when the current running disparity of the 8B/10B decoded byte is positive.	—
Enable `rx_rmfifofull` flag.	This option creates the output port `rx_rmfifofull`. It is a status flag that the rate match block forwards to the FPGA fabric. It indicates when the rate match FIFO block is full (20 words). This signal remains high as long as the FIFO is full and is asynchronous to the receiver data path.	*Rate Match FIFO in GIGE Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable `rx_rmfifoempty` flag.	This option creates the output port `rx_rmfifoempty`. It is a status flag that the rate match block forwards to the FPGA fabric. It indicates when the rate match FIFO block is empty (5 words). This signal remains high as long as the FIFO is empty and is asynchronous to the receiver data path.	*Rate Match FIFO in GIGE Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable `rx_rmfifodatainserted` flag.	This option creates the output port `rx_rmfifodatainserted` flag. It is a status flag that the rate match block forwards to the FPGA fabric. The `rx_rmfifodatainserted` flag is asserted for two clock cycles for each insertion of the first two bytes of the /C2/ ordered-set.	*Rate Match FIFO in GIGE Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–46. MegaWizard Plug-In Manager Options (GIGE Screen for GIGE Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
Enable `rx_rmfifodatadeleted` flag.	This option creates the output port `rx_rmfifodatadeleted`. It is a status flag that the rate match block forwards to the FPGA fabric. The `rx_rmfifodatadeleted` flag is asserted for two clock cycles for each deletion of the first two bytes of the /C2/ ordered-set.	*Rate Match FIFO in GIGE Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable transmitter bit reversal.	Enabling this option reverses every bit of the 10-bit parallel data at the input of the serializer. The 10-bit input to the serializer `D[9:0]` gets reversed to `D[0:9]`.	*8B/10B Encoder* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the word alignment pattern length?	This option sets the word alignment pattern length. The available choices are 7 and 10 for GIGE mode. The default setting for this option is **10**.	*GIGE Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

EDA Screen for GIGE Mode

Figure 1–13 describes the **EDA** screen of the MegaWizard Plug-In Manager for GIGE mode. The **Generate Netlist** option generates a netlist for the third party EDA synthesis tool to estimate timing and resource utilization for the ALTGX instance.

Summary for GIGE Mode

Figure 1–14 describes the **Summary** screen of the MegaWizard Plug-In Manager for GIGE mode. You can select optional files on this page. After you make your selections, click **Finish** to generate the files.

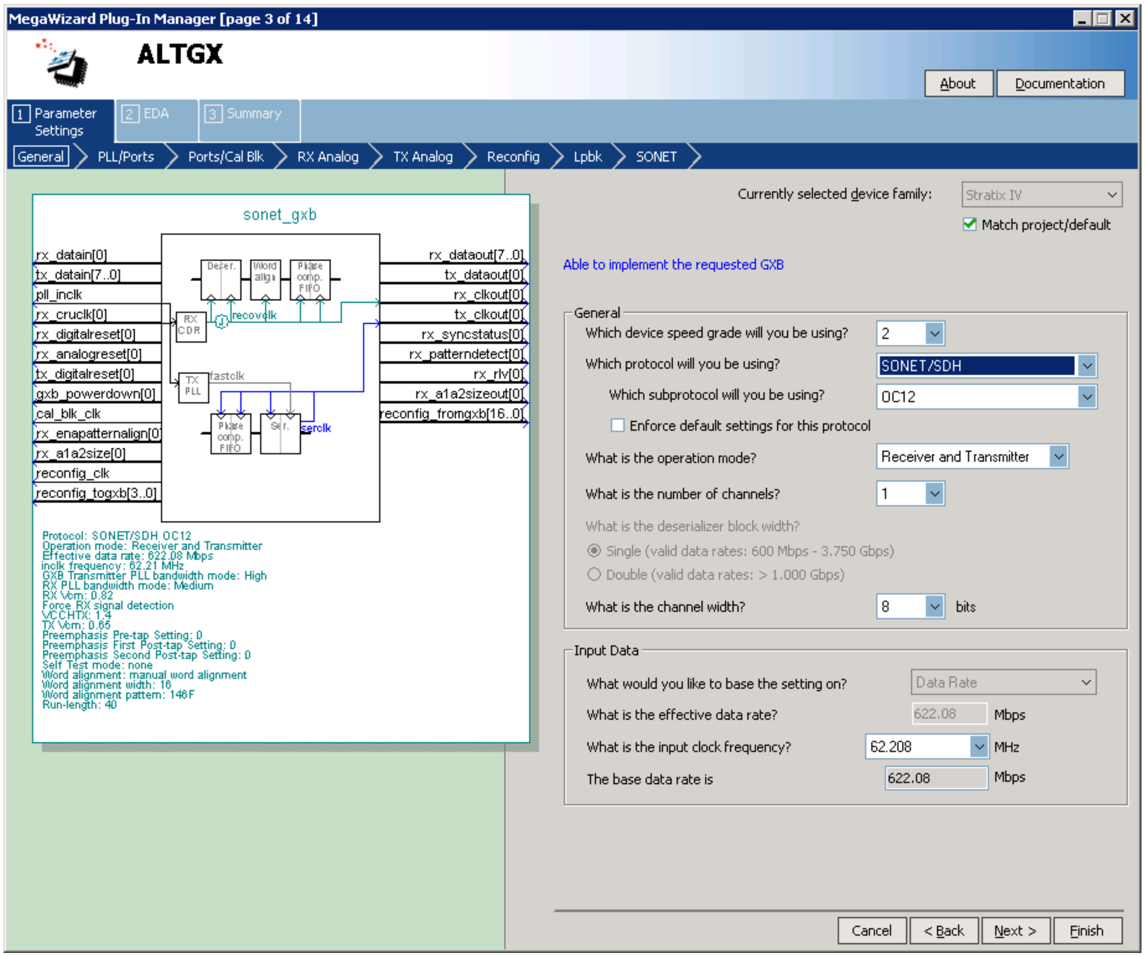
SONET/SDH Mode

This section provides descriptions of the options available on the individual pages of the ALTGX MegaWizard Plug-In Manager for SONET/SDH mode. The MegaWizard Plug-In Manager provides a warning if any of the settings you choose are illegal.

General Screen for SONET/SDH Mode

Figure 1–51 shows the **General** screen of the ALTGX MegaWizard Plug-In Manager for SONET/SDH mode.

Figure 1–51. MegaWizard Plug-In Manager - ALTGX (General Screen for SONET/SDH Mode)



Protocol: SONET/SDH OC12
Operation mode: Receiver and Transmitter
Effective data rate: 622.08 Mbps
inclk frequency: 62.21 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: manual word alignment
Word alignment width: 16
Word alignment pattern: 146F
Run-length: 40

Currently selected device family: Stratix IV
☑ Match project/default

Able to implement the requested GXB

General

Which device speed grade will you be using? 2

Which protocol will you be using? SONET/SDH

Which subprotocol will you be using? OC12

□ Enforce default settings for this protocol

What is the operation mode? Receiver and Transmitter

What is the number of channels? 1

What is the deserializer block width?

● Single (valid data rates: 600 Mbps - 3.750 Gbps)

○ Double (valid data rates: > 1.000 Gbps)

What is the channel width? 8 bits

Input Data

What would you like to base the setting on? Data Rate

What is the effective data rate? 622.08 Mbps

What is the input clock frequency? 62.208 MHz

The base data rate is 622.08 Mbps

Table 1–47 describes the available options on the **General** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–47. MegaWizard Plug-In Manager Options (General Screen for SONET/SDH Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
Which device speed grade will you be using?	Select the speed grade of the device you will be using. The available speed grades are 2, 2x, 3, and 4.	—
Which protocol will you be using?	Selects the specific protocol or modes that the transceiver operates under. For the SONET/SDH mode, you must select the **SONET/SDH** protocol.	*Table 1-18* in the *DC and Switching Characteristics of the Stratix IV Device Family* chapter in volume 5 of the *Stratix IV Device Handbook* and *SONET/SDH Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

Table 1–47. MegaWizard Plug-In Manager Options (General Screen for SONET/SDH Mode) (Part 2 of 3)

ALTGX Setting	Description	Reference
Which subprotocol will you be using?	There are three subprotocols allowed in SONET/SDH mode: OC-12, OC-48, and OC-96. The supported data rates are as follows: ■ OC-12 – 622 Mbps ■ OC-48 – 2488.32 Mbps ■ OC-96 – 4976 Mbps	—
Enforce default settings for this protocol.	Selecting this option skips the **SONET** screen of the SONET/SDH MegaWizard Plug-In Manager. The **SONET** screen allows you to select which SONET/SDH-specific port and word alignment pattern you want to use. If you select this option, all SONET/SDH-specific ports are used and the defaulted alignment pattern is locked at 16'hF628.	—
What is the operation mode?	The **Transmitter only**, **Receiver only**, and **Receiver and Transmitter** modes are allowed in the SONET/SDH protocol.	—
What is the number of channels?	This selects how many duplicate channels this ALTGX instance contains. In SONET/SDH mode, the number of channels increments by one.	—
What is the deserializer block width?	This option sets the transceiver data path width. ■ Single-width – This option is selected automatically in OC-12 and OC-48 configurations. The transceiver data path width is 8 bits. ■ Double-width – This option is selected automatically in OC-96 configurations. The transceiver data path width is 16bits.	—
What is the channel width?	This option selects the FPGA fabric-Transceiver interface width. Depending on the subprotocol selection, choose one of the following: ■ 8 bits for OC-12 ■ 16 bits for OC-48 ■ 32 bits for OC-96	*Byte Serializer* and *Byte Deserializer* sections in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What would you like to base the setting on?	This option is not available for selection in SONET/SDH mode.	—
What is the effective data rate?	The effective data rate is fixed at: ■ 622 Mbps for OC-12 ■ 2488.32 Mbps for OC-48 ■ 4976 Mbps for OC-96	—

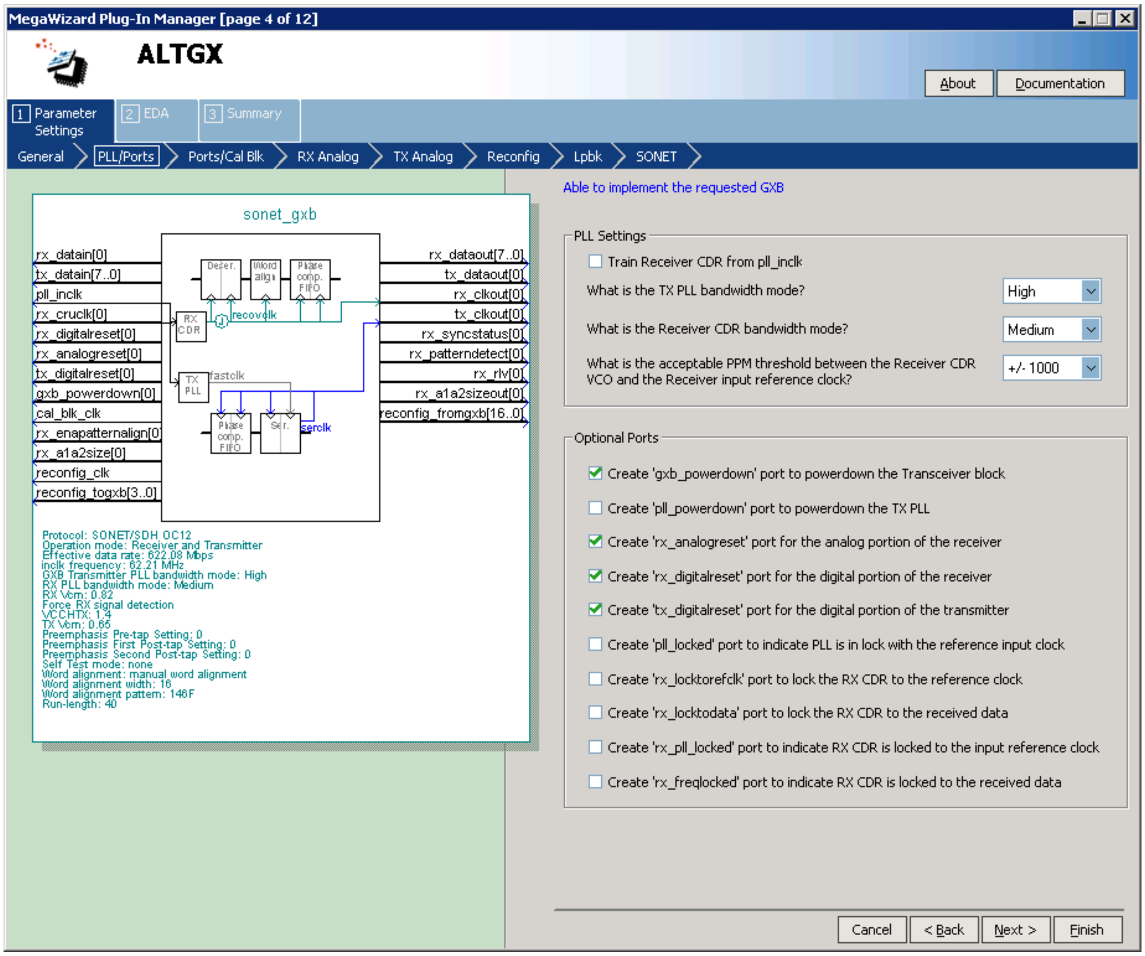
Table 1–47. MegaWizard Plug-In Manager Options (General Screen for SONET/SDH Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
What is the input clock frequency?	Based on the effective data rate value in the **What is the effective data rate?** field, the ALTGX MegaWizard Plug-In Manager determines the input reference clock frequencies depending on the available multiplier settings.	—
Specify base data rate.	This option is not available for selection in SONET/SDH mode as the data rates are fixed in OC-12, OC-48, and OC-96 modes. The ALTGX MegaWizard Plug-In Manager provides you the base data rate options for the VCO in the CMU PLL and receiver CDR in this option.	—

PLL/Ports Screen for SONET/SDH Mode

Figure 1–52 shows the **PLL/Ports** screen of the ALTGX MegaWizard Plug-In Manager for SONET/SDH mode.

Figure 1–52. MegaWizard Plug-In Manager - ALTGX (PLL/Ports Screen)



Protocol: SONET/SDH OC12
Operation mode: Receiver and Transmitter
Effective data rate: 622.08 Mbps
inclk frequency: 62.21 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: manual word alignment
Word alignment width: 16
Word alignment pattern: 146F
Run-length: 40

Able to implement the requested GXB

PLL Settings

□ Train Receiver CDR from pll_inclk

What is the TX PLL bandwidth mode? High

What is the Receiver CDR bandwidth mode? Medium

What is the acceptable PPM threshold between the Receiver CDR VCO and the Receiver input reference clock? +/- 1000

Optional Ports

☑ Create 'gxb_powerdown' port to powerdown the Transceiver block

□ Create 'pll_powerdown' port to powerdown the TX PLL

☑ Create 'rx_analogreset' port for the analog portion of the receiver

☑ Create 'rx_digitalreset' port for the digital portion of the receiver

☑ Create 'tx_digitalreset' port for the digital portion of the transmitter

□ Create 'pll_locked' port to indicate PLL is in lock with the reference input clock

□ Create 'rx_locktorefclk' port to lock the RX CDR to the reference clock

□ Create 'rx_locktodata' port to lock the RX CDR to the received data

□ Create 'rx_pll_locked' port to indicate RX CDR is locked to the input reference clock

□ Create 'rx_freqlocked' port to indicate RX CDR is locked to the received data

Table 1–48 describes the available options on the **PLL/Ports** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–48. MegaWizard Plug-In Manager Options (PLL/Ports Screen for SONET/SDH Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Train receiver CDR from `pll_inclk`.	If you select this option, the input reference clock to the CMU PLL trains the receiver CDR.	*Table 1-2* in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the TX PLL bandwidth mode?	The appropriate bandwidth settings will be fixed after characterization.	*PLL Bandwidth Setting* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the receiver CDR bandwidth mode?	The appropriate bandwidth settings will be fixed after characterization.	—
What is the acceptable PPM threshold between the Receiver CDR VCO and the Receiver input reference clock?	In automatic lock mode, the CDR remains in LTD mode as long as the PPM difference between the CDR VCO output clock and the input reference clock is less than the PPM value that you set in this option. If the PPM difference is greater than the PPM value that you set in this option, the CDR switches to LTR mode. The range of values available in this option is ±62.5 ppm to ±1000 ppm. *(1)*	*Automatic Lock Mode* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `gxb_powerdown` port to power down the Transceiver block.	This is an optional signal. When asserted, it powers down the entire transceiver block. If none of the channels are instantiated in a transceiver block, the Quartus II software automatically powers down the entire transceiver block.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `pll_powerdown` port to power down the TX PLL.	Each transceiver block has two CMU PLLs. Each CMU PLL has a dedicated power down signal called `pll_powerdown`. This signal powers down the CMU PLL.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_analogreset` port for the analog portion of the receiver.	Receiver analog reset port available in **Receiver only** and **Receiver and Transmitter** operation modes. Resets part of the analog portion of the receiver CDR in the receiver channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_digitalreset` port for the digital portion of the receiver.	Receiver digital reset port available in **Receiver only** and **Receiver and Transmitter** operation modes. Resets the PCS portion of the receiver channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–48. MegaWizard Plug-In Manager Options (PLL/Ports Screen for SONET/SDH Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
Create a `tx_digitalreset` port for the digital portion of the transmitter.	Transmitter digital reset port available in **Transmitter only** and **Receiver and Transmitter** operation modes. Resets the PCS portion of the transmitter channel. Altera recommends using this port to implement the recommended reset sequence. The minimum pulse width is two parallel clock cycles.	*User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create a `pll_locked` port to indicate PLL is locked to the input reference clock.	Each CMU PLL has a dedicated `pll_locked` signal that is fed to the FPGA fabric to indicate when the PLL is locked to the input reference clock.	*Transceiver Reset Sequences* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create an `rx_locktorefclk` port to lock the RX CDR to the input reference clock.	This is an optional port. When this signal is asserted high, the LTR/LTD controller forces the receiver CDR to lock to the phase and frequency of the input reference clock. *(1)*, *(2)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create a `rx_locktodata` port to lock the RX CDR to the received data.	This is an optional port. When this signal is asserted high, the LTR/LTD controller forces the receiver CDR to lock to the received data. *(1)*, *(2)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create a `rx_pll_locked` port to indicate RX CDR is locked to the input reference clock.	In LTR mode, this signal is asserted high to indicate that the receiver CDR has locked to the phase and frequency of the input reference clock. In LTD mode, this signal has no significance. *(1)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create a `rx_freqlocked` port to indicate RX CDR is locked to the received data.	This signal gets asserted high to indicate that the receiver CDR has switched from LTR to LTD mode. This signal has relevance only in automatic lock mode and may be required to control the transceiver resets, as discussed in the *User Reset and Power Down Signals* section in the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook*. *(1)*	*Clock and Data Recovery Unit* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

Notes to Table 1–48:

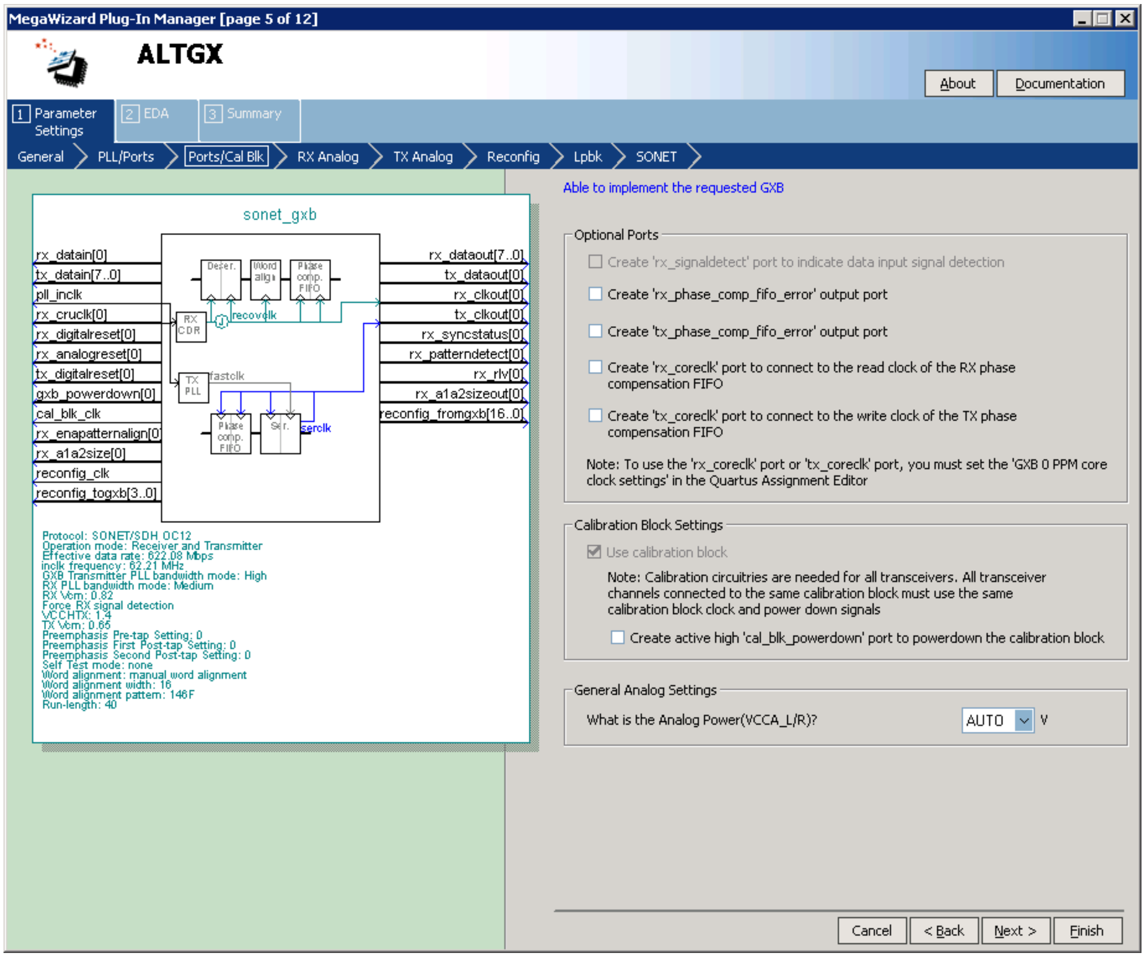
(1) LTR mode is lock-to-reference mode and LTD mode is lock-to-data mode.

(2) When `rx_locktorefclk` and `rx_locktodata` are both asserted high, `rx_locktodata` takes precedence over `rx_locktorefclk`, forcing the CDR to lock to the received data. When both these signals are de-asserted, the LTR/LTD controller is configured in automatic lock mode.

Ports/Cal Blk Screen for SONET/SDH Mode

Figure 1–53 shows the **Ports/Cal Blk** screen of the ALTGX MegaWizard Plug-In Manager for SONET /SDH mode.

Figure 1–53. MegaWizard Plug-In Manager - ALTGX (Ports/Cal Blk Screen)



Protocol: SONET/SDH OC12
Operation mode: Receiver and Transmitter
Effective data rate: 622.08 Mbps
inclk frequency: 62.21 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: manual word alignment
Word alignment width: 16
Word alignment pattern: 146F
Run-length: 40

Able to implement the requested GXB

Optional Ports

- □ Create 'rx_signaldetect' port to indicate data input signal detection
- □ Create 'rx_phase_comp_fifo_error' output port
- □ Create 'tx_phase_comp_fifo_error' output port
- □ Create 'rx_coreclk' port to connect to the read clock of the RX phase compensation FIFO
- □ Create 'tx_coreclk' port to connect to the write clock of the TX phase compensation FIFO

Note: To use the 'rx_coreclk' port or 'tx_coreclk' port, you must set the 'GXB 0 PPM core clock settings' in the Quartus Assignment Editor

Calibration Block Settings

- ☑ Use calibration block

Note: Calibration circuitries are needed for all transceivers. All transceiver channels connected to the same calibration block must use the same calibration block clock and power down signals

- □ Create active high 'cal_blk_powerdown' port to powerdown the calibration block

General Analog Settings

What is the Analog Power(VCCA_L/R)? AUTO V

Table 1–49 describes the available options on the **Ports/Cal Blk** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–49. MegaWizard Plug-In Manager Options (Ports/Cal Blk Screen for SONET/SDH Mode) (Part 1 of 2)

ALTGX Setting	Description	Reference
Create an `rx_signaldetect` port to indicate data input signal detection.	This port is not available in SONET/SDH mode.	*Signal Threshold Detection Circuitry* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.
Create an `rx_phase_comp_fifo_error` output port.	This optional output port indicates receiver phase compensation FIFO overflow or underrun condition.	*Receiver Phase Compensation FIFO Error Flag* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

Table 1–49. MegaWizard Plug-In Manager Options (Ports/Cal Blk Screen for SONET/SDH Mode) (Part 2 of 2)

ALTGX Setting	Description	Reference
Create a `tx_phase_comp_fifo_error` output port.	This optional output port indicates transmitter phase compensation FIFO overflow or underrun condition.	*TX Phase Compensation FIFO Status Signal* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_coreclk` port to connect to the read clock of the RX phase compensation FIFO.	The parallel output data from the receiver can be clocked using this optional input port. It allows you to clock the read side of the receiver phase compensation FIFO with a user-provided clock (FPGA fabric clock, FPGA fabric-transceiver interface clock, or input reference clock).	FPGA fabric-Transceiver Interface Clocking section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_coreclk` port to connect to the write clock of the TX phase compensation FIFO.	The parallel transmitter data generated in the FPGA fabric can be clocked using this optional input port. It allows you to clock the write side of the transmitter phase compensation FIFO with a user-provided clock (FPGA fabric clock, FPGA fabric-transceiver interface clock, or input reference clock).	FPGA fabric-Transceiver Interface Clocking section in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
Use calibration block.	Calibration block is always enabled.	*Calibration Blocks* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an active high `cal_blk_powerdown` to power down the calibration block.	Asserting this signal high powers down the calibration block. A high-to-low transition on this signal restarts calibration.	*Input Signals to the Calibration Block* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the analog power ($V_{CCA\_L/R}$)?	The options available for selection for OC-12 and OC-48 are 2.5 V, 3.0 V and AUTO. The options available for selection, for OC-96 are 3.0 V and AUTO. 3.0 V : upto 8.5 Gbps 2.5 V : upto 4.25 Gbps AUTO : The ALTGX MegaWizard Plug-In Manager will automatically set V_{CCA} to 2.5 V and V_{CCH} (transmitter buffer voltage) to 1.5 V for the VCO base data rates less than 4.25 Gbps. -OR- V_{CCA} to 3.0 V and V_{CCH} (transmitter buffer voltage) to 1.4 V for the VCO base data rates greater than 4.25 Gbps.	*General Requirements to Combine Channels* section in the *Configuring Multiple Protocols and Data Rates* chapter in volume 2 of the *Stratix IV Device Handbook.*

RX Analog Screen for SONET/SDH Mode

Figure 1–6 describes the **RX Analog** screen of the ALTGX MegaWizard Plug-In Manager for SONET/SDH mode. Table 1–4 explains each of the parameter settings available for the **RX Analog** screen.

TX Analog Screen for SONET/SDH Mode

Figure 1–7 describes the **TX Analog** screen of the ALTGX MegaWizard Plug-In Manager for SONET/SDH mode. Table 1–5 explains each of the parameter settings available for the **TX Analog** screen.

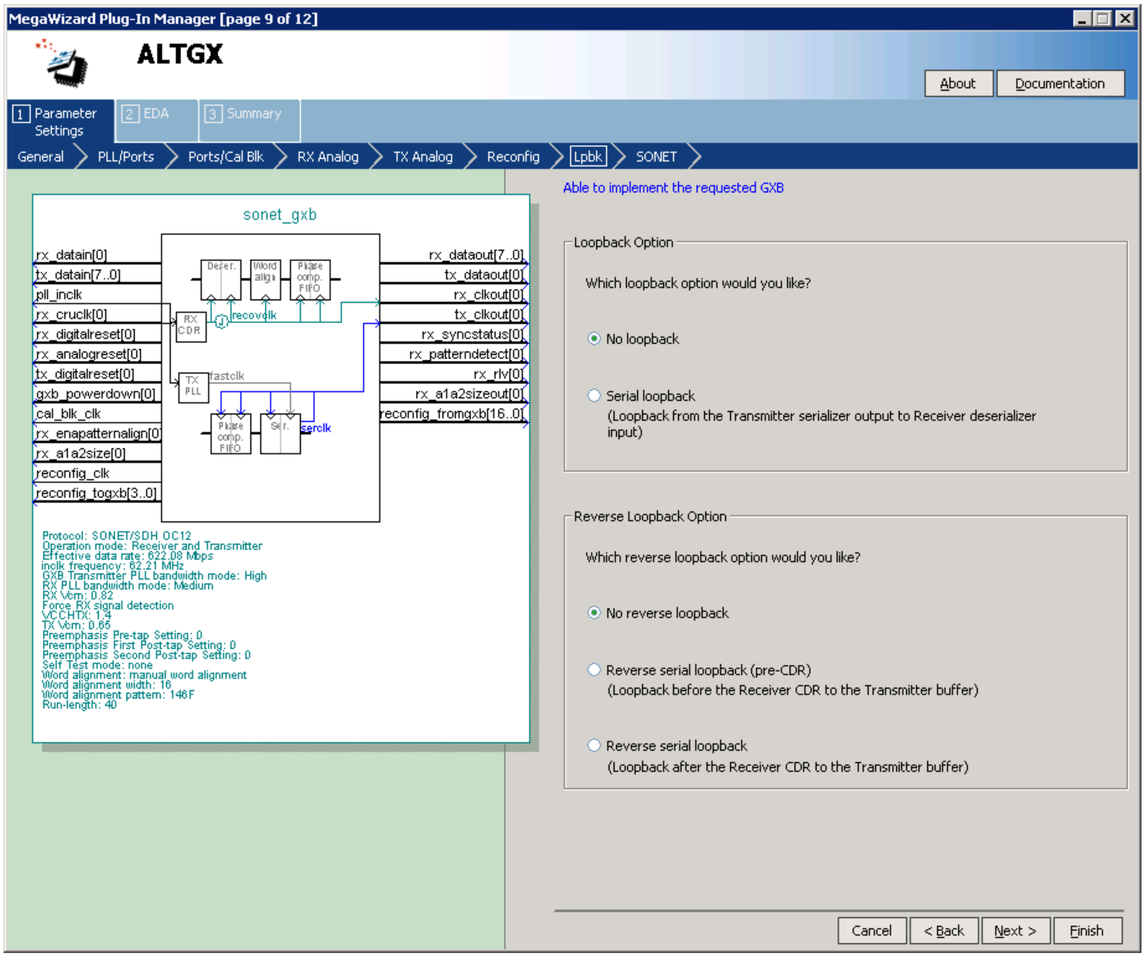
Reconfig Screen for SONET/SDH Mode

Figure 1–8 describes the **Reconfig** screen of the ALTGX MegaWizard Plug-In Manager for SONET/SDH mode. Table 1–6 explains each of the parameter settings available for the **Reconfig** screen.

Lpbk Screen for SONET/SDH Mode

Figure 1–54 shows the **Lpbk** screen of the ALTGX MegaWizard Plug-In Manager for SONET/SDH mode.

Figure 1–54. MegaWizard Plug-In Manager - ALTGX (Lpbk Screen) for SONET/SDH Mode



Protocol: SONET/SDH OC12
Operation mode: Receiver and Transmitter
Effective data rate: 622.08 Mbps
inclk frequency: 62.21 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: manual word alignment
Word alignment width: 16
Word alignment pattern: 146F
Run-length: 40

Able to implement the requested GXB

Loopback Option

Which loopback option would you like?

● No loopback

○ Serial loopback
(Loopback from the Transmitter serializer output to Receiver deserializer input)

Reverse Loopback Option

Which reverse loopback option would you like?

● No reverse loopback

○ Reverse serial loopback (pre-CDR)
(Loopback before the Receiver CDR to the Transmitter buffer)

○ Reverse serial loopback
(Loopback after the Receiver CDR to the Transmitter buffer)

Table 1–50 describes the available options on the **Lpbk** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

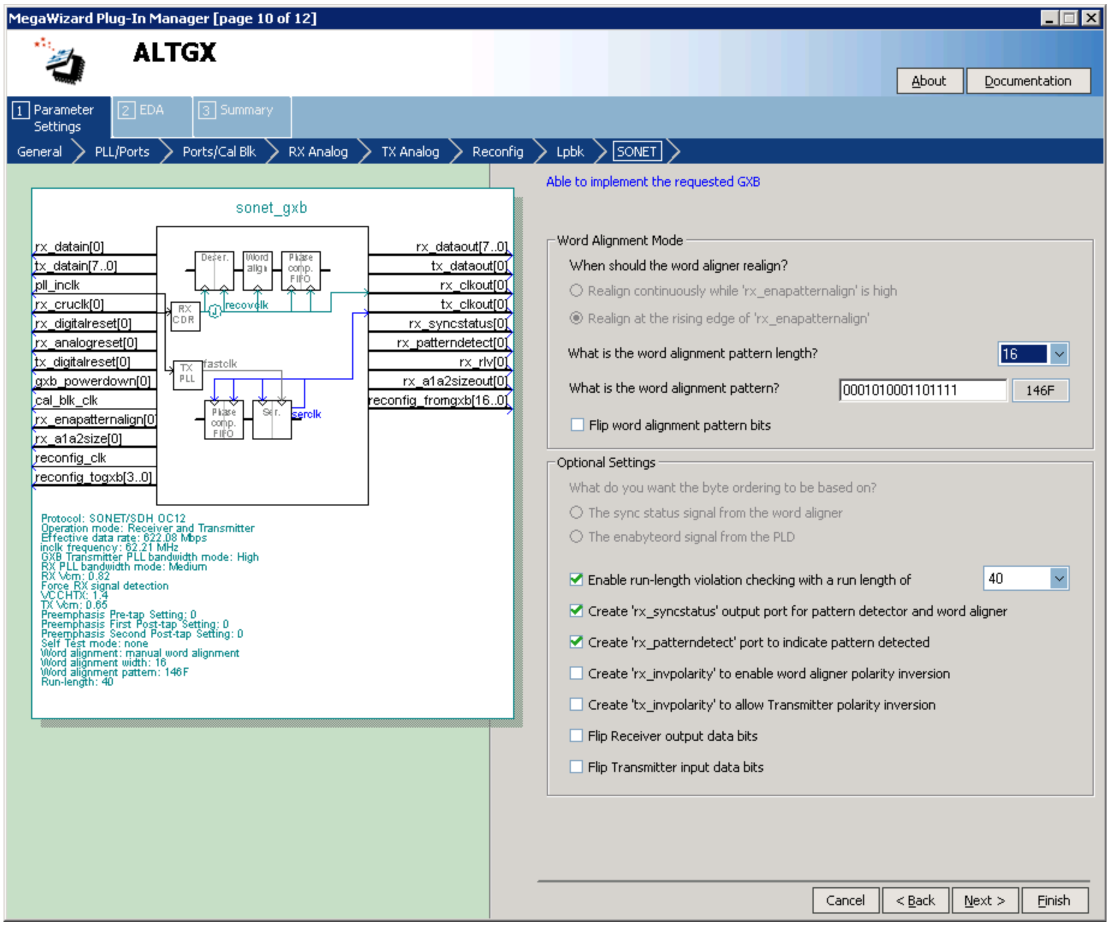
Table 1–50. MegaWizard Plug-In Manager Options (Lpbk Screen for SONET/SDH Mode)

ALTGX Setting	Description	Reference
Which loopback option would you like?	There are two options available in SONET/SDH mode: ■ No loopback — This is the default mode. ■ Serial loopback — If you select serial loopback, the `rx_seriallpbken` port is available to control the serial loopback feature dynamically. 1'b1 — enables serial loopback 1'b0 — disables serial loopback This signal is asynchronous to the receiver data path.	*Serial Loopback* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Which reverse loopback option would you like?	There are three options available in Basic mode: ■ No reverse loopback — This is the default mode. ■ Reverse Serial loopback (pre-CDR) — This is the loopback before the receiver's CDR block to the transmitter buffer. The receiver path in the PCS is active but the transmitter side is not. ■ Reverse Serial loopback — This is a loopback after the receiver's CDR block to the transmitter buffer. The receiver path in the PCS is active but the transmitter side is not.	*Loopback Modes* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

SONET/SDH Screen for SONET/SDH Mode

Figure 1–55 shows the **SONET/SDH** screen of the ALTGX MegaWizard Plug-In Manager for SONET /SDH mode. If the **Enforce default settings for this protocol** option is selected, this page does not appear in the MegaWizard Plug-In Manager.

Figure 1–55. MegaWizard Plug-In Manager - ALTGX (SONET Screen)



Protocol: SONET/SDH OC12
Operation mode: Receiver and Transmitter
Effective data rate: 622.08 Mbps
inclk frequency: 62.21 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: manual word alignment
Word alignment width: 16
Word alignment pattern: 146F
Run-length: 40

Able to implement the requested GXB

Word Alignment Mode

When should the word aligner realign?

○ Realign continuously while 'rx_enapatternalign' is high

● Realign at the rising edge of 'rx_enapatternalign'

What is the word alignment pattern length? 16

What is the word alignment pattern? 0001010001101111 146F

□ Flip word alignment pattern bits

Optional Settings

What do you want the byte ordering to be based on?

○ The sync status signal from the word aligner

○ The enabyteord signal from the PLD

☑ Enable run-length violation checking with a run length of 40

☑ Create 'rx_syncstatus' output port for pattern detector and word aligner

☑ Create 'rx_patterndetect' port to indicate pattern detected

□ Create 'rx_invpolarity' to enable word aligner polarity inversion

□ Create 'tx_invpolarity' to allow Transmitter polarity inversion

□ Flip Receiver output data bits

□ Flip Transmitter input data bits

Table 1–51 describes the available options on the **SONET/SDH** screen of the MegaWizard Plug-In Manager for your ALTGX custom megafunction variation.

Table 1–51. MegaWizard Plug-In Manager Options (SONET Screen for SONET/SDH Mode) (Part 1 of 3)

ALTGX Setting	Description	Reference
When should the word aligner realign?	This option is not available in SONET/SDH mode. In SONET/SDH mode, the word aligner operates in manual alignment mode. By default, the ALTGX MegaWizard Plug-In Manager sets the behavior of the word aligner such that re-alignment occurs when there is a rising edge of the `rx_enapatternalign` input signal in SONET/SDH mode.	*Word Aligner in Single-Width Mode with 8-bit PCA-PMS Interface Modes* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
What is the word alignment pattern length?	This option sets the length of the word alignment pattern. The following options are available: ■ OC-12 – only 16-bit pattern is allowed ■ OC-48 – only 16-bit pattern is allowed ■ OC-96 – 16-bit and 32-bit patterns are allowed	*SONET/SDH Mode (OC-12, OC-48, and OC-96)* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–51. MegaWizard Plug-In Manager Options (SONET Screen for SONET/SDH Mode) (Part 2 of 3)

ALTGX Setting	Description	Reference
What is the word alignment pattern?	Enter the word alignment pattern here. By default, the pattern that appears in the MegaWizard Plug-In Manager is '0001010001101111' (16'h146F).	*SONET/SDH Mode (OC-12, OC-48, and OC-96)* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Flip word alignment pattern bits.	This option is enabled in the MegaWizard Plug-In Manager by default. This option reverses the order of the alignment pattern at a bit level to support MSB-to-LSB transmission in SONET/SDH mode. The ALTGX MegaWizard Plug-In Manager flips the bit order of the default word alignment pattern '0001010001101111 '(16'h146F) and uses the flipped version '1111011000101000' (16'hF628) as the word alignment pattern.	—
What do you want the byte ordering to be based on?	This option allows you to trigger the byte ordering block either on the rising edge of the `rx_syncstatus` signal or the user-controlled `rx_enabyteord` signal from the FPGA fabric. The byte ordering block is enabled only in OC-48 mode.	*Byte Ordering Block* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Enable run-length violation checking with a run length of.	This option creates the output signal `rx_rlv`. Enabling this option also activates the run-length violation circuit. If the number of continuous 1's and 0's exceeds the number that you set in this option, the run-length violation circuit asserts the `rx_rlv` signal. The `rx_rlv` signal is asynchronous to the receiver data path and is asserted for a minimum of two recovered clock cycles in OC-12 and OC-48 modes. Similarly, it is asserted for a minimum of three recovered clock cycles in the OC-96 mode. For the OC-12 and OC-48 modes, the run length limits are 4 to 128 in increments of 4. For the OC-96 mode, the run length limits are 5 to 160 in increments of 5.	*Programmable Run Length Violation Detection* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_syncstatus` output port for pattern detector and word aligner.	This is an output status signal that the word aligner forwards to the FPGA fabric to indicate that synchronization has been achieved. This signal is synchronous with the parallel receiver data on the `rx_dataout` port. The signal width is 1-bit, 2-bits, and 4-bits for a channel width of 8-bits, 16-bits, and 32-bits, respectively.	*Table 1-24 and Word Aligner in Single-Width Mode with 8-bit PCA-PMS Interface Modes* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create an `rx_patterndetect` port to indicate pattern detected.	This is an output status signal that the word aligner forwards to the FPGA fabric to indicate that the word alignment pattern programmed has been detected in the current word boundary. The signal width is 1-bit, 2-bits, and 4-bits for a channel width of 8-bits, 16-bits, and 32-bits, respectively.	*Table 1-24 and Word Aligner in Single-Width Mode with 8-bit PCA-PMS Interface Modes* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

Table 1–51. MegaWizard Plug-In Manager Options (SONET Screen for SONET/SDH Mode) (Part 3 of 3)

ALTGX Setting	Description	Reference
Create a `rx_invpolarity` port to enable word aligner polarity inversion.	This optional port allows you to dynamically reverse the polarity of every bit of the received data at the input of the word aligner. Use this option when the positive and negative signals of the differential input to the receiver (`rx_datain`) are erroneously swapped on the board.	*Receiver Polarity Inversion* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Create a `tx_invpolarity` port to allow Transmitter polarity inversion.	This optional port allows you to dynamically reverse the polarity of every bit of the data word fed to the serializer in the transmitter data path. Use this option when the positive and negative signals of the differential output from the transmitter (`tx_dataout`) are erroneously swapped on the board.	*Transmitter Polarity Inversion* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Flip receiver output data bits.	This option reverses the bit order of the parallel receiver data at a byte level at the output of the receiver phase compensation FIFO to support MSB-to-LSB transmission in SONET/SDH mode. For example, if the 16-bit parallel receiver data at the output of the receiver phase compensation FIFO is '10111100 10101101' (16'hBCAD), enabling this option reverses the data on the `rx_dataout` port to '00111101 10110101' (16'h3DB5).	*SONET/SDH Mode (OC-12, OC-48, and OC-96)* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
Flip transmitter input data bits.	This option reverses the bit order of the parallel transmitter data at a byte level at the input of the transmitter phase compensation FIFO to support MSB-to-LSB transmission protocols in SONET/SDH mode. For example, if the 16-bit parallel transmitter data at the `tx_datain` port is '10111100 10101101' (16'hBCAD), enabling this option reverses the input data to the transmitter phase compensation FIFO to '00111101 10110101' (16'h3DB5).	*SONET/SDH Mode (OC-12, OC-48, and OC-96)* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

EDA Screen for SONET/SDH Mode

Figure 1–13 describes the **EDA** screen of the MegaWizard Plug-In Manager for SONET/SDH mode. The **Generate Netlist** option generates a netlist for the third party EDA synthesis tool to estimate timing and resource utilization for the ALTGX instance.

Summary for SONET/SDH Mode

Figure 1–14 describes the **Summary** screen of the MegaWizard Plug-In Manager for SONET/SDH mode. You can select optional files on this page. After you make your selections, click **Finish** to generate the files.

Referenced Documents

This chapter references the following documents:

- *DC and Switching Characteristics of the Stratix IV Device Family* chapter in volume 5 of the *Stratix IV Device Handbook*
- *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*
- *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook.*
- *Configuring Multiple Protocols and Data Rates* chapter in volume 2 of the *Stratix IV Device Handbook.*
- *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook.*

Document Revision History

Table 1–52 shows the revision history for this document.

Table 1–52. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008, v2.0	■ Updated Table 1–1 ■ Updated Figure 1–8 ■ Updated Table 1–6 ■ Updated Table 1–9	—
June 2008, v1.1	Minor text edit.	—
May 2008, v1.0	Initial Release.	—

2. Transceiver Design Flow Guide

SIV53002-2.0

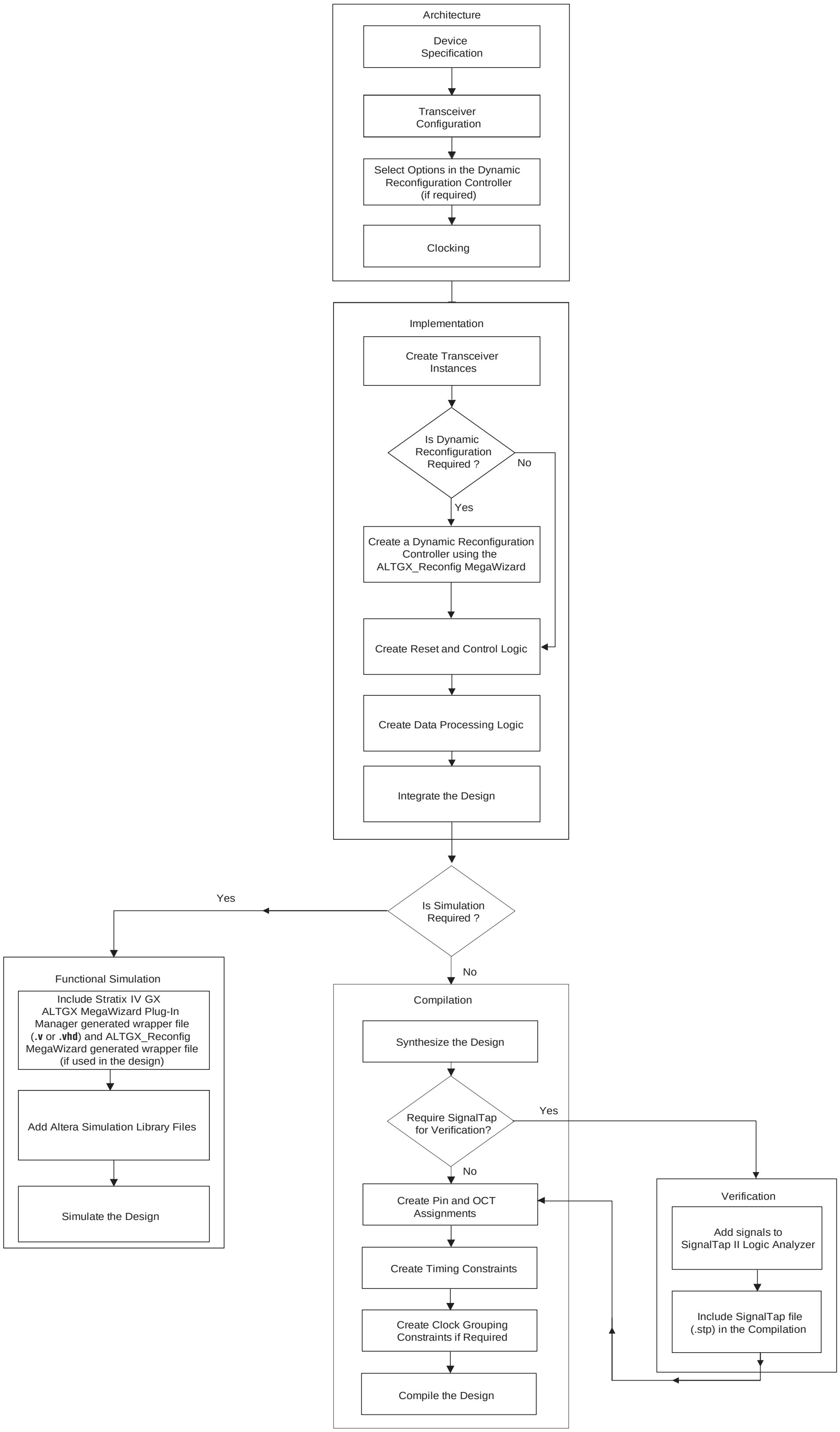
Overview

In this chapter, Altera® recommends using a generic design flow to simplify Stratix® IV GX transceiver-based designs. This chapter discusses the design flow from a transceiver perspective. The transceiver-based design is divided into phases and detailed in the following sections of this chapter:

- "Architecture" on page 2–3
- "Implementation and Integration" on page 2–6
- "Compilation" on page 2–8
- "Verification" on page 2–10
- "Simulation" on page 2–10

Figure 2–1 shows the flow chart of the different stages. Each of the stages is explained in the sections that follow.

Figure 2–1. Flow Chart of the Different Stages in a Transceiver-Based Design


Architecture
Device Specification
Transceiver Configuration
Select Options in the Dynamic Reconfiguration Controller (if required)
Clocking
Implementation
Create Transceiver Instances
Is Dynamic Reconfiguration Required ?
No
Yes
Create a Dynamic Reconfiguration Controller using the ALTGX_Reconfig MegaWizard
Create Reset and Control Logic
Create Data Processing Logic
Integrate the Design
Is Simulation Required ?
Yes
No
Functional Simulation
Include Stratix IV GX ALTGX MegaWizard Plug-In Manager generated wrapper file (.v or .vhd) and ALTGX_Reconfig MegaWizard generated wrapper file (if used in the design)
Add Altera Simulation Library Files
Simulate the Design
Compilation
Synthesize the Design
Require SignalTap for Verification?
Yes
No
Create Pin and OCT Assignments
Create Timing Constraints
Create Clock Grouping Constraints if Required
Compile the Design
Verification
Add signals to SignalTap II Logic Analyzer
Include SignalTap file (.stp) in the Compilation

Architecture

The first step in creating a transceiver-based design is to map the system requirements with the Stratix IV GX device supported features. The Stratix IV GX device contains multiple transceiver channels that can be configured in multiple data rates and protocols. It also provides multiple transceiver clocking options. For your design, identify the transceiver capabilities and various clocking options to ensure that the transceiver meets your system requirements.

This section describes the critical parameters that need to be identified as part of this architecture phase.

Device Specification

Ensure that the following device specifications meet your requirements:

- Check the device data sheet to ensure that the transceivers meet the data rate and electrical requirements for your target high-speed interface application, such as jitter specification, V_{OD} range, and so on.
- Check whether the device family that you select supports the design requirements, such as number of transceiver channels, FPGA logic density, memory elements, and DSP blocks.
- If you intend to migrate to a higher logic density or higher transceiver count device in the future, ensure that the migration device is available.

For information about device characteristics, refer to the *Transceiver Performance Specifications* section in the *DC and Switching Characteristics of Stratix IV Devices* chapter in volume 4 of the *Stratix IV Device Handbook*.

Transceiver Configuration

Ensure that the following transceiver configuration meets your requirements:

- Check whether the transceiver PCS and PMA functional blocks are compliant with your system requirements. For example, check whether the rate match (clock rate compensation) FIFO in the receiver channel PCS meets the PPM specifications required for your application.

For more information about transceiver specifications, refer to the *Transceiver Performance Specifications* section of the *DC and Switching Characteristics of Stratix IV Devices* chapter in volume 4 of the *Stratix IV Device Handbook*.

- The Quartus® II software provides an ALTGX MegaWizard® Plug-In Manager user interface to select the transceiver PCS and PMA functional blocks. Check whether the ALTGX MegaWizard Plug-In Manager settings allow you to select required functional blocks for your transceiver configuration.

  The options that you can enable depend on transceiver configuration parameters, such as data rate and single or double width mode.

For more information about the ALTGX MegaWizard Plug-In Manager, refer to the *ALTGX Megafunction User Guide* chapter in volume 3 of the *Stratix IV Device Handbook*.

- Identify the functional blocks that you need to implement in the FPGA fabric.

  In some cases, you might need to implement some of the functional blocks in the FPGA fabric to meet your application requirements. Altera has provided a few examples of functional blocks that are typically implemented in the FPGA fabric to interface with the transceiver in "Create Data Processing and Other User Logic" on page 2–8.

- Check whether the PMA settings match your requirements.

  Stratix IV GX transceiver channels provide pre-emphasis and equalization options to compensate for interconnect losses.

For more information, refer to the *Transmitter Output Buffer* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook.*

- Check whether the loop-back features are available for the selected functional mode.

  The Stratix IV GX transceiver provides diagnostic loop-back features between the transmitter channel and the receiver channel. These loop-back features help in debugging your design. If you intend to use serial loop-back to verify design functionality, you might require user logic that mimics the upstream system and performs as a handshake mechanism with the receiver side of the transceiver channel. This helps to check design functionality using serial loop-back at a protocol level.

- If your design uses multiple transceiver channels within the same transceiver block, based on the transceiver channel configuration, the Quartus II software might impose some restrictions on combining these channels.

For more information about these restrictions, refer to the *Configuring Multiple Protocols and Data Rates* chapter in volume 2 of the *Stratix IV Device Handbook*.

Dynamic Reconfiguration

Stratix IV GX transceivers enable you to use the transceivers in multiple-link interconnect environments by allowing you to dynamically reconfigure the PMA controls (V_{OD}, Preemphasis, Equalization, and DC gain). The PMA controls can be reconfigured without affecting any other transceiver channel or the logic in the FPGA fabric. The Quartus II software provides a dynamic reconfiguration controller, a soft logic that controls the transceiver reconfiguration with minimal user interface logic. This soft logic can be generated using the ALTGX_Reconfig MegaWizard interface.

For more information about this interface, refer to the *ALTGX_Reconfig MegaWizard* setup guide chapter in volume 3 of *Stratix IV Handbook*.

Note that all the receiver channels in the Stratix IV GX device require offset cancellation to counter offset variations in process, voltage, and temperature on the receiver. The dynamic reconfiguration controller initiates the sequence to perform offset cancellation on the receiver channels.

Therefore, if you configure the Stratix IV GX transceiver channel in the **Receiver only** or **Transmitter and Receiver** configuration, you must instantiate a dynamic reconfiguration controller.

For more information about offset cancellation, refer to the offset cancellation section in the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*

Clocking

The Stratix IV GX transceiver can be clocked by various input reference clocks, such as:

- Dedicated transceiver reference clock (`refclk`) pins. Altera recommends using the `refclk` resource whenever possible since `refclk` yields reduced jitter on transmitted data.
- Clock sources connected to global clock lines.
- Clock outputs from the PLLs in the FPGA fabric.

Identify the input reference clock sources for the transceiver channels, for example:

- Ensure that the selected device has the required number of input reference clock resources to implement your design.
- Ensure that the transceiver clock input supports the required I/O standards.
- Clocking restrictions:
  - Check whether the allowed frequencies for the transceiver input reference clocks meet the requirements.
  - If you use the PLL cascade clock, understand the restrictions on the PLL cascade clock usage.

For more information about the transceiver input reference clocks, I/O standards, and PLL cascade clock use, refer to the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*.

For transceiver-FPGA interface clocking:

- Ensure that the transceiver-FPGA interface clock frequencies meet your system requirements.

For information about transceiver specifications, refer to the *Transceiver Block Characteristics* section in the *DC and Switching Characteristics of Stratix IV Devices* chapter in volume 4 of the *Stratix IV Device Handbook*.

- Identify the clocking scheme to clock the transceiver data to the logic in the FPGA fabric. For example, if your design has multiple transceiver channels that run at the same data rate and are connected to the one upstream link, you might be able to use a single transceiver-FPGA clock to provide clocks to the transceiver data path, which can conserve clock routing resources.

For information about transceiver clocking, refer to the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*.

After you identify the required transceiver parameters, start the implementation and integration phase.

Implementation and Integration

The implementation and integration phase can be subdivided into three steps:

- Create transceiver instances
- Create reset logic to control the FPGA fabric and transceivers
- Create data processing and other user logic

Create Transceiver Instances

The Quartus II software provides the ALTGX MegaWizard Plug-In Manager to create the transceiver instance. In the architecture phase, you identified the transceiver configuration for the design. Using the ALTGX MegaWizard Plug-In Manager, select the appropriate parameters that apply to your architecture requirements.

For more information about using the ALTGX MegaWizard Plug-In Manager and the functionality of the different options and signals available, refer to the *ALTGX Megafunction User Guide* chapter in volume 3 of the *Stratix IV Device Handbook*.

The ALTGX MegaWizard Plug-In Manager provides multiple parameters and options. Some of the parameters required for the design are:

- Transceiver data rate
- Number of channels you require with the same configuration

  When you set the number of channels to **2** and complete the MegaWizard Plug-In Manager instantiation, the two channels will have identical configurations.
- Channel interface width – 8, 10, 16, 20, 32, and 40 (the available options depend on whether you use single-width or double-width mode in the **General** screen)
- Input reference clock frequencies

  The ALTGX MegaWizard Plug-In Manager automatically computes the allowed input clock frequency values based on the data rate.
- Select the reset signals
  - `tx_digitalreset`, `rx_analogreset`, and `rx_digitalreset`

    These signals are required to reset the PCS and PMA functional blocks of the transceiver channel
  - `pll_powerdown`

    This signal resets the CMU PLL that provides clocks to the transmitter channel.

- Required functional blocks

  Navigate through the different screens to select the required functional blocks, such as 8B/10B encoder and decoder, word alignment settings, and byte ordering.

  - Determine the output differential voltage (V_{OD}) and common mode voltage (V_{CM}) and set any pre-emphasis values based on the target protocol application requirements.
  - Select the status and control signals

    Examples of status and control signals include `rx_enapatternalign`, `pll_locked`, `rx_freqlocked`, `rx_syncstatus`, `rx_patterndetect`, and so on. The `pll_locked` and `rx_freqlocked` signals are useful for controlling transceiver resets.

If you determine that your application requires dynamic reconfiguration, select the options in the **Reconfig** screen of the ALTGX MegaWizard interface.

- Each transceiver channel has both transmit and receive parallel clocks to interface with the FPGA-fabric. Depending on your system, when multiple transceiver channels are used, you might be able to share the transmitter and receiver parallel clocks of one channel with other channels. If your design requires sharing a clock resource, select the `tx_coreclk` and `rx_coreclk` ports.

The conditions as to when to share the transceiver-FPGA fabric interface clocks are provided in the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*.

After you have selected all the parameters, click **Finish** to create the ALTGX instance.

Create Dynamic Reconfiguration Controller Instances

Use the ALTGX_Reconfig MegaWizard interface to create the dynamic reconfiguration controller instance.

Refer to *Stratix IV Dynamic Reconfiguration* in volume 2 of the *Stratix IV Handbook* for more information about using the signals.

Create Reset Control Logic

The reset sequence is important for initializing the transceiver functional blocks to proper operating condition. Altera recommends a reset sequence for different transceiver configuration and protocol functional modes. The ALTGX MegaWizard Plug-In Manager provides the `tx_digitalreset`, `rx_analogreset`, `rx_digitalreset`, and `pll_powerdown` signals to reset the different functional blocks of the transceiver. The CMU PLL can be reset using the `pll_powerdown` signal. For transceiver instances that share the same CMU PLL, the `pll_powerdown` port of these instances must be driven by the same logic.

For more information about reset sequences, refer to the *Reset Control and Powerdown* chapter in volume 2 of the *Stratix IV Device Handbook*.

Create Data Processing and Other User Logic

A typical transceiver-based design consists of custom data processing and other user logic in the design. Custom logic is application specific and therefore, not discussed in this chapter. This section is limited to some examples of user logic that are typically required in the FPGA fabric to interface with the transceiver for specific transceiver configurations.

PPM Detector When Receiver CDR Is Used in Manual Lock Mode

Each receiver channel contains a CDR that can be used in automatic lock mode or manual lock mode. If you use the CDR in manual lock mode, you can control the timing of the CDR to lock to the input reference clock or lock to the recovered data, using the `rx_locktorefclk` and `rx_locktodata` ports. In asynchronous serial interface systems, there can be parts per million (PPM) differences between the two communicating systems. User logic in the FPGA fabric might require knowledge of the PPM differences so that it can trigger a sequence of corrective actions when the PPM difference goes beyond the desired limit. When you use the receiver CDR in manual lock mode, you might need to implement the PPM detector in the FPGA fabric.

Sync State Machine in Manual Word Alignment Mode

Each receiver channel contains a synchronization state machine in the PCS that can be enabled in some functional modes. The synchronization state machine triggers the loss of synchronization status to the FPGA fabric, based on invalid 8b/10b code groups.

In some functional modes in which the synchronization state machine in the PCS is not available, you might need to implement your custom logic in the FPGA fabric to indicate the loss of synchronization of received data.

After you implement all of the required logic, integrate the transceiver instances with the remaining logic and provide the appropriate transceiver-FPGA fabric interface clocking. Synthesize the design using third-party synthesis tools, such as Synplicity, or use the Quartus II software synthesis tool. This allows you to detect all the syntax errors in your design.

Compilation

When you compile the design, the Quartus II software generates an SRAM Object File (**.sof)** file or programmer object file (**.pof**) that can be downloaded to the Stratix IV GX hardware. Typically, the first step in compiling the design is assigning pin locations for the I/Os and clocks. You can use the **Assignment Editor** in the Quartus II software to assign pins.

☞ For a basic tutorial on the Quartus II software, open the Quartus II software, click the **Help** menu, and click the **Tutorial** option.

- Stratix IV GX transceivers support a variety of I/O standards for the input reference clocks and serial data pins. Assign pins and the logic level standard (for example, 1.5 V PCML, LVDS) for the input and output pins.

- If you share the same transceiver-FPGA fabric interface clocks for multiple transceiver channels (`tx_coreclk` and `rx_coreclk`) in your design, set the **0 ppm** constraints. These constraints enable the Quartus II software to relax the legality check restrictions on clocking.

For more information, refer to the "Common Clock Driver Selection Rules" section of the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*.

- For transceiver serial pins and `refclk` pins, set the on-chip termination (OCT) resistor settings.

For more information about supported OCT settings, refer to "Transmitter Output Buffer" section of *Stratix IV Transceiver Architecture* in volume 2 of the *Stratix IV Device Handbook*.

- Create timing constraints for the clocks and data paths. You can use the TimeQuest Timing Analyzer to set timing constraints.

For more information on using the TimeQuest Timing Analyzer, refer to the *Quartus II Development Software Handbook*.

- Compile the design. This generates an **.sof** file that can be downloaded in the FPGA.

The Quartus II software generates multiple report files that contain information such as transceiver configuration and clock resource utilization. The following section discusses the report files relevant to transceivers and clock resource utilization.

Report Files

The Quartus II software provides a report file in the synthesis, fitter, map, placement, and assembler stages. The report file provides useful information on the device and transceiver configuration generated by the Quartus II software. In this section, only the reports provided in the fitter stage are discussed. To access the report, click on the **Processing** menu, select the **Compilation Report** option, and expand the **Fitter** tab.

Fitter Summary

The fitter summary provides high-level information on FPGA fabric resources and transceiver channels utilized by the design.

Pin-Out File

Select the **Pin-Out file** option under the **Fitter** tab.

The Quartus II software displays the I/O standards and bank numbers of all the pins (used and unused) needed to connect to the board. The Quartus II software also generates a **.pin** file with the above information. Altera recommends that you use the **.pin** file as a guideline. Use the pin connection guidelines for board layout.

For more information about pin connection guidelines for board layout see the *Stratix IV GX Device Family Pin Connection Guidelines*.

Resource Section

Expand the **Resource Section** option under the **Fitter** tab.

The **GXB Transmitter channel** tab provides generated settings for all the transmitter channels instantiated in the design.

The **GXB Transmitter PLL** tab provides generated settings for all the transmitter PLLs instantiated in the design.

The **GXB Receiver channel** tab provides generated settings for all the receiver channels instantiated in the design.

The **Global and other fast signals** tab displays the list of clock and other signals in the design that are assigned to the global and regional clock resources.

You can use the report file to verify whether the transceiver settings, such as data rate, are generated per your settings in the ALTGX MegaWizard Plug-In Manager.

Verification

The Quartus II software provides the SignalTap® Logic Analyzer that allows you to verify the design functionality using the on-chip logic analyzer. SignalTap provides options to create multiple sets of signals that can be sampled using different trigger clocks. You can add the signals to the SignalTap Logic Analyzer and save the file. The Quartus II software saves it with an **.stp** extension. When you include this **.stp** file along with the design files and compile the design, the Quartus II software creates an **.sof** file that allows you to verify the functionality of the signals that you added in the SignalTap Logic Analyzer file.

From the Quartus II software, you can run the **.stp** file that connects to the device through the JTAG port and displays the signal transitions.

For more information about using SignalTap, refer to the *In-System Design Debugging* section in volume 3 of the *Quartus II Development Software Handbook*.

The Stratix IV GX transceiver provides diagnostic loop-back features between the transmitter and receiver channels. These features provide the ability to verify the functionality of PCS and PMA blocks.

For more information, refer to the *Loop-Back* section in the *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

Simulation

This section discusses the functional simulation of the Stratix IV GX transceiver.

Functional Simulation

You can use the ALTGX MegaWizard-generated wrapper file to simulate the instantiated transceiver configuration in third-party simulation software such as ModelSim. For simulation, specific Altera simulation library files are required (listed in Table 2–1). The following library files are available in VHDL and verilog versions:

- 220pack

- 220model
- altera_mf_components
- altera_mf
- sgate_pack
- sgate
- stratixiv_hssi_component
- stratixiv_hssi_atoms

These simulation files are available under the following folder in the Quartus II installation directory: <*Quartus II installation folder*>`/eda/sim_lib`

For VHDL simulation using ModelSim, create the following libraries in your ModelSim project:

- lpm
- sgate
- altera_mf
- stratixiv_hssi

These simulation files are available under the <*Quartus II installation folder\quartus\eda\sim_lib*>.

Compile the simulation files into the libraries specified in Table 2–1.

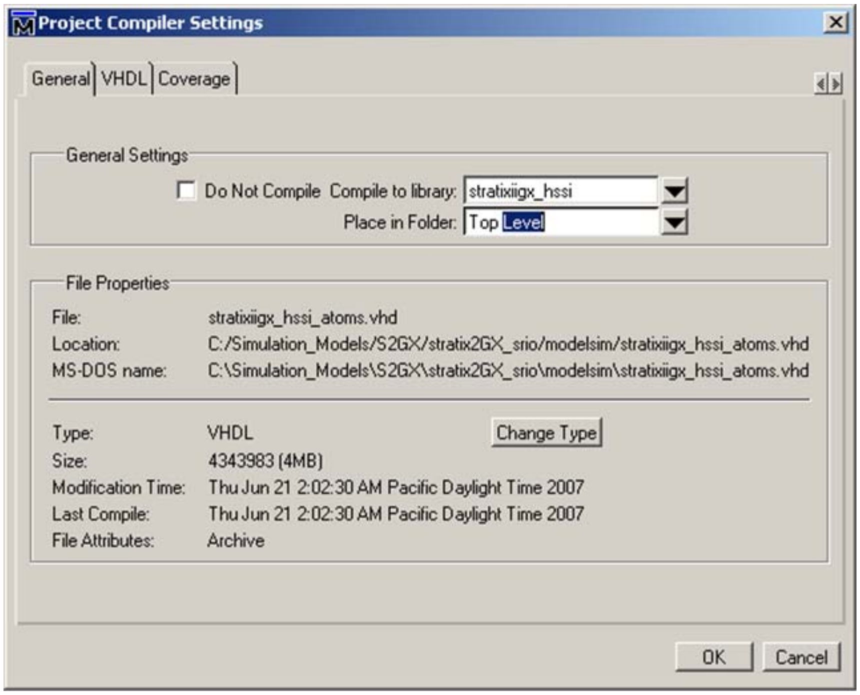
Table 2–1. Library to Compile Simulation Files

Altera Simulation Files	Library
220pack	lpm
220model	lpm
sgate pack	sgate
sgate	sgate
altera_mf_components	altera_mf
altera_mf	altera_mf
stratixiv_hssi_component	stratixiv_hssi
stratixiv_hssi_atoms	stratixiv_hssi
user design files	work

For example, to compile a file into a specific library using ModelSim, right click on the file and select **Properties**, then click the **General** tab.

In the **Compile to library** option, select the corresponding library for the file selected. Figure 2–2 shows the ModelSim window compilation of files in a specific library for the Stratix II GX device.

Figure 2–2. ModelSim Option to Compile Files in a Specific Library



General Settings

□ Do Not Compile Compile to library: stratixiigx_hssi

Place in Folder: Top Level

File Properties

File: stratixiigx_hssi_atoms.vhd
Location: C:/Simulation_Models/S2GX/stratix2GX_srio/modelsim/stratixiigx_hssi_atoms.vhd
MS-DOS name: C:\Simulation_Models\S2GX\stratix2GX_srio\modelsim\stratixiigx_hssi_atoms.vhd

Type: VHDL
Size: 4343983 (4MB)
Modification Time: Thu Jun 21 2:02:30 AM Pacific Daylight Time 2007
Last Compile: Thu Jun 21 2:02:30 AM Pacific Daylight Time 2007
File Attributes: Archive

Include all the libraries in the search path. Add the ALTGX and ALTGX_RECONFIG MegaWizard Plug-In Manager-generated wrapper files (**.v** or **.vhd**) and all of the design files to the library. Compile all the library files first, then the design files, and then run the simulation.

For Verilog simulation, add the ALTGX and ALTGX_Reconfig MegaWizard Plug-In Manager-generated Verilog wrapper files (**.v**), the Altera library files, and all of the design files. Compile all the library files first, then the simulation model file, followed by the design files. Then, run the simulation.

The above mentioned guidelines are illustrated in the following section "Example 1 – Fibre Channel Protocol Application".

Example 1 – Fibre Channel Protocol Application

Assume that you want to implement a fibre channel protocol application using three transceiver channels. Consider the following system requirements:

- Required number of transceiver channels: 3
- All the channels should be placed in the same transceiver block
- All the channels should have independent control to reset their PCS and PMA functional blocks

Table 2–2 shows the transceiver channel configuration considered in Example 1.

Table 2–2. Transceiver Channel Configuration for Example 1

Channels	Mode of Operation	Data Rate	Input Reference Clock Frequency
0	Receiver and Transmitter	FC4G (4.25 Gbps)	106.25 MHz
1	Receiver and Transmitter	FC1G (1.0625 Gbps)	53.125 MHz
2	Transmitter Only	FC4G (4.25 Gbps)	106.25 MHz

Architecture

In this phase, check whether the Stratix IV GX device supports or meets the design requirements.

Device Specification

Consider the following questions before setting device-specific parameters:

- Meet the fibre channel protocol electrical requirements?

  Yes

Refer to the *Transceiver Performance Characteristics* section in the *DC and Switching Characteristics of Stratix IV Devices* chapter in volume 4 of the *Stratix IV Device Handbook*.

- Three transceiver channels are available?

  Yes

- Support for 4.25-Gbps and 1.0625-Gbps data rates?

  Yes. Two CMU PLLs are available within each transceiver block to support two different transmitter data rates. Each receiver channel contains a dedicated receiver CDR that supports 4.25-Gbps and 1.0625-Gbps data rates.

For the maximum data rates supported, refer to the "Transceiver Performance Characteristics" section in the *DC and Switching Characteristics of Stratix IV Devices* chapter in volume 4 of the *Stratix IV Device Handbook*.

Transceiver Configuration

Functional Blocks

The fibre-channel protocol uses an 8B/10B encoder and requires clock rate compensation. Consider the following questions before configuring the transceiver:

- Is the 8b/10b encoder in the PCS block fibre-channel compliant?

  No. The fibre-channel protocol consists of two different End-of-Frame (EOFt) ordered sets. The correct EOFt ordered set sent by user logic depends on the ending disparity of the word preceeding the EOFt. The Stratix IV GX transceiver does not provide running disparity flags to the user logic. Therefore, the user logic might not be able to select the correct EOFt ordered set.

- Is there any workaround?

  Yes. Implement the 8b/10b encoder in the FPGA fabric.

- Is the clock rate compensation block in the PCS available without an 8b/10b encoder?

  No. This can be implemented in the FPGA fabric.

The design requires a "Transmitter and Receiver" configuration for two channels and a "Transmitter Only" configuration for one channel

- Does the Stratix IV GX transceiver support these two configurations, and allow you to combine them within the same transceiver block?

  Yes.

  The available FPGA fabric interface width is 20 or 40 bits to support 4.25-Gbps and 1.0625 -Gbps data rates, respectively. This FPGA fabric interface facilitates 8b/10b encoding and decoding in the FPGA fabric without any additional rearrangement of the received parallel data to a 10-bit boundary.

Dynamic Reconfiguration

If your application requires you to dynamically reconfigure the transceiver PMA controls, ensure that you understand the settings, options, and user logic required to enable this feature.

For more information about understanding interfacing between the ALTGX and the ALTGX_reconfig instances, refer to the "Interfacing ALTGX_reconfig Instance and ALTGX Instances" section in the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*.

For more information about initiating read and write transactions, refer to the "Dynamically Reconfiguring PMA Controls" section in the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook.*

Clocking

Consider the following questions with respect to clocking:

- Support for two different input reference clocks?

  Yes. The Stratix IV GX transceiver has two `refclk` pins for each transceiver block.

- Do the `refclk` pins support the required frequency range?

  Yes. The minimum frequency range of `refclk` is 50 MHz; the maximum frequency range is 622.08 MHz.

- Can transceiver-FPGA fabric interface clocking be shared?

  The design requires independent control on all channels, so you must not share the transceiver-FPGA fabric interface clock of one channel with another channel. Each of the channels must use its own `tx_clkout` and `rx_clkout` signals to clock the data between the transceiver channels and the FPGA fabric.

- Does the Stratix IV GX transceiver support this feature?

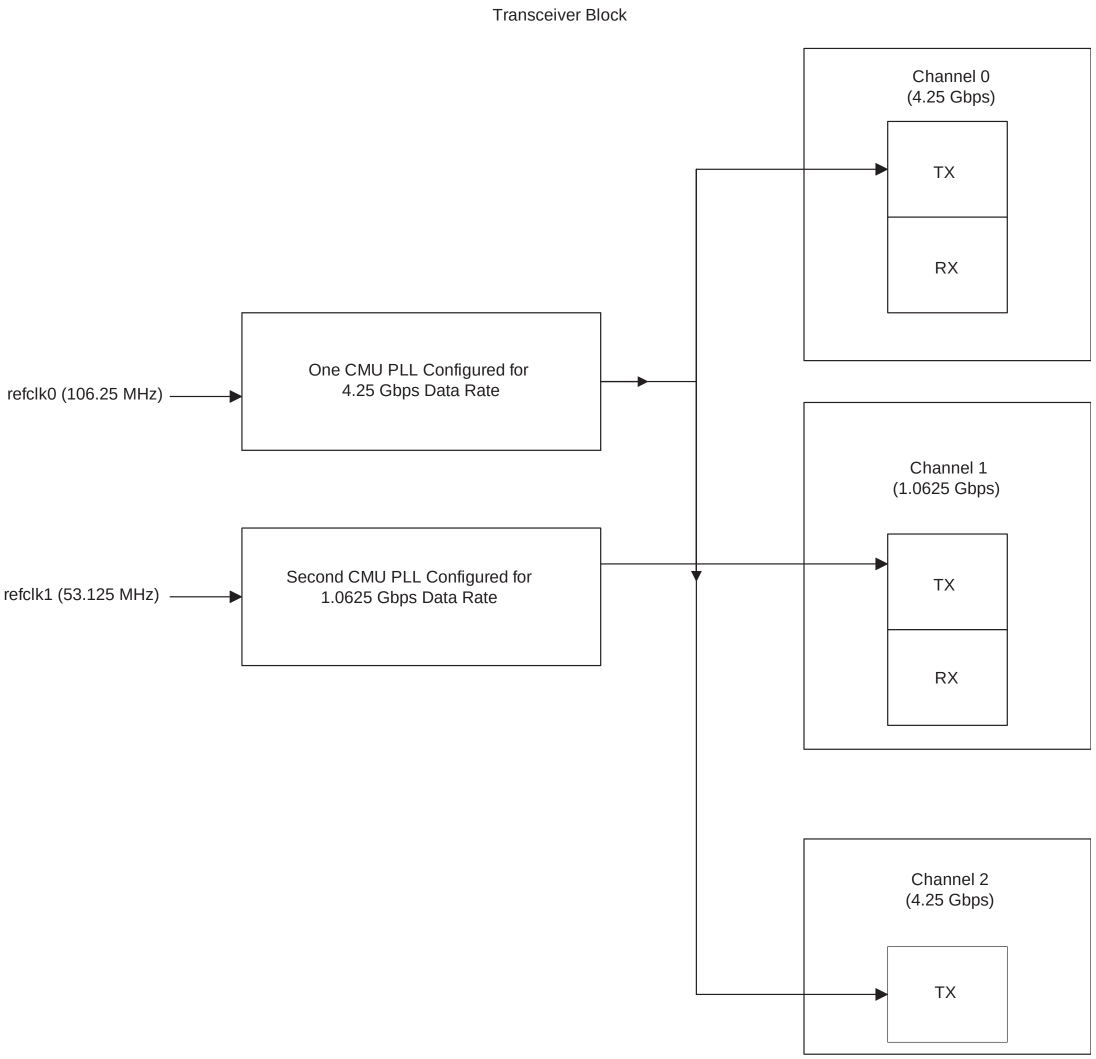
  Yes.

For more information about clocking the transmitter and receiver channel data path for this type of configuration, refer to the "Non-Bonded Transceiver Clocking" section of the *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*.

Figure 2–3 shows the top-level block diagram of the transceiver setup for this design example.

☞ Note that the transmitter side receives its clocks from the CMU PLLs. The receiver side contains its dedicated CDR that provides the high-speed serial and low-speed parallel clocks to its PMA and PCS blocks, respectively. Figure 2–3 shows the clocking only for the transmitter side.

Figure 2–3. Top-Level Block Diagram of the Transceiver Setup



Implementation

Create the transceiver instance using the ALTGX MegaWizard Plug-In Manager.

For a description of the individual fields, refer to the *ALTGX Megafunction User Guide* chapter in volume 3 of the *Stratix IV Device Handbook*. This section discusses setting the values required for this design.

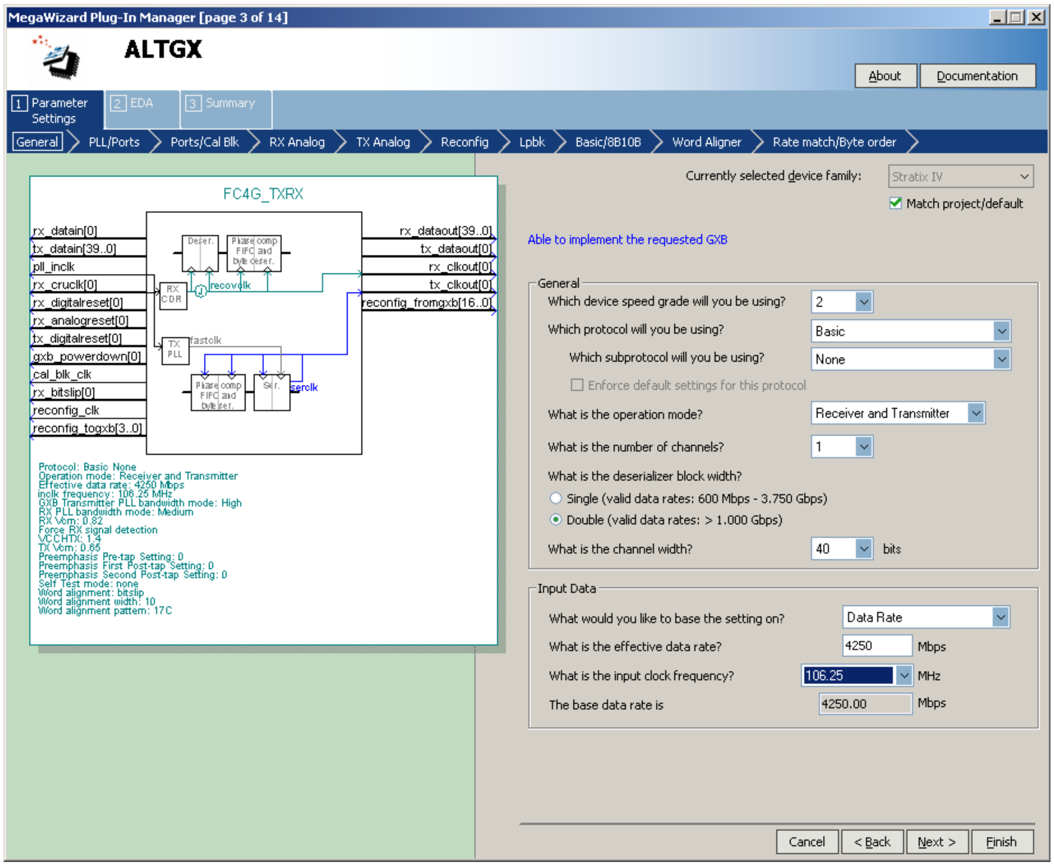
Create the Transceiver Instance for an FC4G Configuration (Channel 0)

Figure 2–4 through Figure 2–14 show the different options available in the ALTGX MegaWizard Plug-In Manager to create the transceiver channel instance for the FC4G data rate. This instance is used for channel 0, with settings on the following screens:

- **General** screen

  The Stratix IV GX transceiver can be configured for fibre channel protocol using Basic mode. Set the fields with the values shown in Figure 2–4.

Figure 2–4. FC4G Instance (General Screen)



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 4250 Mbps
inclk frequency: 106.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip
Word alignment width: 10
Word alignment pattern: 17C

Currently selected device family: Stratix IV
☑ Match project/default

Able to implement the requested GXB

General
Which device speed grade will you be using? 2
Which protocol will you be using? Basic
Which subprotocol will you be using? None
☐ Enforce default settings for this protocol
What is the operation mode? Receiver and Transmitter
What is the number of channels? 1
What is the deserializer block width?
○ Single (valid data rates: 600 Mbps - 3.750 Gbps)
● Double (valid data rates: > 1.000 Gbps)
What is the channel width? 40 bits

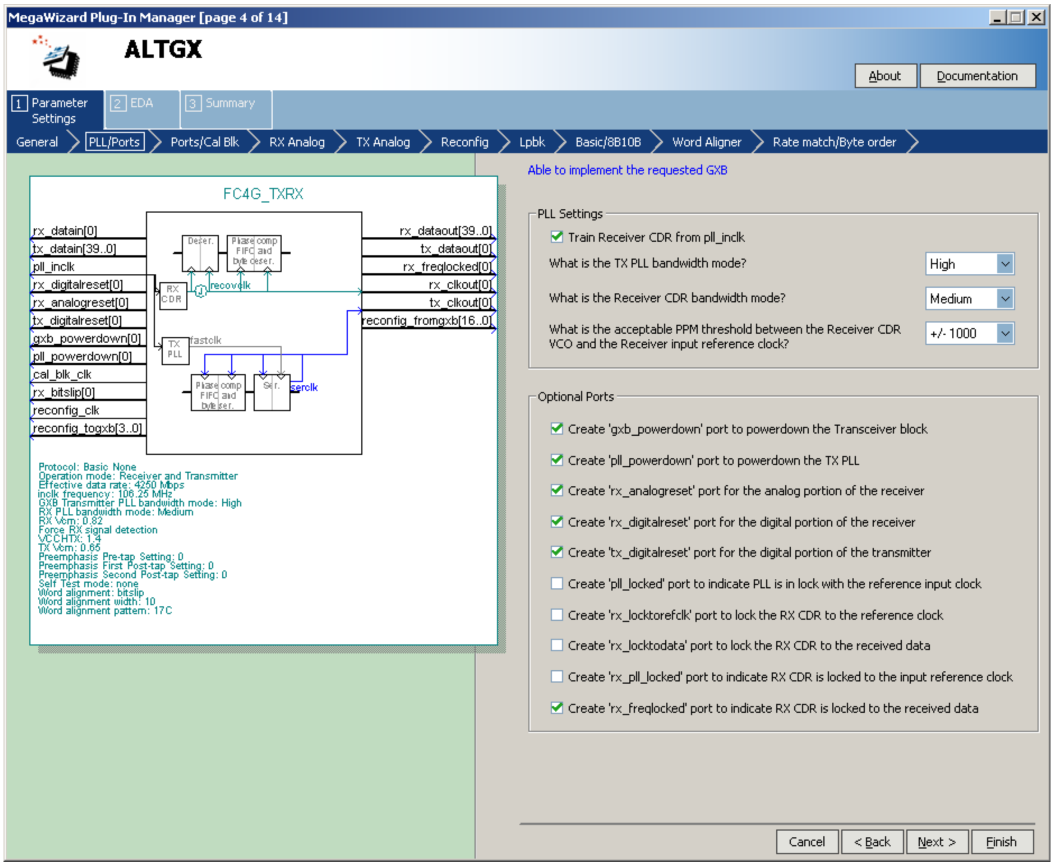
Input Data
What would you like to base the setting on? Data Rate
What is the effective data rate? 4250 Mbps
What is the input clock frequency? 106.25 MHz
The base data rate is 4250.00 Mbps

- **PLL/Ports** screen

  The **Train Receiver CDR from PLL inclk** option is checked, as shown in Figure 2–5.

  When you select this option, the same input reference clock used for the CMU PLL is provided as a training clock to the receiver CDR.

Figure 2–5. FC4G Instance (PLL/Ports) Screen



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 4250 Mbps
inclk frequency: 106.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip
Word alignment width: 10
Word alignment pattern: 17C

Able to implement the requested GXB

PLL Settings

☑ Train Receiver CDR from pll_inclk

What is the TX PLL bandwidth mode? High

What is the Receiver CDR bandwidth mode? Medium

What is the acceptable PPM threshold between the Receiver CDR VCO and the Receiver input reference clock? +/- 1000

Optional Ports

☑ Create 'gxb_powerdown' port to powerdown the Transceiver block

☑ Create 'pll_powerdown' port to powerdown the TX PLL

☑ Create 'rx_analogreset' port for the analog portion of the receiver

☑ Create 'rx_digitalreset' port for the digital portion of the receiver

☑ Create 'tx_digitalreset' port for the digital portion of the transmitter

□ Create 'pll_locked' port to indicate PLL is in lock with the reference input clock

□ Create 'rx_locktorefclk' port to lock the RX CDR to the reference clock

□ Create 'rx_locktodata' port to lock the RX CDR to the received data

□ Create 'rx_pll_locked' port to indicate RX CDR is locked to the input reference clock

☑ Create 'rx_freqlocked' port to indicate RX CDR is locked to the received data

The pll_powerdown signal is checked. This signal allows you to power down the CMU PLL. Use this signal as part of your reset sequence.

The pll_locked signal is checked. This signal indicates whether the CMU PLL is locked to the input reference clock. The user logic waits until the pll_locked signal goes high before transmitting data.

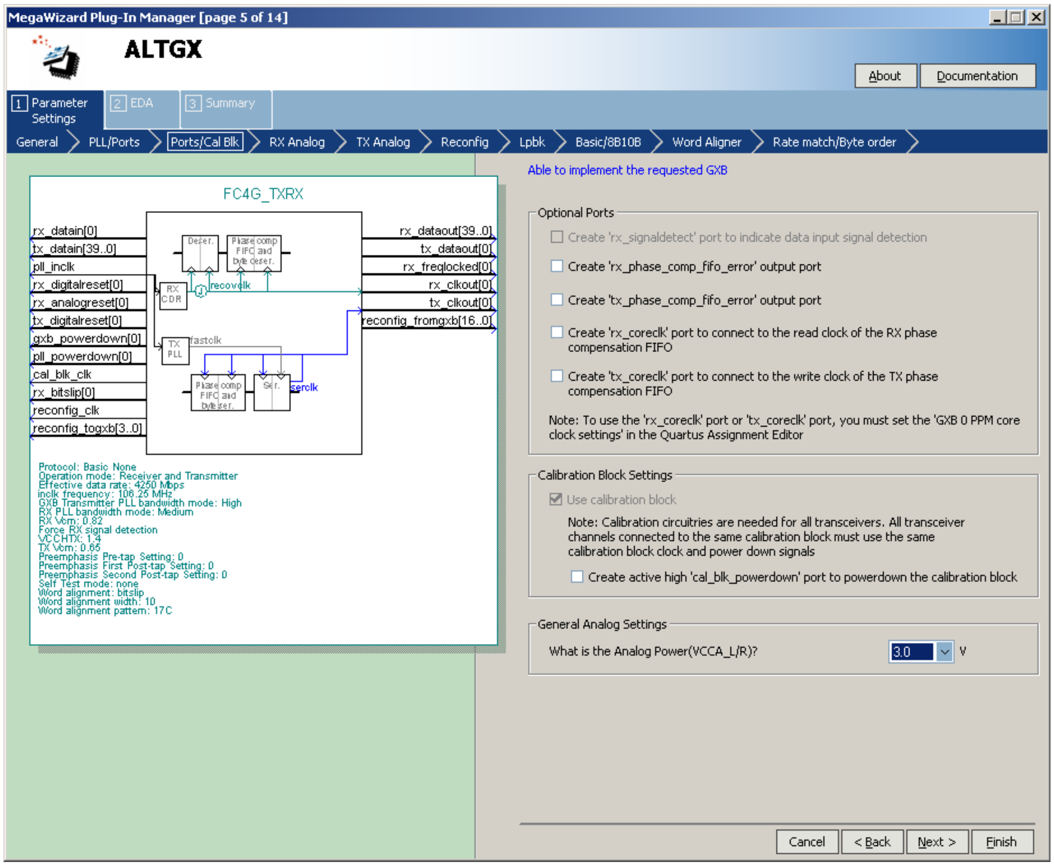
The rx_freqlocked signal is checked. This signal indicates whether the receiver CDR is locked to data. When the receiver CDR is configured in automatic lock mode, assert the rx_digitalreset signal if the rx_freqlocked signal goes low to keep the receiver PCS under reset. Altera recommends specific transceiver reset sequences to ensure proper device operation.

For more information about receiver CDR and lock modes, refer to the “Receiver Modules” section of *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*.

- **Ports /Cal Blk** screen

  The calibration block is required; therefore, this option is always enabled. Select the options shown in Figure 2–6.

Figure 2–6. FC4G Instance (Ports/Cal Blk) Screen



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 4250 Mbps
inclk frequency: 106.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip
Word alignment width: 10
Word alignment pattern: 17C

Able to implement the requested GXB

Optional Ports

☐ Create 'rx_signaldetect' port to indicate data input signal detection

☐ Create 'rx_phase_comp_fifo_error' output port

☐ Create 'tx_phase_comp_fifo_error' output port

☐ Create 'rx_coreclk' port to connect to the read clock of the RX phase compensation FIFO

☐ Create 'tx_coreclk' port to connect to the write clock of the TX phase compensation FIFO

Note: To use the 'rx_coreclk' port or 'tx_coreclk' port, you must set the 'GXB 0 PPM core clock settings' in the Quartus Assignment Editor

Calibration Block Settings

☑ Use calibration block

Note: Calibration circuitries are needed for all transceivers. All transceiver channels connected to the same calibration block must use the same calibration block clock and power down signals

☐ Create active high 'cal_blk_powerdown' port to powerdown the calibration block

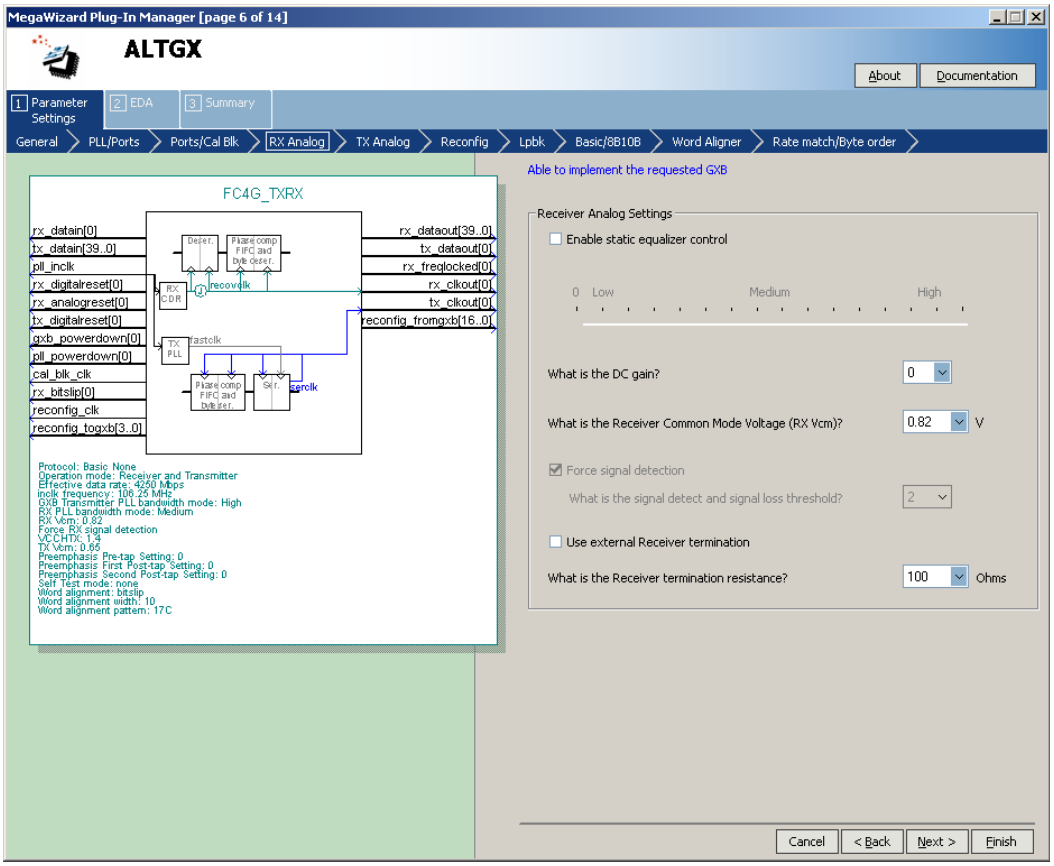
General Analog Settings

What is the Analog Power(VCCA_L/R)? 3.0 V

- **RX Analog** screen

  Select the options shown in Figure 2–7.

Figure 2–7. FC4G Instance (RxAnalog) Screen



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 4250 Mbps
inclk frequency: 106.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip
Word alignment width: 10
Word alignment pattern: 17C

Able to implement the requested GXB

Receiver Analog Settings

☐ Enable static equalizer control

What is the DC gain? 0

What is the Receiver Common Mode Voltage (RX Vcm)? 0.82 V

☑ Force signal detection

What is the signal detect and signal loss threshold? 2

☐ Use external Receiver termination

What is the Receiver termination resistance? 100 Ohms

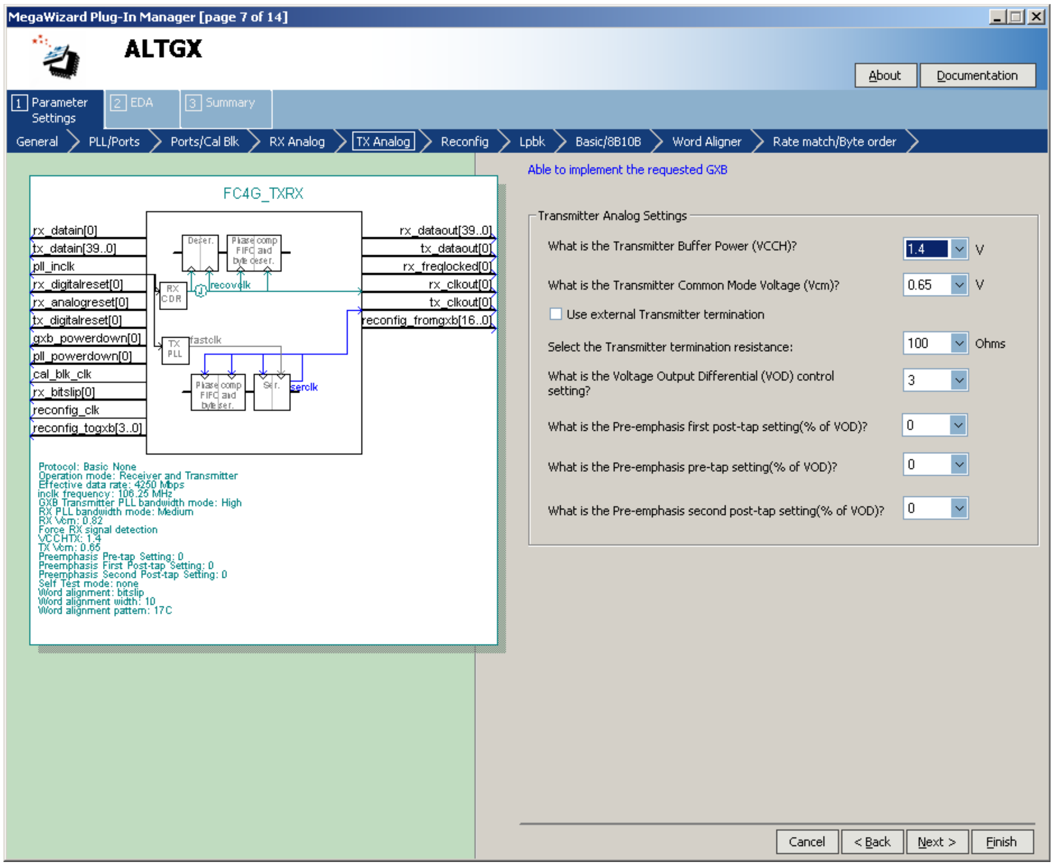
For a description of the individual fields, refer to the *ALTGX Megafunction User Guide* chapter in volume 3 of the *Stratix IV Device Handbook*.

- **TX Analog** screen

  Select the output differential voltage and common mode voltage values that meet the fibre channel protocol specification

  If you intend to transmit data through faulty interconnects, select the pre-emphasis settings shown in Figure 2–8.

Figure 2–8. FC4G Instance (TX Analog) Screen



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 4250 Mbps
inclk frequency: 106.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip
Word alignment width: 10
Word alignment pattern: 17C

Able to implement the requested GXB

Transmitter Analog Settings

What is the Transmitter Buffer Power (VCCH)? 1.4 V

What is the Transmitter Common Mode Voltage (Vcm)? 0.65 V

□ Use external Transmitter termination

Select the Transmitter termination resistance: 100 Ohms

What is the Voltage Output Differential (VOD) control setting? 3

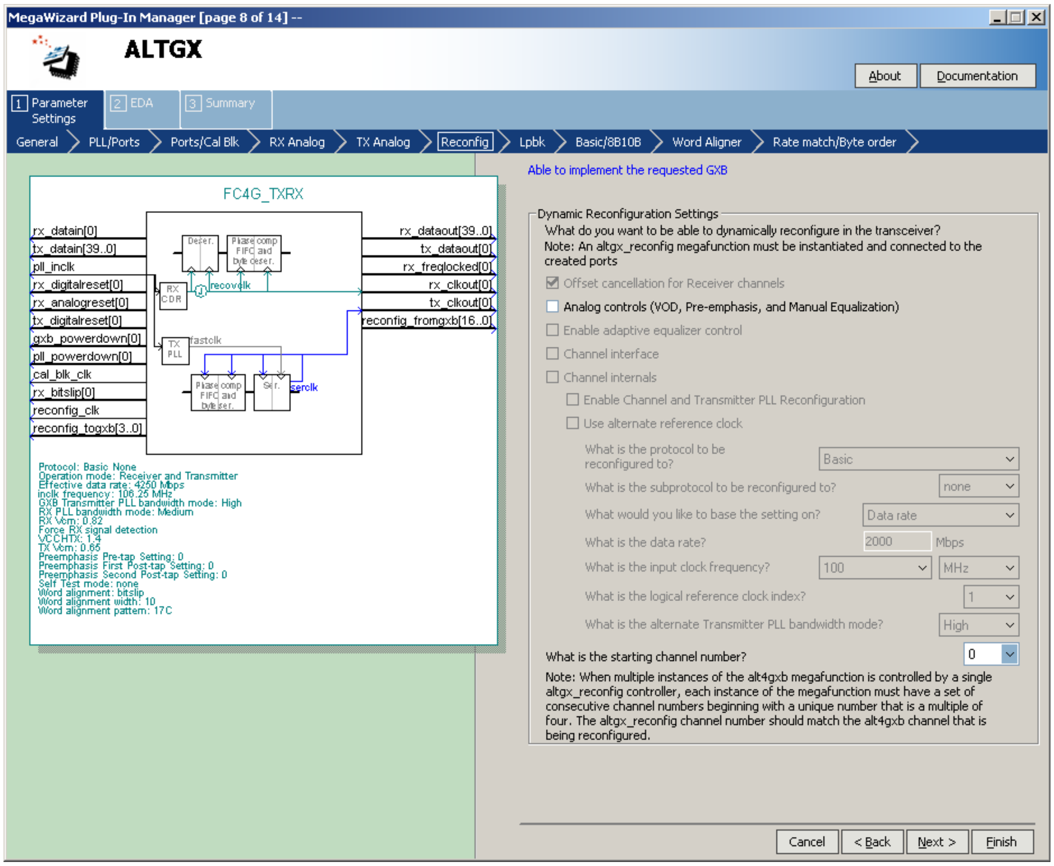
What is the Pre-emphasis first post-tap setting(% of VOD)? 0

What is the Pre-emphasis pre-tap setting(% of VOD)? 0

What is the Pre-emphasis second post-tap setting(% of VOD)? 0

For more information about pre-emphasis settings, refer to the *DC and Switching Characteristics of Stratix IV Devices* chapter in volume 4 of the *Stratix IV Device Handbook*.

Figure 2–9. FC4G Instance (Reconfig) Screen



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 4250 Mbps
inclk frequency: 106.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip
Word alignment width: 10
Word alignment pattern: 17C

Able to implement the requested GXB

Dynamic Reconfiguration Settings

What do you want to be able to dynamically reconfigure in the transceiver?
Note: An altgx_reconfig megafunction must be instantiated and connected to the created ports

☑ Offset cancellation for Receiver channels
☐ Analog controls (VOD, Pre-emphasis, and Manual Equalization)
☐ Enable adaptive equalizer control
☐ Channel interface
☐ Channel internals
☐ Enable Channel and Transmitter PLL Reconfiguration
☐ Use alternate reference clock

What is the protocol to be reconfigured to? Basic
What is the subprotocol to be reconfigured to? none
What would you like to base the setting on? Data rate
What is the data rate? 2000 Mbps
What is the input clock frequency? 100 MHz
What is the logical reference clock index? 1
What is the alternate Transmitter PLL bandwidth mode? High

What is the starting channel number? 0

Note: When multiple instances of the alt4gxb megafunction is controlled by a single altgx_reconfig controller, each instance of the megafunction must have a set of consecutive channel numbers beginning with a unique number that is a multiple of four. The altgx_reconfig channel number should match the alt4gxb channel that is being reconfigured.

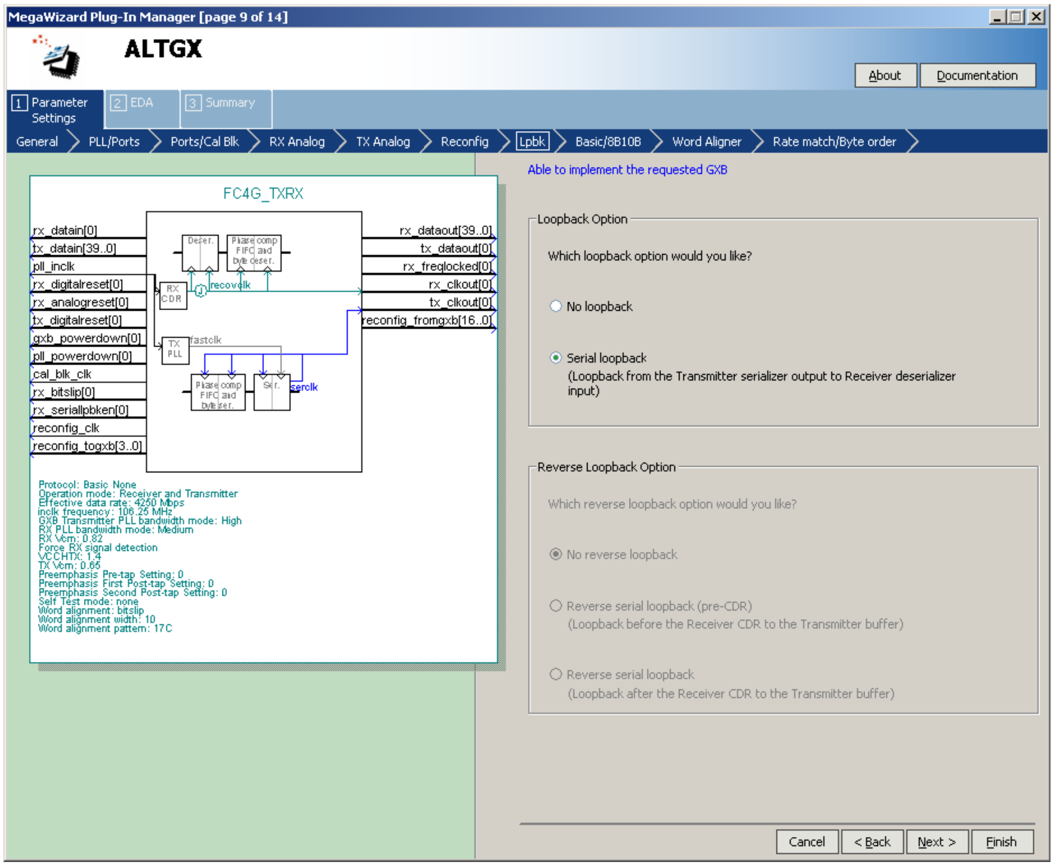
Because offset cancellation is required for receiver channels, the **Offset Cancellation for Receiver Channels** option is automatically checked. Make sure that you connect the `reconfig_fromgxb` and `reconfig_togxb` ports with the dynamic reconfiguration controller.

Set the starting channel number to **0**. For more information about selecting the starting channel numbers, refer to the Logical Channel Addressing section of the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Handbook*.

- **Lpbk** screen

  The serial loopback option is enabled, as shown in Figure 2–10.

Figure 2–10. FC4G Instance – Lpbk Screen



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 4250 Mbps
inclk frequency: 106.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip
Word alignment width: 10
Word alignment pattern: 17C

Able to implement the requested GXB

Loopback Option

Which loopback option would you like?

○ No loopback

● Serial loopback
(Loopback from the Transmitter serializer output to Receiver deserializer input)

Reverse Loopback Option

Which reverse loopback option would you like?

● No reverse loopback

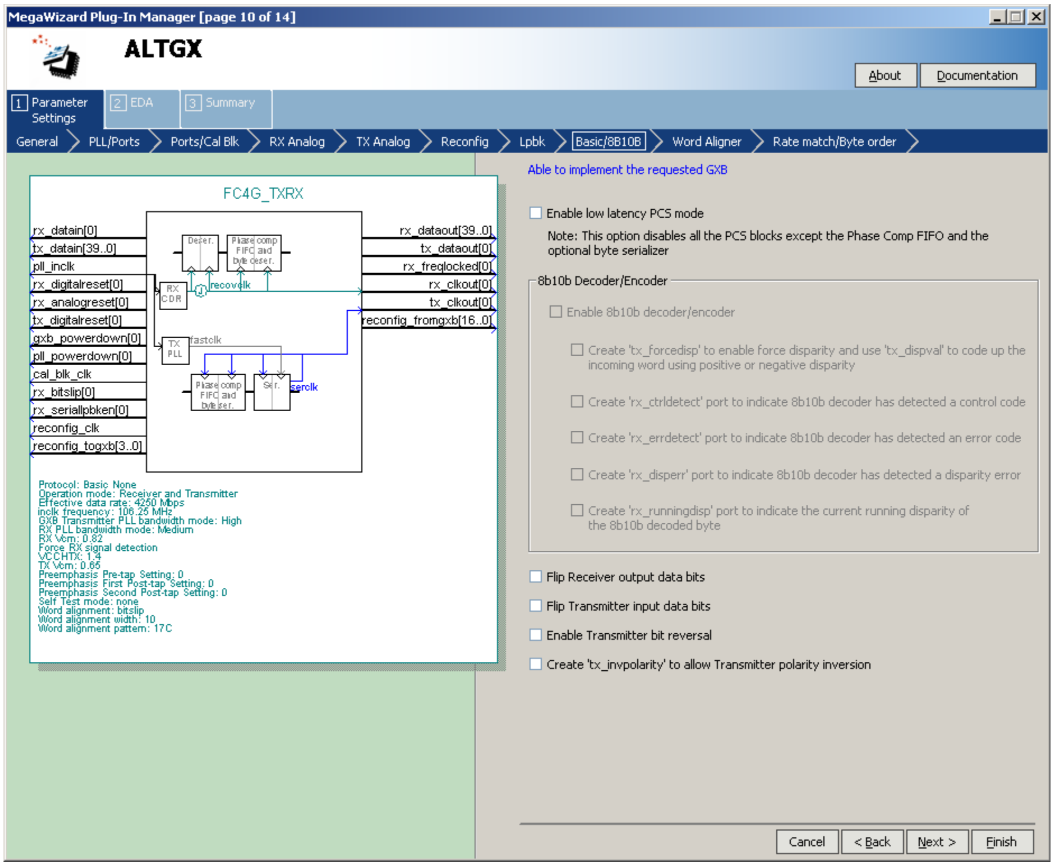
○ Reverse serial loopback (pre-CDR)
(Loopback before the Receiver CDR to the Transmitter buffer)

○ Reverse serial loopback
(Loopback after the Receiver CDR to the Transmitter buffer)

- **Basic/8B10B** screen

The Basic/8B10B screen is shown in Figure 2–11.

Figure 2–11. FC4G Instance - Basic /8B10B



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 4250 Mbps
inclk frequency: 106.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip
Word alignment width: 10
Word alignment pattern: 17C

Able to implement the requested GXB

☐ Enable low latency PCS mode

Note: This option disables all the PCS blocks except the Phase Comp FIFO and the optional byte serializer

8b10b Decoder/Encoder

☐ Enable 8b10b decoder/encoder

☐ Create 'tx_forcedisp' to enable force disparity and use 'tx_dispval' to code up the incoming word using positive or negative disparity

☐ Create 'rx_ctrldetect' port to indicate 8b10b decoder has detected a control code

☐ Create 'rx_errdetect' port to indicate 8b10b decoder has detected an error code

☐ Create 'rx_disperr' port to indicate 8b10b decoder has detected a disparity error

☐ Create 'rx_runningdisp' port to indicate the current running disparity of the 8b10b decoded byte

☐ Flip Receiver output data bits

☐ Flip Transmitter input data bits

☐ Enable Transmitter bit reversal

☐ Create 'tx_invpolarity' to allow Transmitter polarity inversion

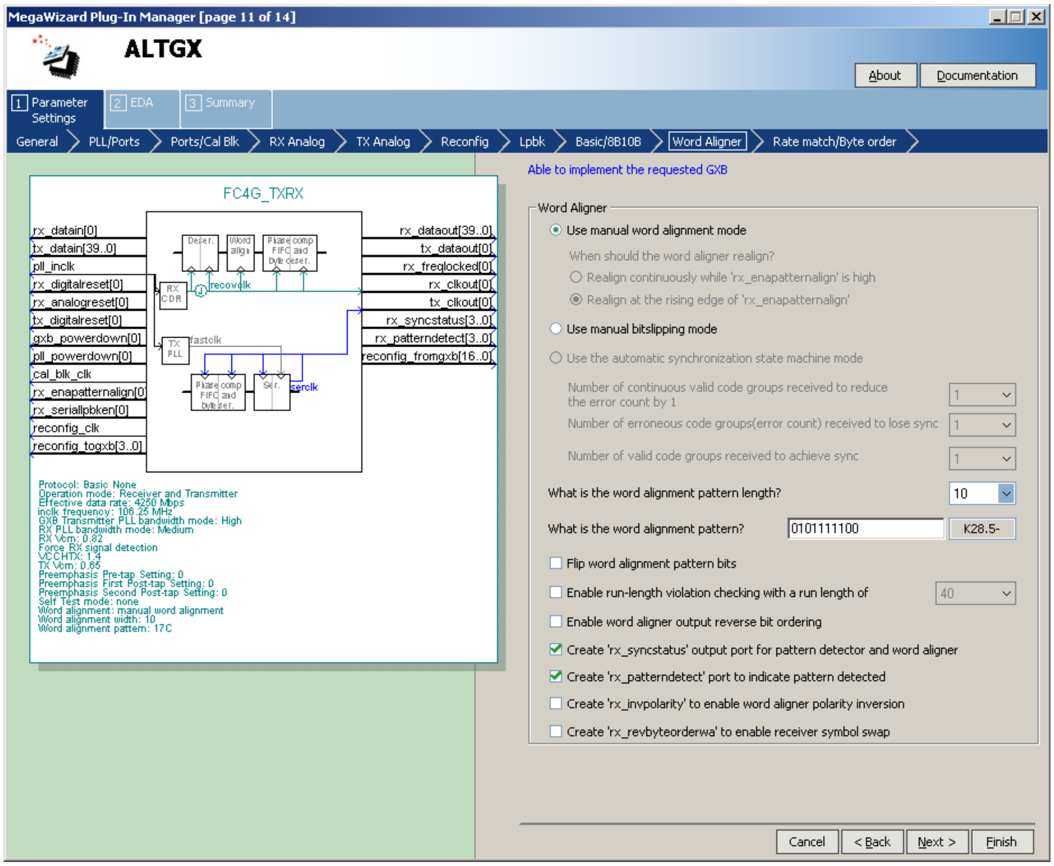
Note to Figure 2–11:

(1) The 8B/10B encoder is not compatible with the fibre channel protocol application, so the option is unchecked.

- **Word Aligner** screen

  The fibre channel protocol requires that K28.5 be used to align the byte boundary. In the **What is the word alignment pattern?** field, set one of the 10-bit disparity values to **K28.5**. The word aligner automatically detects when the other disparity value is received.

Figure 2–12. FC4G Instance – Word Aligner Screen



MegaWizard Plug-In Manager [page 11 of 14]

ALTGX

Able to implement the requested GXB

Word Aligner

● Use manual word alignment mode

When should the word aligner realign?

○ Realign continuously while 'rx_enapatternalign' is high

● Realign at the rising edge of 'rx_enapatternalign'

○ Use manual bitslipping mode

○ Use the automatic synchronization state machine mode

Number of continuous valid code groups received to reduce the error count by 1: 1

Number of erroneous code groups(error count) received to lose sync: 1

Number of valid code groups received to achieve sync: 1

What is the word alignment pattern length? 10

What is the word alignment pattern? 0101111100 K28.5-

☐ Flip word alignment pattern bits

☐ Enable run-length violation checking with a run length of 40

☐ Enable word aligner output reverse bit ordering

☑ Create 'rx_syncstatus' output port for pattern detector and word aligner

☑ Create 'rx_patterndetect' port to indicate pattern detected

☐ Create 'rx_invpolarity' to enable word aligner polarity inversion

☐ Create 'rx_revbyteorderwa' to enable receiver symbol swap

FC4G_TXRX

Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 4250 Mbps
inclk frequency: 106.25 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: manual word alignment
Word alignment width: 10
Word alignment pattern: 17C

The `rx_patterndetect` and the `rx_syncstatus` signals are selected. The `rx_patterndetect` signal indicates whenever the word alignment pattern is detected in the word boundary. Click **Finish** to exit the MegaWizard Plug-In Manager.

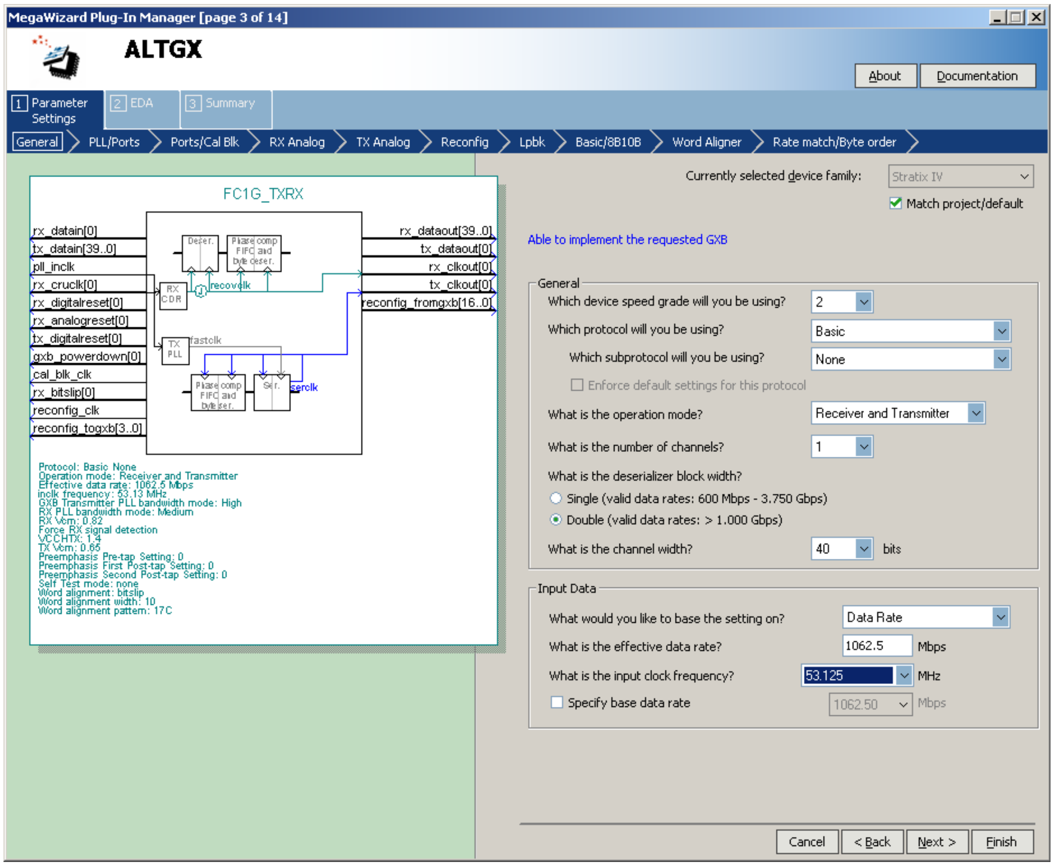
Create the Transceiver Instance for an FC1G Configuration (Channel 1)

Creating the instance for FC1G is very similar to that of the FC4G configuration, with the following changes:

- **General** screen

Set the values shown in Figure 2–13, and in the **Reconfig** screen set the starting channel number to **4**.

Figure 2–13. FC1G Instance (Channel 1) – General Screen



Protocol: Basic None
Operation mode: Receiver and Transmitter
Effective data rate: 1062.5 Mbps
inclk frequency: 53.13 MHz
GXB Transmitter PLL bandwidth mode: High
RX PLL bandwidth mode: Medium
RX Vcm: 0.82
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip
Word alignment width: 10
Word alignment pattern: 17C

Currently selected device family: Stratix IV

☑ Match project/default

Able to implement the requested GXB

General

Which device speed grade will you be using? 2

Which protocol will you be using? Basic

Which subprotocol will you be using? None

☐ Enforce default settings for this protocol

What is the operation mode? Receiver and Transmitter

What is the number of channels? 1

What is the deserializer block width?

○ Single (valid data rates: 600 Mbps - 3.750 Gbps)

● Double (valid data rates: > 1.000 Gbps)

What is the channel width? 40 bits

Input Data

What would you like to base the setting on? Data Rate

What is the effective data rate? 1062.5 Mbps

What is the input clock frequency? 53.125 MHz

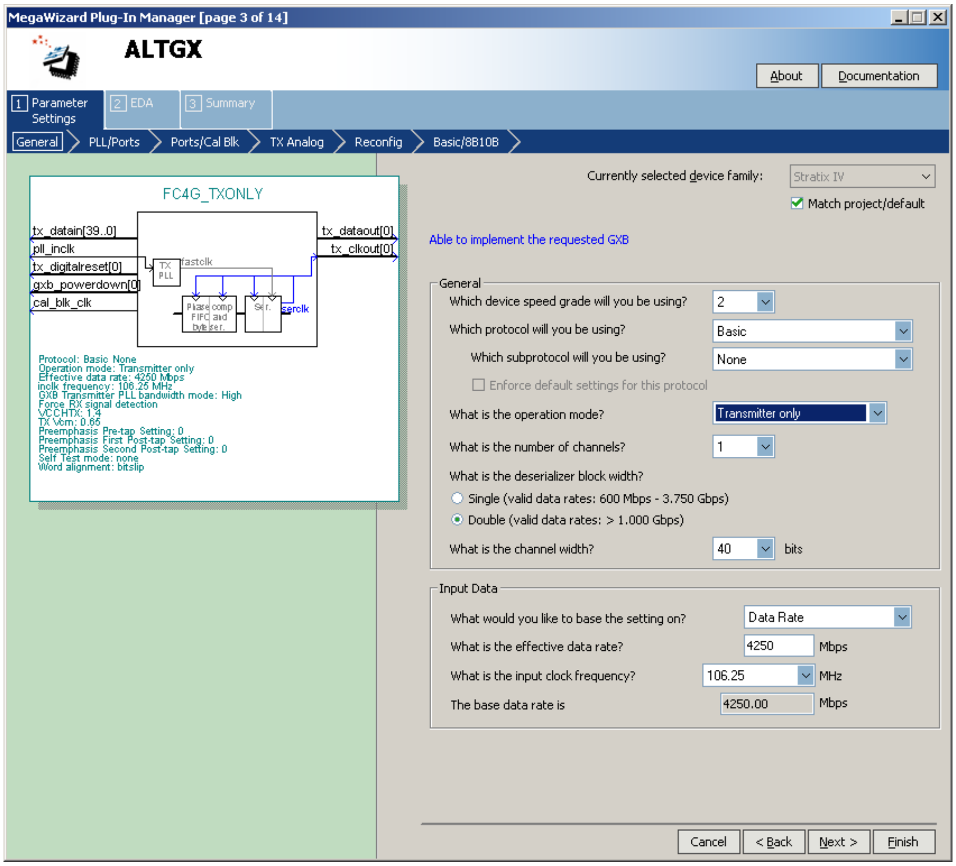
☐ Specify base data rate 1062.50 Mbps

Create the Instance for an FC4G Configuration - “Transmitter Only” (Channel 2)

This configuration is similar to the channel 0 configuration, with the following changes:

- Set the operation mode to **Transmitter Only**, as shown in Figure 2–14. Note that since this is a transmitter only instance, all the options relevant to the receiver are not available in the ALTGX MegaWizard Plug-In Manager.

Figure 2–14. FC4G_TXONLY Instance (Channel 1) – General Screen



Protocol: Basic None
Operation mode: Transmitter only
Effective data rate: 4250 Mbps
inclk frequency: 106.25 MHz
GXB Transmitter PLL bandwidth mode: High
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip

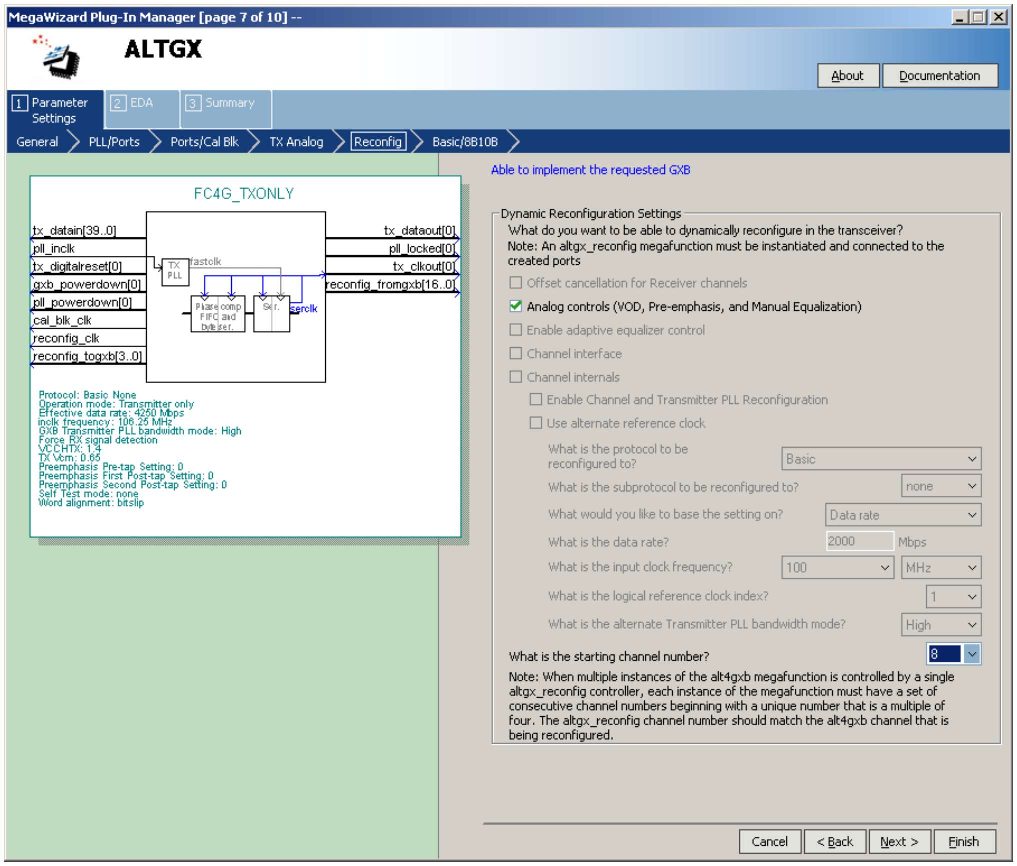
Currently selected device family: Stratix IV
☑ Match project/default

Able to implement the requested GXB

General
Which device speed grade will you be using? 2
Which protocol will you be using? Basic
Which subprotocol will you be using? None
☐ Enforce default settings for this protocol
What is the operation mode? Transmitter only
What is the number of channels? 1
What is the deserializer block width?
○ Single (valid data rates: 600 Mbps - 3.750 Gbps)
● Double (valid data rates: > 1.000 Gbps)
What is the channel width? 40 bits

Input Data
What would you like to base the setting on? Data Rate
What is the effective data rate? 4250 Mbps
What is the input clock frequency? 106.25 MHz
The base data rate is 4250.00 Mbps

Figure 2–15. FC4G_TXONLY Instance (Reconfig) Screen



Protocol: Basic None
Operation mode: Transmitter only
Effective data rate: 4250 Mbps
inclk frequency: 106.25 MHz
GXB Transmitter PLL bandwidth mode: High
Force RX signal detection
VCCHTX: 1.4
TX Vcm: 0.65
Preemphasis Pre-tap Setting: 0
Preemphasis First Post-tap Setting: 0
Preemphasis Second Post-tap Setting: 0
Self Test mode: none
Word alignment: bitslip

Able to implement the requested GXB

Dynamic Reconfiguration Settings

What do you want to be able to dynamically reconfigure in the transceiver?
Note: An altgx_reconfig megafunction must be instantiated and connected to the created ports

- ☐ Offset cancellation for Receiver channels
- ☑ Analog controls (VOD, Pre-emphasis, and Manual Equalization)
- ☐ Enable adaptive equalizer control
- ☐ Channel interface
- ☐ Channel internals
  - ☐ Enable Channel and Transmitter PLL Reconfiguration
  - ☐ Use alternate reference clock

What is the protocol to be reconfigured to? Basic
What is the subprotocol to be reconfigured to? none
What would you like to base the setting on? Data rate
What is the data rate? 2000 Mbps
What is the input clock frequency? 100 MHz
What is the logical reference clock index? 1
What is the alternate Transmitter PLL bandwidth mode? High

What is the starting channel number? 8

Note: When multiple instances of the alt4gxb megafunction is controlled by a single altgx_reconfig controller, each instance of the megafunction must have a set of consecutive channel numbers beginning with a unique number that is a multiple of four. The altgx_reconfig channel number should match the alt4gxb channel that is being reconfigured.

In the **Reconfig** screen (Figure 2–15), select the **Analog controls** option even if you do not intend to dynamically reconfigure the PMA controls. Selecting this option is required for this example scenario because of the following reasons:

- For a transmitter only instance, offset cancellation is not available and therefore the `reconfig_fromgxb` and `reconfig_togxb` ports are not available.
- The other two instances (containing a receiver channel) have these ports available because offset cancellation is automatically enabled.
- If one transceiver instance has the `reconfig_fromgxb/reconfig_togxb` ports enabled, the Quartus II software requires the other transceiver instances to have these ports enabled to combine them in the same transceiver block. Therefore, for this transmiter only instance, **Analog options...** must be selected.

Set the starting channel number to **8**.

For more information about the requirements to combine multiple transceiver instances, refer to the “Combining Transceiver Instances with Dynamic Reconfiguration Enabled” section in the *Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Handbook*.

Create the Dynamic Reconfiguration Controller (ALTGX_Reconfig) Instance

This section only discusses the relevant options that must be set to implement the application. Refer to the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Handbook* for additional information.

Figure 2–16 shows the options that you must set (assuming that you do not require dynamic reconfiguration of the PMA controls in the transceiver channels).

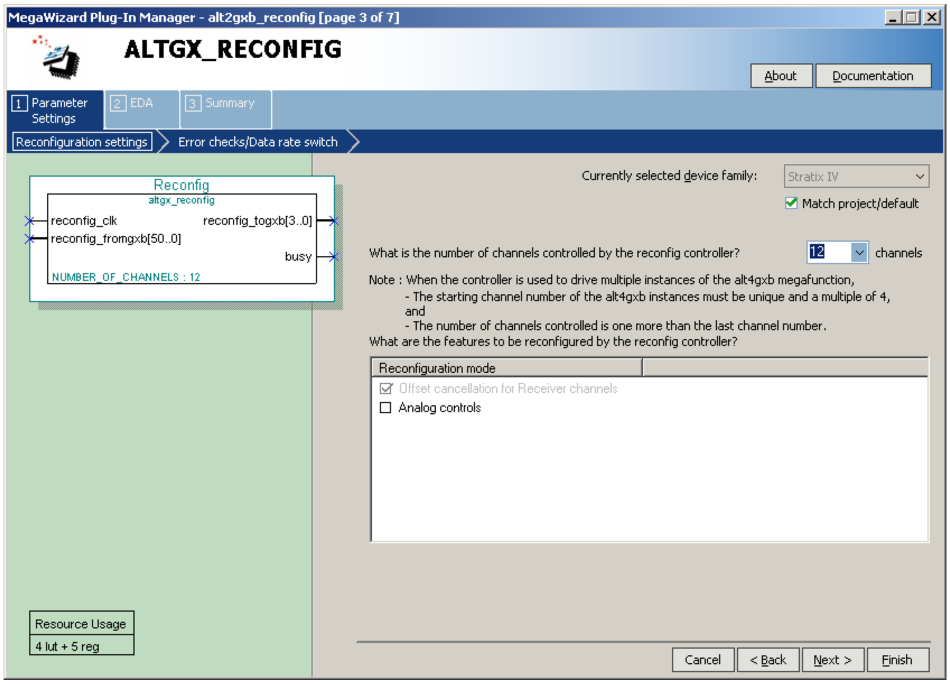
For more information about selecting the **Number of Channels** option, refer to the Total Number of Transceiver Channels Controlled by the ALTGX_Reconfig Instance section in the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Handbook*.

Connect:

- reconfig_fromgxb[16:0] of ALTGX_Reconfig instance to FC4G instance (channel0)
- reconfig_fromgxb[33:17] to FC1G instance (channel1)
- reconfig_fromgxb[50:34] to FC4G-Transmitter only instance (channel2)

Connect the reconfig_togxb[3:0] of the ALTGX_Reconfig instance to all three transceiver instances.

Figure 2–16. ALTGX_Reconfig (Reconfiguration Settings) Screen



MegaWizard Plug-In Manager - alt2gxb_reconfig [page 3 of 7]

ALTGX_RECONFIG

Currently selected device family: Stratix IV

☑ Match project/default

What is the number of channels controlled by the reconfig controller? 12 channels

Note : When the controller is used to drive multiple instances of the alt4gxb megafunction,
- The starting channel number of the alt4gxb instances must be unique and a multiple of 4, and
- The number of channels controlled is one more than the last channel number.

What are the features to be reconfigured by the reconfig controller?

Reconfiguration mode

☑ Offset cancellation for Receiver channels

☐ Analog controls

Resource Usage
4 lut + 5 reg

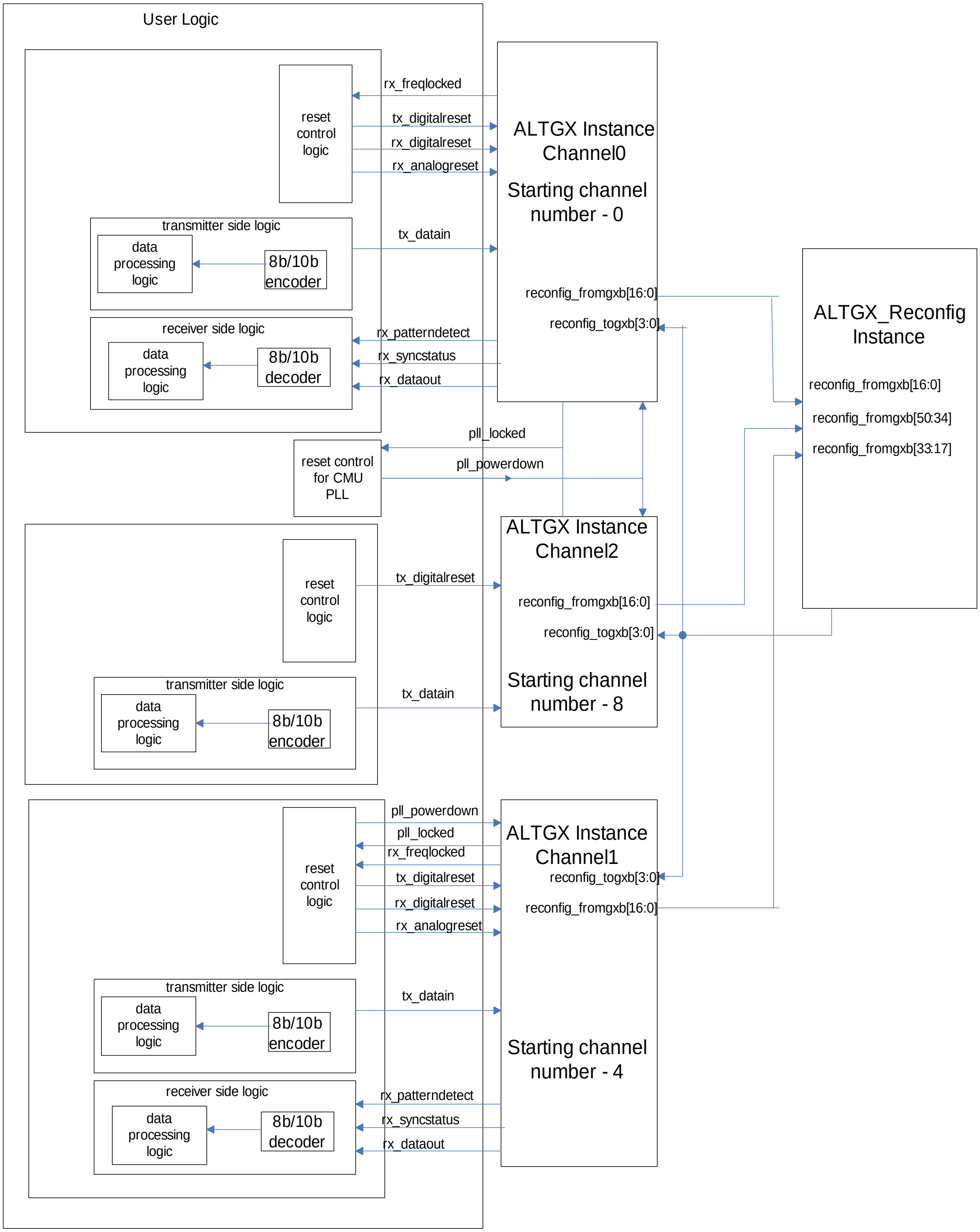
Create the Reset Logic to Control the FPGA Fabric and Transceivers

The design requires independent control on each channel. Altera recommends creating independent reset control logic for each channel.

In this design, channel 0 and channel 2 share the same CMU PLL (since they are configured at the same data rate) and channel 1 uses the second CMU PLL. When you create a transmitter only or receiver and transmitter instance, the ALTGX MegaWizard Plug-In Manager provides a `pll_powerdown` signal to reset the CMU PLL that provides clocks to the transmitter channel. In this design example, since channels 0 and 2 share the same CMU PLL, drive the `pll_powerdown` port of channel 0 and channel 2 ALTGX instance from the same logic.

Channels 0, 1, and 2 have separate `rx_digitalreset`, `rx_analogreset`, and `tx_digitalreset` signals. Figure 2–17 shows the interface between the three transceiver instances and the FPGA fabric.

Figure 2–17. Transceiver – FPGA Fabric Interface



Create the Data Processing and Other User Logic

For this design example, the 8B/10B encoder and decoder must be implemented in the FPGA fabric. Figure 2–17 shows the block diagram of the logic on the transmitter and receiver side and the system logic controls for all channels in the FPGA fabric. This block diagram is a representation of a typical system and may not exactly show the different blocks in a practical application. Interface all the logic blocks with the transceiver.

If you would like to add SignalTap for verification, first complete the synthesis, then add the transceiver-FPGA fabric or other user logic signals in SignalTap. Then compile the design to generate the **.sof** file.

Compilation

Assign pins for the input and output signals in your design. The Quartus II software versions 8.1 and earlier do not allow pin assignments for the Stratix IV GX device.

Set the OCT values for the transceiver serial pins, add timing constraints for the clocks and data paths in your logic, then compile the design.

Simulating the Design

To simulate the design, follow the steps outlined in "Simulation" on page 2–10.

Documents Referenced

This chapter references the following documents:

- *ALTGX Megafunction User Guide* chapter in volume 3 of the *Stratix IV Device Handbook*
- *Configuring Multiple Protocols and Data Rates* chapter in volume 2 of the *Stratix IV Device Handbook*
- *DC and Switching Characteristics of Stratix IV Devices* chapter in volume 4 of the *Stratix IV Device Handbook*
- *In-System Design Debugging* section in volume 3 of the *Quartus II Development Software Handbook*
- *Quartus II Development Software Handbook*
- *Reset Control and Powerdown* chapter in volume 2 of the *Stratix IV Device Handbook*
- *Stratix IV GX Device Family Pin Connection Guidelines*
- *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*
- *Stratix IV Transceiver Architecture* chapter in volume 2 of the *Stratix IV Device Handbook*
- *Stratix IV Transceiver Clocking* chapter in volume 2 of the *Stratix IV Device Handbook*

Document Revision History

Table 2–3 shows the revision history for this document.

Table 2–3. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.0	■ Added "Transceiver Configuration" on page 2–3 ■ Added "Create Dynamic Reconfiguration Controller Instances" on page 2–7 ■ "Dynamic Reconfiguration" on page 2–14 ■ Updated "Create the Instance for an FC4G Configuration - "Transmitter Only" (Channel 2)" on page 2–27 ■ Added "Create the Dynamic Reconfiguration Controller (ALTGX_Reconfig) Instance" on page 2–29 ■ Updated Figure 2–1, Figure 2–4, Figure 2–5, Figure 2–6, Figure 2–7, Figure 2–8, Figure 2–10, Figure 2–11, Figure 2–12, Figure 2–13, and Figure 2–14 ■ Added Figure 2–9, Figure 2–15, and Figure 2–16	—
May 2008 v1.0	Initial release	—

3. Porting a Stratix II GX Transceiver Design to a Stratix IV GX Device

SIV53003-1.0

Introduction

This chapter provides information about how to move a Stratix® II transceiver design into a Stratix IV GX device. To make the process as simple as possible, a report that helps you make the transfer and a method for transfer are discussed:

- "Quartus II Software Migration Guide"— a report that helps you understand the changes required to port a design developed for a Stratix II GX transceiver over to a Stratix IV GX device. The report only provides information for transceivers.
- "Opening ALT2GXB Instances in the ALTGX MegaWizard Plug-In Manager" — a method for converting a Stratix II GX transceiver instance to a Stratix IV GX transceiver instance.

Quartus II Software Migration Guide

The migration guide is a report generated by the Quartus® II 8.0 software when you compile a copy of your existing Stratix II GX design and set the device family to Stratix IV GX. The report shows the changes that you must implement in the transceiver instances to target a Stratix IV GX device. The report also shows the differences in the transceiver behavior that can be expected when a current Stratix II GX design is migrated to a Stratix IV GX device. The Quartus II software does not modify the existing design when you compile a Stratix II GX design targeting a Stratix IV GX device. The Quartus II software ignores all pin assignments. The **.pin**, **.sof**, and **.pof** files are not generated.

☞ Keep the original copy of the existing Stratix II GX project in a different directory so that you can make changes to the current copy of your design to target the Stratix IV GX device.

The migration guide report consists of four tabs:

- "GXB Behavior Differences Tab" on page 3–2
- "GXB Assignment Changes Needed Tab" on page 3–2
- "GXB Parameter Changes Needed Tab" on page 3–3
- "GXB Port Changes Needed Tab" on page 3–4

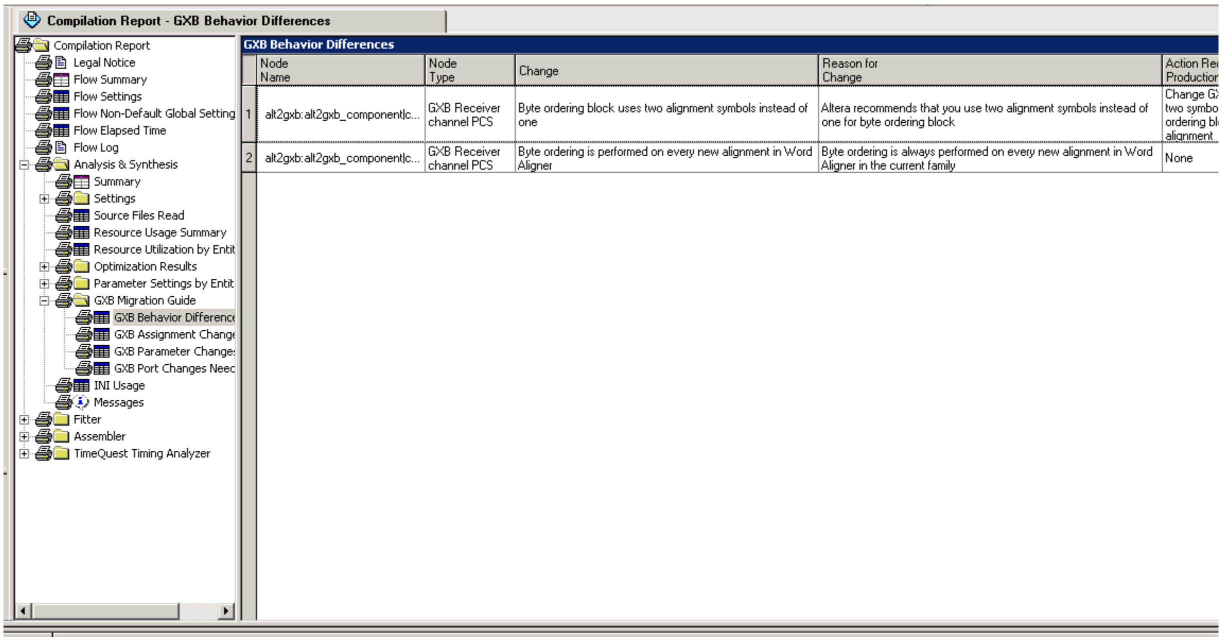
GXB Behavior Differences Tab

The **Behavior Differences** tab shows the differences in behavior in the transceiver functional blocks between Stratix II GX and Stratix IV GX devices. For example, for the `rx_syncstatus`-based byte ordering, in a Stratix II GX device, the byte ordering block triggers only once after the `rx_syncstatus` goes high. To retrigger the byte ordering block, `rx_digitalreset` must be asserted. In Stratix IV GX devices, the byte ordering retriggers every time `rx_syncstatus` is asserted. If you have enabled the `rx_syncstatus`-based byte ordering feature in your Stratix II GX design, the migration report shows (in the **Change** column) that you can expect the byte ordering block to retrigger every time `rx_syncstatus` is asserted in a Stratix IV GX device.

☞ Based on the behavioral differences between the Stratix II GX device and the Stratix IV GX device shown in the report, determine whether the existing user logic requires changes to the interface to target the design for the Stratix IV GX device.

Figure 3–1 displays the **GXB Behavior Differences** tab.

Figure 3–1. GXB Behavior Differences Tab



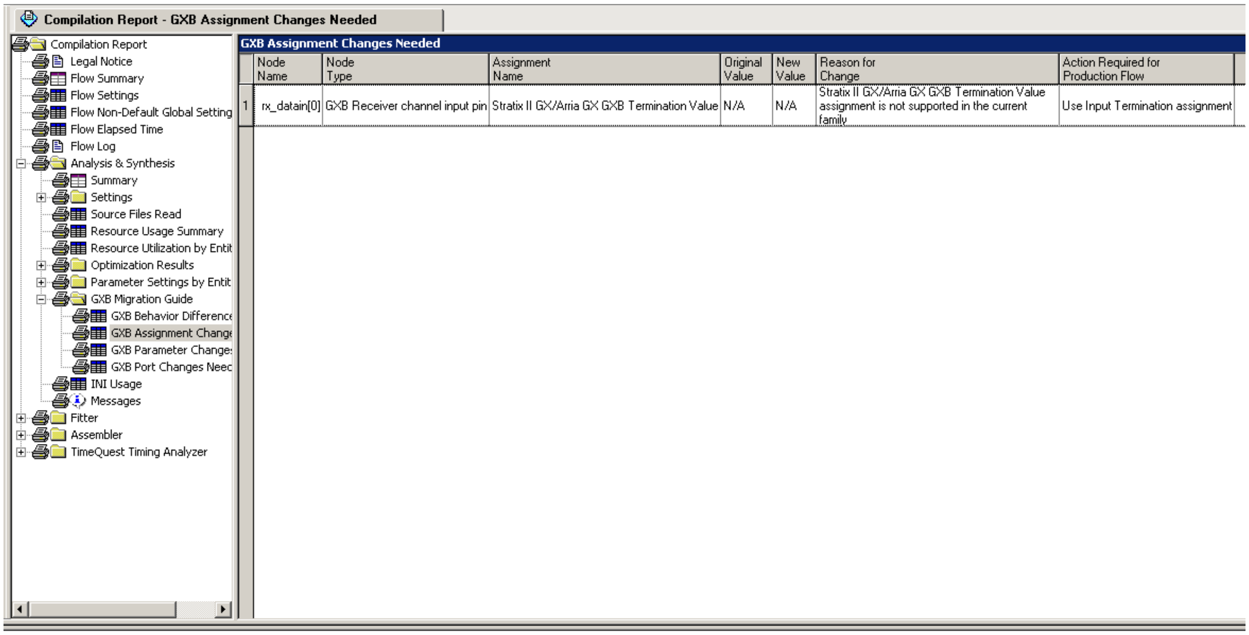
GXB Behavior Differences

	Node Name	Node Type	Change	Reason for Change	Action Re(Productior
1	alt2gxb:alt2gxb_component|c...	GXB Receiver channel PCS	Byte ordering block uses two alignment symbols instead of one	Altera recommends that you use two alignment symbols instead of one for byte ordering block	Change G) two symbo ordering bl alignment
2	alt2gxb:alt2gxb_component|c...	GXB Receiver channel PCS	Byte ordering is performed on every new alignment in Word Aligner	Byte ordering is always performed on every new alignment in Word Aligner in the current family	None

GXB Assignment Changes Needed Tab

The **GXB Assignment Changes Needed** tab shows the changes required in the assignments to target the Stratix II GX design for a Stratix IV GX device. Figure 3–2 shows that the transceiver on chip termination setting has changed. The **Action Required for Production Flow** (last column) shows the new assignment setting that is required to compile the design for a Stratix IV GX device. You can add the new assignment using the Assignment Editor.

Figure 3–2. GXB Assignment Changes Needed Tab



GXB Assignment Changes Needed

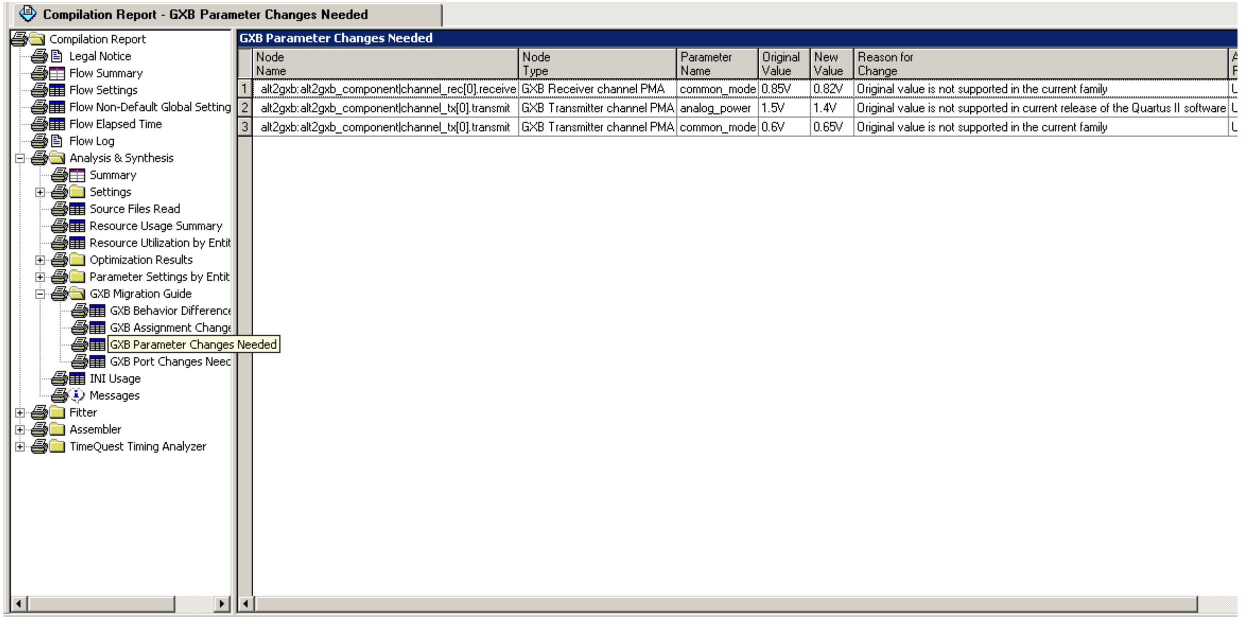
	Node Name	Node Type	Assignment Name	Original Value	New Value	Reason for Change	Action Required for Production Flow
1	rx_datain[0]	GXB Receiver channel input pin	Stratix II GX/Arria GX GXB Termination Value	N/A	N/A	Stratix II GX/Arria GX GXB Termination Value assignment is not supported in the current family	Use Input Termination assignment

GXB Parameter Changes Needed Tab

The **GXB Parameter Changes Needed** tab shows the parameter differences you can expect between a Stratix II GX and a Stratix IV GX transceiver instance. For example, the receiver common mode voltage (V_{CM}) in a Stratix II GX device is 0.85 V. In a Stratix IV GX device the voltage is 0.82 V. The parameter name column shows the parameters that have changed. The original value and the new value column show the values in the Stratix II GX and Stratix IV GX device, respectively. In Figure 3–3, the GXB Migration Guide report shows the receiver common mode voltage for the Stratix II GX device as 0.85 V. This is the value that you selected in the ALT2GXB MegaWizard Plug-In Manager for the Stratix II GX device.

You can select the appropriate parameters for the Stratix IV GX transceiver instance in the ALTGX MegaWizard Plug-In Manager interface.

Figure 3–3. GXB Parameter Changes Needed Tab



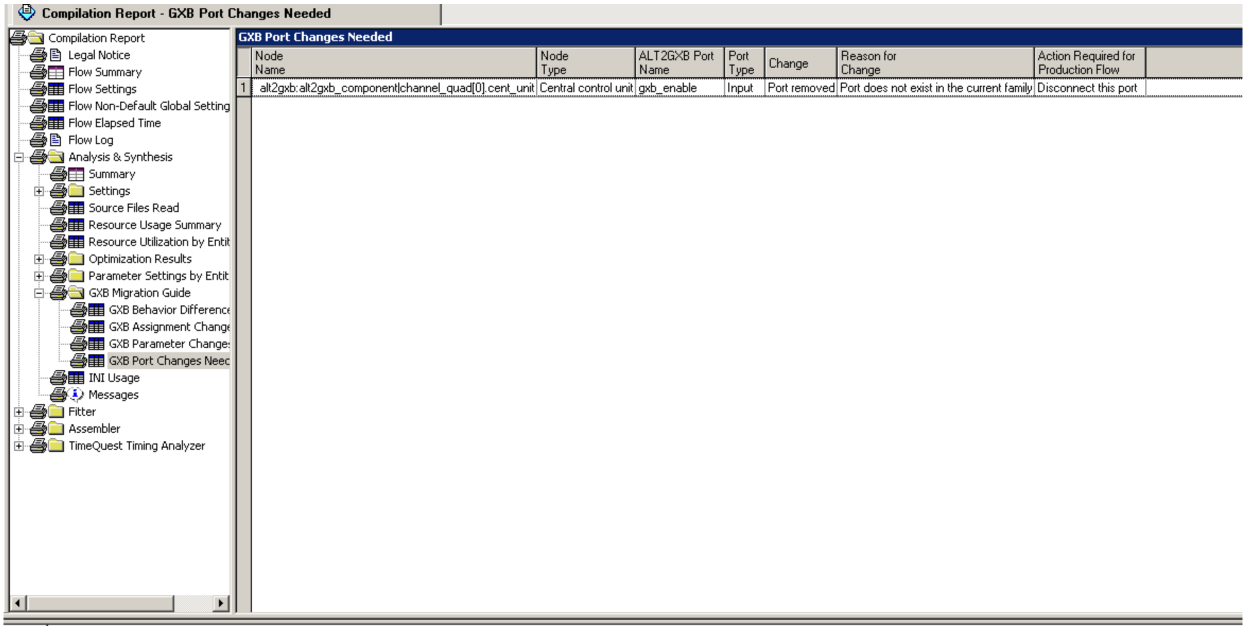
GXB Parameter Changes Needed

	Node Name	Node Type	Parameter Name	Original Value	New Value	Reason for Change	A F
1	alt2gxb:alt2gxb_component|channel_rec[0].receive	GXB Receiver channel PMA	common_mode	0.85V	0.82V	Original value is not supported in the current family	L
2	alt2gxb:alt2gxb_component|channel_tx[0].transmit	GXB Transmitter channel PMA	analog_power	1.5V	1.4V	Original value is not supported in current release of the Quartus II software	L
3	alt2gxb:alt2gxb_component|channel_tx[0].transmit	GXB Transmitter channel PMA	common_mode	0.6V	0.65V	Original value is not supported in the current family	L

GXB Port Changes Needed Tab

The **GXB Port Changes Needed** tab shows the ports that are different between Stratix II GX and Stratix IV GX transceiver instances. For example, in a Stratix II GX device, you can enable an optional `gxb_enable` input port through the ALT2GXB MegaWizard Plug-In Manager. In a Stratix IV GX device, this port is not supported. If you compile a Stratix II GX design that has the `gxb_enable` port, the **Port Differences** tab shows that this port is not available in the Stratix IV GX device, as shown in Figure 3–4.

Figure 3–4. GXB Port Changes Needed Tab



GXB Port Changes Needed

	Node Name	Node Type	ALT2GXB Port Name	Port Type	Change	Reason for Change	Action Required for Production Flow
1	alt2gxb:alt2gxb_component|channel_quad[0].cent_unit	Central control unit	gxb_enable	Input	Port removed	Port does not exist in the current family	Disconnect this port

To save the report file, right-click on the tabs under **GXB Migration Guide** and select **Save current report section** as a **.txt** or a **.csv** (to open in Microsoft Excel) format.

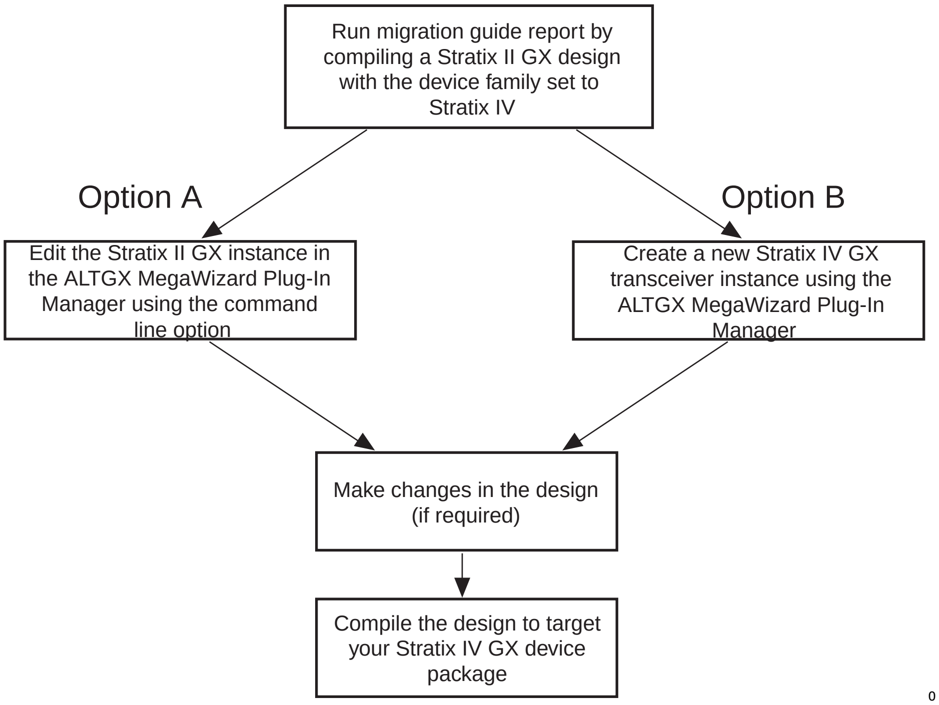
Targeting the Stratix II GX Design for a Stratix IV GX Device

After you review the migration report and understand the differences between Stratix II GX transceivers and Stratix IV GX transceivers, follow these steps:

1. Create a Stratix IV GX transceiver instance using the ALTGX MegaWizard Plug-In Manager. You can use one of the following methods:
   - Open the Stratix II GX transceiver instance in the ALTGX MegaWizard Plug-In Manager using the steps discussed in the section "Opening ALT2GXB Instances in the ALTGX MegaWizard Plug-In Manager" on page 3–6 and generate the Stratix IV transceiver instance.
   - Use the ALTGX MegaWizard Plug-In Manager to create a new interface, set the family to **Stratix IV** and select the settings you require on the appropriate screens.
2. Make modifications in your transceiver instance port list (if required).
3. Make modifications in your design to interface with the transceiver (if required).
4. Provide pin assignments for the input and output signals in the design.
5. Compile the design.

Figure 3–5 shows the steps required to compile a Stratix II GX design for a Stratix IV GX device.

Figure 3–5. Compiling a Stratix II GX Transceiver Design for a Stratix IV GX Device


Run migration guide report by compiling a Stratix II GX design with the device family set to Stratix IV
Option A
Option B
Edit the Stratix II GX instance in the ALTGX MegaWizard Plug-In Manager using the command line option
Create a new Stratix IV GX transceiver instance using the ALTGX MegaWizard Plug-In Manager
Make changes in the design (if required)
Compile the design to target your Stratix IV GX device package
0

Opening ALT2GXB Instances in the ALTGX MegaWizard Plug-In Manager

Version 8.0 of the Quartus II software provides the ALTGX MegaWizard Plug-In Manager, which can be used to create transceiver channel instances for Stratix II GX, Stratix IV GX, and Arria® GX devices.

The ALT2GXB MegaWizard Plug-In Manager is available for Stratix II GX and Arria GX devices, but not for Stratix IV GX devices.

If you have created a transceiver channel configuration for a Stratix II GX device, you can use the same configuration for a Stratix IV GX device. Edit the Stratix II GX transceiver instance using the ALTGX MegaWizard Plug-In Manager to create a transceiver instance for the Stratix IV GX device.

Complete the following steps to use this method:

1. From the command line, go to the directory where you have the transceiver instance files (**.v** or **.vhd**) and type the following command:
   `>>qmegawiz wizard=ALTGX <`*filename.extension*`>`

   ☞ *filename.extension* refers to the name of the ALT2GXB MegaWizard Plug-in Manager generated wrapper file.

Example 3–1 shows a sample version of this technique.

Example 3–1. Using the Same Stratix II GX Transceiver Channel Configuration for a Stratix IV GX Device

```
>>qmegawiz wizard=ALTGX alt2gxb_transceiverchannel.v
```

This command opens the ALTGX MegaWizard Plug-In Manager.

2. Set the device family to **Stratix IV**.
3. Ensure that the desired options are enabled and the transceiver settings contain the required values (for example, Transmitter common mode voltage - VCM setting) in the different screens of the ALTGX MegaWizard Plug-In Manager.
4. Click **Finish**.

The ALTGX MegaWizard Plug-In Manager generates a wrapper file for the Stratix IV GX device. This file can be used for both synthesis and functional simulation.

☞ When you open a Stratix II GX instance in the ALTGX MegaWizard Plug-In Manager and set the device family to **Stratix IV**, the ALTGX MegaWizard Plug-In Manager converts the settings that you set for your Stratix II GX device to the settings allowed for the Stratix IV GX device.

For example, for a Stratix II GX device, assume you set the receiver common mode voltage setting (VCM) to 0.85 V (RX Analog screen) in the ALT2GXB MegaWizard Plug-In Manager. When you open this instance in the ALTGX MegaWizard Plug-In Manager and set the device to Stratix IV, the ALTGX MegaWizard Plug-In Manager sets the VCM setting to 0.82 V which is the appropriate setting for a Stratix IV GX device.

☞ Check that transceiver settings contain the appropriate values and be sure to enable all the required options for your design.

Document Revision History

Table 3–1 shows the revision history for this document.

Table 3–1. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008, v2.0	Minor text edits.	Minor edits.
July 2008, v1.0	Initial release.	—

4. Stratix IV ALTGX_RECONFIG Megafunction User Guide

SIV53004-1.0

Introduction

The MegaWizard® Plug-In Manager in the Quartus® II software creates or modifies design files that contain custom megafunction variations. These auto-generated MegaWizard files can then be instantiated in a design file. The MegaWizard Plug-In Manager allows you to specify options for the ALTGX_RECONFIG megafunction.

Start the MegaWizard Plug-In Manager using one of the following methods:

- Choose the **MegaWizard Plug-In Manager** command (Tools menu).
- When working in the Block Editor (schematic symbol), open the Edit menu and choose **Insert Symbol**. The **Symbol** dialog box appears. In the **Symbol** dialog box, click **MegaWizard Plug-In Manager**.
- Start the stand-alone version of the MegaWizard Plug-In Manager by typing the following command at the command prompt: `qmegawiz`.

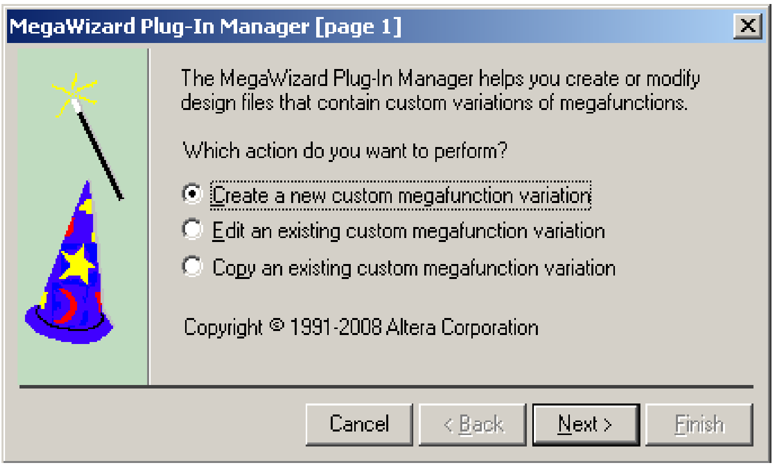
Dynamic Reconfiguration

This section describes the options available on the individual pages of the ALTGX_RECONFIG MegaWizard Plug-In Manager.

The MegaWizard Plug-In Manager provides a warning if any of the settings you choose are illegal.

Figure 4–1 shows the first page of the MegaWizard Plug-In Manager. To generate an ALTGX_RECONFIG custom megafunction variation, select **Create a new custom megafunction variation**. Click **Next**.

Figure 4–1. MegaWizard Plug-In Manager (Page 1)



MegaWizard Plug-In Manager [page 1]

The MegaWizard Plug-In Manager helps you create or modify design files that contain custom variations of megafunctions.

Which action do you want to perform?

● Create a new custom megafunction variation

○ Edit an existing custom megafunction variation

○ Copy an existing custom megafunction variation

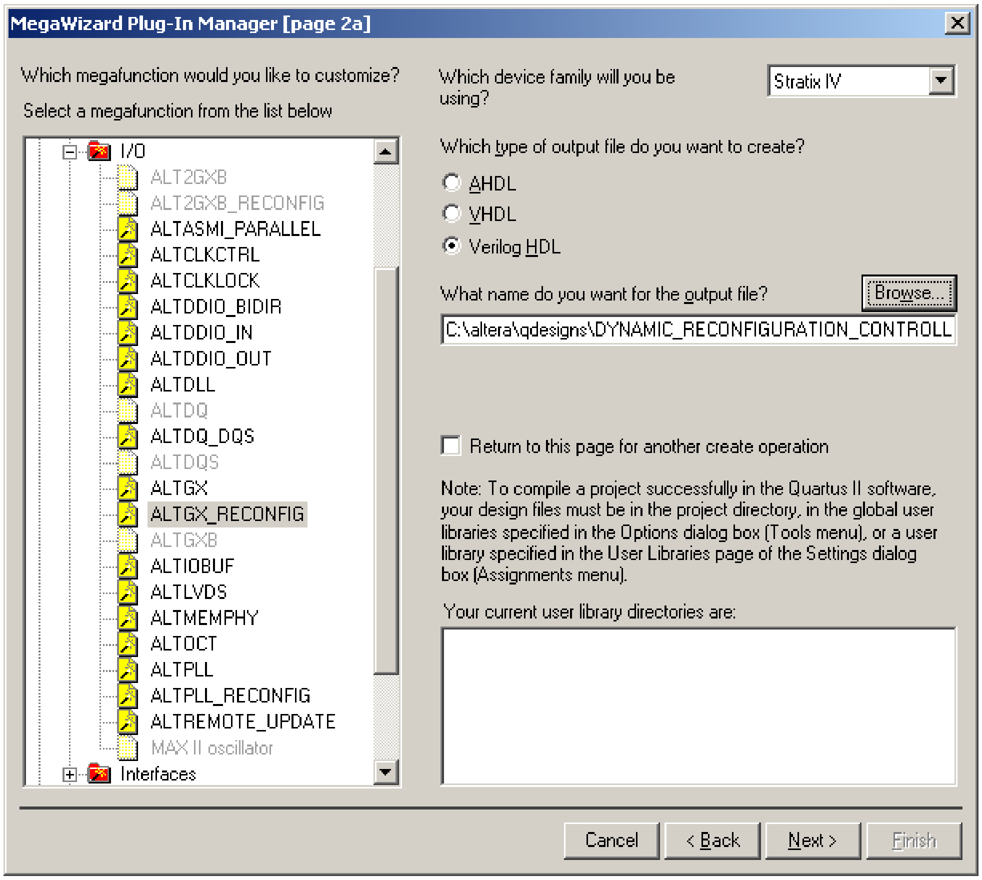
Copyright © 1991-2008 Altera Corporation

Figure 4–2 shows the second page of the MegaWizard Plug-In Manager. Select the following options (click **Next** when you are done):

1. In the list of megafunctions on the left, click the “+” icon beside the I/O item. From the options presented, choose **ALTGX_RECONFIG megafunction**.
2. From the drop-down menu beside **Which device family will you be using?** select **Stratix IV**.
3. From the radio buttons under **Which type of output file do you want to create?** choose your output file format (**AHDL**, **VHDL**, or **Verilog HDL**).
4. In the box beneath **What name do you want for the output file?** enter the file name or click the **Browse** button to search for it.

☞ For the design to compile successfully, always enable the dynamic reconfiguration controller for all the ALTGX instances in the design.

Figure 4–2. MegaWizard Plug-In Manager—ALTGX_RECONFIG (Page 2)



MegaWizard Plug-In Manager [page 2a]

Which megafunction would you like to customize?

Select a megafunction from the list below

Which device family will you be using? Stratix IV

Which type of output file do you want to create?

○ AHDL
○ VHDL
● Verilog HDL

What name do you want for the output file?

C:\altera\qdesigns\DYNAMIC_RECONFIGURATION_CONTROLL

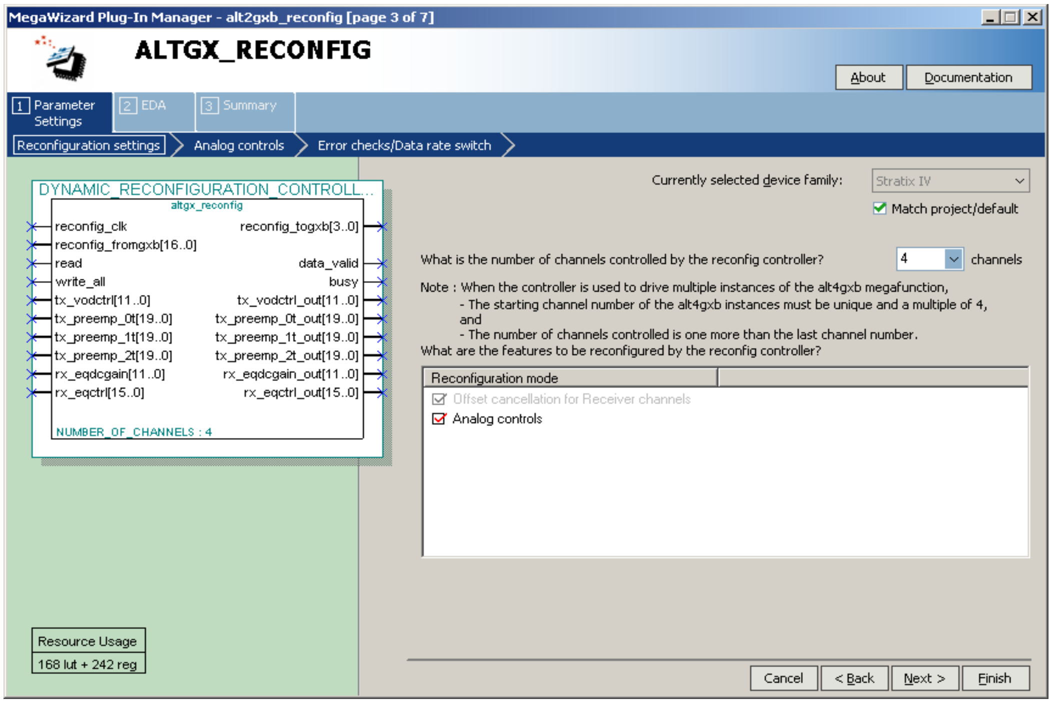
□ Return to this page for another create operation

Note: To compile a project successfully in the Quartus II software, your design files must be in the project directory, in the global user libraries specified in the Options dialog box (Tools menu), or a user library specified in the User Libraries page of the Settings dialog box (Assignments menu).

Your current user library directories are:

Figure 4–3 shows page 3 of the ALTGX_RECONFIG MegaWizard Plug-In Manager. From the drop-down menu, select the number of channels controlled by the dynamic reconfiguration controller. In the Quartus II software version 8.1, you can enable the **Analog controls** feature (for PMA controls reconfiguration) of the dynamic reconfiguration controller.

Figure 4–3. MegaWizard Plug-In Manager - ALTGX_RECONFIG (Reconfiguration Settings) (Page 3)



MegaWizard Plug-In Manager - alt2gxb_reconfig [page 3 of 7]

ALTGX_RECONFIG

Currently selected device family: Stratix IV

☑ Match project/default

What is the number of channels controlled by the reconfig controller? 4 channels

Note : When the controller is used to drive multiple instances of the alt4gxb megafunction,
- The starting channel number of the alt4gxb instances must be unique and a multiple of 4,
and
- The number of channels controlled is one more than the last channel number.

What are the features to be reconfigured by the reconfig controller?

Reconfiguration mode

☐ Offset cancellation for Receiver channels

☑ Analog controls

Resource Usage
168 lut + 242 reg

Table 4–1 describes the available options on page 3 of the MegaWizard Plug-In Manager for your ALTGX_RECONFIG custom megafunction variation. Select the **Match project/default** option if you want to change the device **Currently selected device family** options.

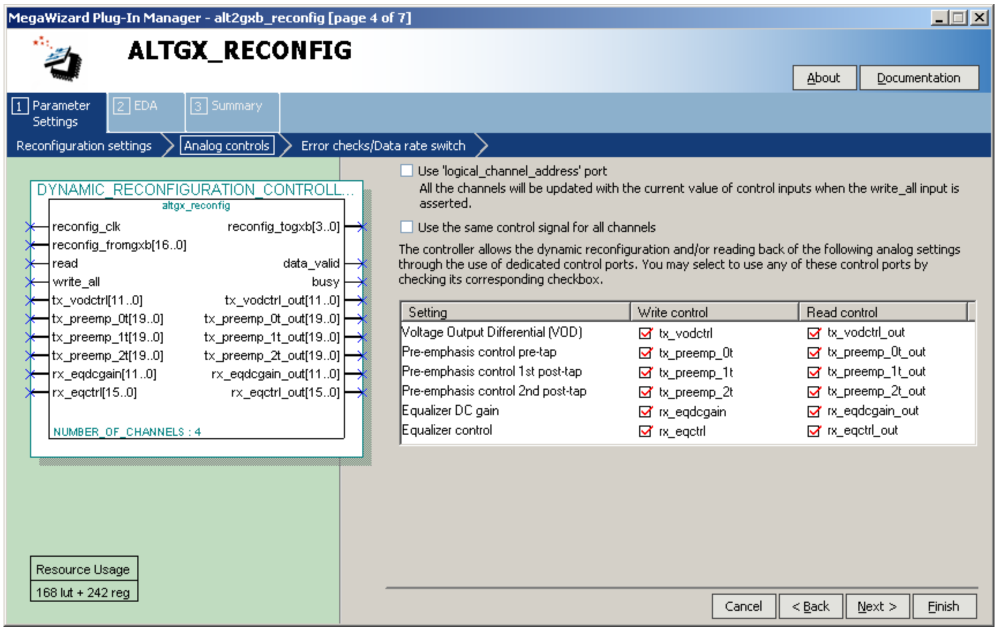
Make your selections on Page 3, then click **Next**.

Table 4–1. MegaWizard Plug-In Manager Options (Page 3)

ALTGX_RECONFIG Setting	Description	Reference
What is the number of channels controlled by the reconfig controller?	Determine the highest logical channel address amongst all the ALTGX instances connected to the ALTGX_RECONFIG instance. Round it up to the next multiple of four and set that number in this option. Depending on this setting, the ALTGX_RECONFIG MegaWizard Plug-in Manager generates the appropriate signal width for the interface signal (`reconfig_fromgxb`) between the ALTGX_RECONFIG and ALTGX instances. It also gives the necessary bus width for all the selected physical media attachment (PMA) signals. Depending on the number of channels set, the resource estimate changes because this is a soft implementation that uses fabric logic resources. The resource estimate is shown in the bottom left of Page 3 of the MegaWizard Plug-in Manager.	"Total Number of Channels Controlled by the ALTGX_RECONFIG Instance" section of the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*.
What are the features to be reconfigured by the reconfig controller?	This feature is always enabled by default: ■ **Offset Cancellation for Receiver Channels—**Once the device powers up, the dynamic reconfiguration controller performs offset cancellation on the receiver portion of all the transceiver channels controlled by it.	"Offset Cancellation Control for Receiver Channels" section in the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*.
	This feature is available for selection: ■ **Analog Controls—**Allows dynamic reconfiguration of PMA controls like Equalization, Pre-emphasis, DC Gain, and VOD.	"PMA Controls Reconfiguration" section of the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*.

Figure 4–4 shows page 4 of the ALTGX_RECONFIG MegaWizard Plug-In Manager.

Figure 4–4. MegaWizard Plug-In Manager—ALTGX_RECONFIG (Analog Controls) (Page 4)



□ Use 'logical_channel_address' port

All the channels will be updated with the current value of control inputs when the write_all input is asserted.

□ Use the same control signal for all channels

The controller allows the dynamic reconfiguration and/or reading back of the following analog settings through the use of dedicated control ports. You may select to use any of these control ports by checking its corresponding checkbox.

Setting	Write control	Read control
Voltage Output Differential (VOD)	☑ tx_vodctrl	☑ tx_vodctrl_out
Pre-emphasis control pre-tap	☑ tx_preemp_0t	☑ tx_preemp_0t_out
Pre-emphasis control 1st post-tap	☑ tx_preemp_1t	☑ tx_preemp_1t_out
Pre-emphasis control 2nd post-tap	☑ tx_preemp_2t	☑ tx_preemp_2t_out
Equalizer DC gain	☑ rx_eqdcgain	☑ rx_eqdcgain_out
Equalizer control	☑ rx_eqctrl	☑ rx_eqctrl_out

Table 4–2 describes the available options on page 4 of the MegaWizard Plug-In Manager for your ALTGX_RECONFIG custom megafunction variation.

Make your selections on page 4, then click **Next**.

Table 4–2. MegaWizard Plug-In Manager Options (Page 4) (Part 1 of 2)

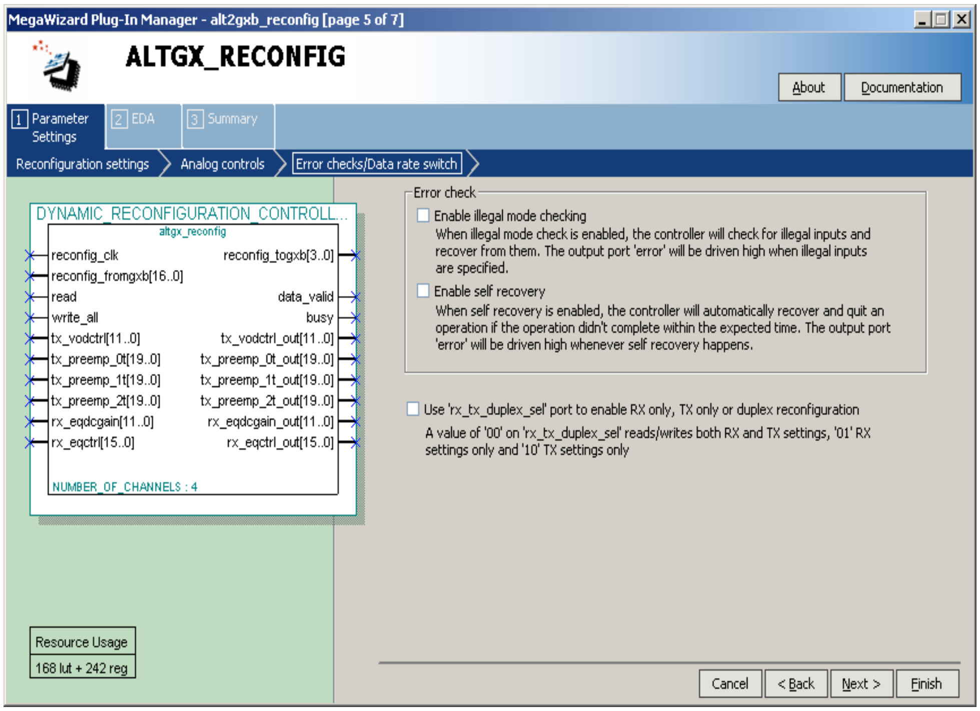
ALTGX_RECONFIG Setting	Description	Reference
Use '`logical_channel_address`' port	This option is available for selection when the number of channels controlled by the ALTGX_RECONFIG instance is more than one. The `logical_channel_address` port is enabled by the ALTGX_RECONFIG MegaWizard Plug-In Manager when you enable this option. The dynamic reconfiguration controller will reconfigure only the channel whose logical channel address is specified at the `logical_channel_address` port. The width of this port is selected by the ALTGX_RECONFIG MegaWizard Plug-In Manager depending on the number of channels controlled by the dynamic reconfiguration controller. The maximum width of the `logical_channel_address` port is 9-bits.	"Dynamically Reconfiguring PMA Controls" section of the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*.
Use the same control signal for all channels	This option is available for selection when the number of channels controlled by the ALTGX_RECONFIG instance is more than one. The dynamic reconfiguration controller will write the same control signals to all the channels connected to it, when you enable this option. This option is not available for selection when you enable the previous **Use 'logical_channel_address' port** option.	

Table 4–2. MegaWizard Plug-In Manager Options (Page 4) (Part 2 of 2)

ALTGX_RECONFIG Setting	Description	Reference
Write Control	The PMA control ports available to write various analog settings to the transceiver channels controlled by the dynamic reconfiguration controller are as follows: ■ `tx_vod_ctrl` — Voltage Output Differential (VOD) — 3-bits per channel ■ `tx_preemp_0t` — Pre-emphasis control pre-tap — 5-bits per channel ■ `tx_preemp_1t` — Pre-emphasis control 1st post-tap — 5-bits per channel ■ `tx_preemp_2t` — Pre-emphasis control 2nd post-tap — 5-bits per channel ■ `rx_eqdcgain` — Equalizer DC gain — 3-bits per channel ■ `rx_eqctrl` — Equalizer control — 4-bits per channel These are optional signals. The signal widths are based on the setting you entered for the **What is the number of channels controlled by the reconfig controller?** option and whether you enable the **Use 'logical_channel_address' port** option. At least one of these PMA control ports must be enabled to configure and use the dynamic reconfiguration controller.	"Dynamically Reconfiguring PMA Controls" section of the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*.
Read Control	The PMA control ports available to read the existing values from the transceiver channels controlled by the dynamic reconfiguration controller are as follows: ■ `tx_vodctrl_out` — Voltage Output Differential (VOD) — 3-bits per channel ■ `tx_preemp_0t_out` — Pre-emphasis control pre-tap — 5-bits per channel ■ `tx_preemp1t_out` — Pre-emphasis control 1st post-tap — 5-bits per channel ■ `tx_preemp_2t_out` — Pre-emphasis control 2nd post-tap — 5-bits per channel ■ `rx_eqdcgain_out` — Equalizer DC gain — 3-bits per channel ■ `rx_eqctrl_out` — Equalizer control — 4-bits per channel These are optional signals. The signal widths are based on the setting you entered for the **What is the number of channels controlled by the reconfig controller?** option and whether you enable the **Use 'logical_channel_address' port** option. The PMA controls are available for selection only if the corresponding write control is selected. Read and write transactions cannot be performed simultaneously.	

Figure 4–5 shows page 5 of the ALTGX_RECONFIG MegaWizard Plug-In Manager.

Figure 4–5. MegaWizard Plug-In Manager—ALTGX_RECONFIG (Error Checks/Data Rate Switch)



Error check

□ Enable illegal mode checking

When illegal mode check is enabled, the controller will check for illegal inputs and recover from them. The output port 'error' will be driven high when illegal inputs are specified.

□ Enable self recovery

When self recovery is enabled, the controller will automatically recover and quit an operation if the operation didn't complete within the expected time. The output port 'error' will be driven high whenever self recovery happens.

□ Use 'rx_tx_duplex_sel' port to enable RX only, TX only or duplex reconfiguration

A value of '00' on 'rx_tx_duplex_sel' reads/writes both RX and TX settings, '01' RX settings only and '10' TX settings only

Table 4–3 describes the available options on page 5 of the MegaWizard Plug-In Manager for your ALTGX_RECONFIG custom megafunction variation.

Make your selections on page 5, then click **Next**.

Table 4–3. MegaWizard Plug-In Manager Options (Page 5) (Part 1 of 2)

ALTGX_RECONFIG Setting	Description	Reference
Enable illegal mode checking	When you select this option, the ALTGX_RECONFIG MegaWizard provides the `error` output port. The dynamic reconfiguration controller checks for specific unsupported options within two `reconfig_clk` cycles, de-asserts the `busy` signal, and asserts the `error` output port for two `reconfig_clk` cycles. The dynamic reconfiguration controller does not execute the unsupported operation.	"Error Indication in the ALTGX_RECONFIG MegaWizard Plug-In Manager" section of the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*.

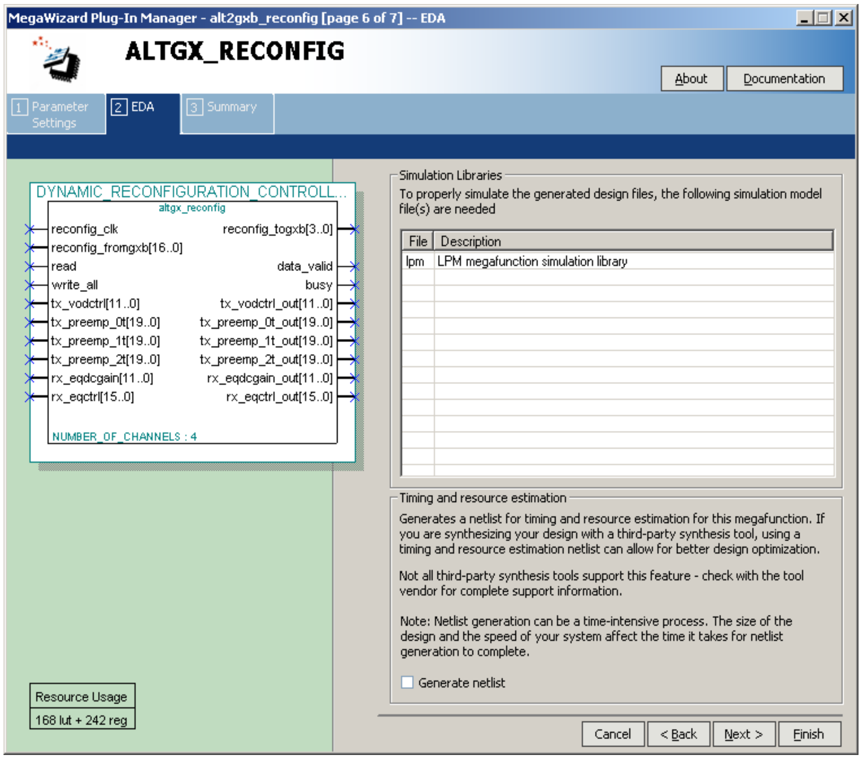
Table 4–3. MegaWizard Plug-In Manager Options (Page 5) (Part 2 of 2)

ALTGX_RECONFIG Setting	Description	Reference
Enable self recovery	When you select this option, the ALTGX_RECONFIG MegaWizard provides the `error` output port. The dynamic reconfiguration controller quits an operation if it did not complete within the expected number of clock cycles. After recovering from the illegal operation, the dynamic reconfiguration controller de-asserts the `busy` signal and asserts the `error` output port for two `reconfig_clk` cycles.	"Error Indication in the ALTGX_RECONFIG MegaWizard Plug-In Manager" section of the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*.
Use the `rx_tx_duplex_sel` port to enable RX only, TX only or duplex configuration.	You can read or write the receiver and transmitter settings, or only the receiver settings, or only the transmitter settings based on what value you set at the `rx_tx_duplex_sel` port. ■ 00 - Duplex mode ■ 01 - RX only mode ■ 10 - TX only mode ■ 11 - unsupported value (do not use this value) If you disable the `rx_tx_duplex_sel` port, the dynamic reconfiguration controller will read or write both the receiver and transmitter settings.	"Dynamically Reconfiguring PMA Controls" section of the *Stratix IV Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*.

Figure 4–6 shows page 6 (the Simulation Libraries page) of the MegaWizard Plug-In Manager for the Dynamic Reconfiguration selection.

Click **Next**.

Figure 4–6. MegaWizard Plug-In Manager—ALTGX_RECONFIG (Simulation Libraries)



Simulation Libraries

To properly simulate the generated design files, the following simulation model file(s) are needed

File	Description
lpm	LPM megafunction simulation library

Timing and resource estimation

Generates a netlist for timing and resource estimation for this megafunction. If you are synthesizing your design with a third-party synthesis tool, using a timing and resource estimation netlist can allow for better design optimization.

Not all third-party synthesis tools support this feature - check with the tool vendor for complete support information.

Note: Netlist generation can be a time-intensive process. The size of the design and the speed of your system affect the time it takes for netlist generation to complete.

□ Generate netlist

Table 4–4 describes the available option on page 6 of the MegaWizard Plug-In Manager for your ALTGX_RECONFIG custom Megafunction variation.

Make your selections on page 6, then click **Next**.

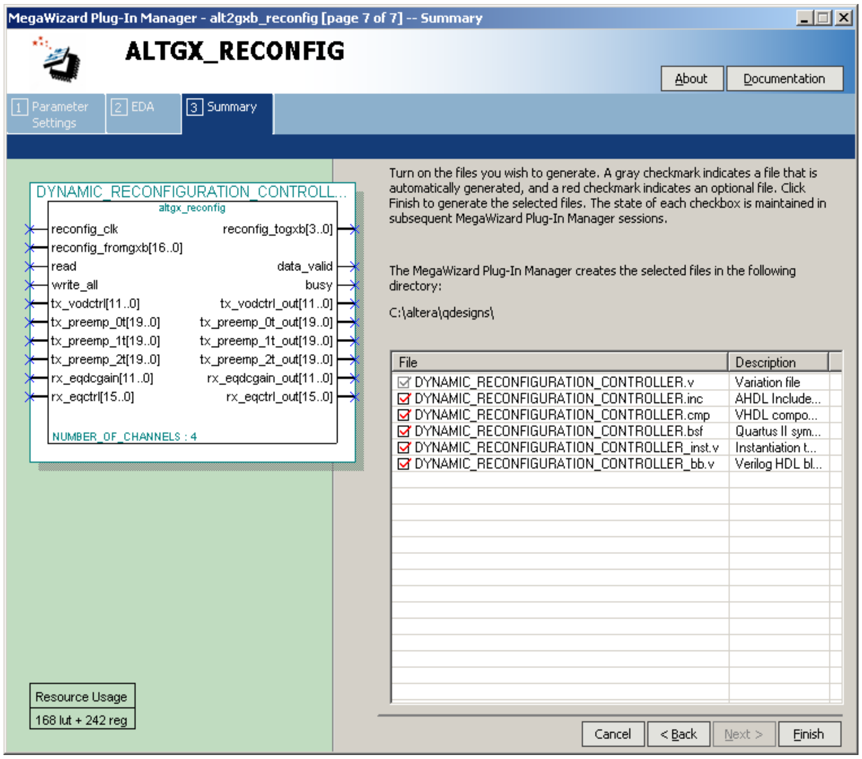
Table 4–4. MegaWizard Plug-In Manager Options (Page 6)

ALTGX_RECONFIG Setting	Description
Generate a netlist for synthesis area and timing estimation	Selecting this option generates a netlist file that third-party synthesis tools can use to estimate the timing and resource usage

Figure 4–7 shows page 7 (the last page) of the MegaWizard Plug-In Manager for the Dynamic Reconfiguration protocol set up. You can select optional files on this page.

After you make your selections, click **Finish** to generate the files.

Figure 4–7. MegaWizard Plug-In Manager—ALTGX_RECONFIG (Summary)



Turn on the files you wish to generate. A gray checkmark indicates a file that is automatically generated, and a red checkmark indicates an optional file. Click Finish to generate the selected files. The state of each checkbox is maintained in subsequent MegaWizard Plug-In Manager sessions.

The MegaWizard Plug-In Manager creates the selected files in the following directory:

C:\altera\qdesigns\

File	Description
☑ DYNAMIC_RECONFIGURATION_CONTROLLER.v	Variation file
☑ DYNAMIC_RECONFIGURATION_CONTROLLER.inc	AHDL Include...
☑ DYNAMIC_RECONFIGURATION_CONTROLLER.cmp	VHDL compo...
☑ DYNAMIC_RECONFIGURATION_CONTROLLER.bsf	Quartus II sym...
☑ DYNAMIC_RECONFIGURATION_CONTROLLER_inst.v	Instantiation t...
☑ DYNAMIC_RECONFIGURATION_CONTROLLER_bb.v	Verilog HDL bl...

Referenced Documents

This chapter references the following document:

- *Dynamic Reconfiguration* chapter in volume 2 of the *Stratix IV Device Handbook*

Document Revision History

Table 4–5 shows the revision history for this chapter.

Table 4–5. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008, v1.0	Added chapter to the Stratix IV Device Handbook	—

Stratix IV Device Handbook, Volume 4

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

SIV5V4-2.1

About this Handbook

This handbook provides comprehensive information about the Altera® Stratix® IV family of devices.

How to Contact Altera

For the most up-to-date information about Altera products, see the following table.

Contact (Note 1)	Contact Method	Address
Technical support	Website	www.altera.com/support
Technical training	Website	www.altera.com/training
	Email	custrain@altera.com
Altera literature services	Email	literature@altera.com
Non-technical support (General)	Email	nacomp@altera.com
(Software Licensing)	Email	authorization@altera.com

Note:

(1) You can also contact your local Altera sales office or sales representative.

Typographic Conventions

The following table shows the typographic conventions that this document uses.

Visual Cue	Meaning
Bold Type with Initial Capital Letters	Indicates command names, dialog box titles, dialog box options, and other GUI labels. For example, **Save As** dialog box. For GUI elements, capitalization matches the GUI.
bold type	Indicates directory names, project names, disk drive names, file names, file name extensions, dialog box options, software utility names, and other GUI labels. For example, **\qdesigns** directory, **d:** drive, and **chiptrip.gdf** file.
Italic Type with Initial Capital Letters	Indicates document titles. For example, *AN 519: Stratix IV Design Guidelines.*
Italic type	Indicates variables. For example, *n* + 1. Variable names are enclosed in angle brackets (< >). For example, <*file name*> and <*project name*>**.pof** file.
Initial Capital Letters	Indicates keyboard keys and menu names. For example, Delete key and the Options menu.
“Subheading Title”	Quotation marks indicate references to sections within a document and titles of Quartus II Help topics. For example, “Typographic Conventions.”

Visual Cue	Meaning
`Courier type`	Indicates signal, port, register, bit, block, and primitive names. For example, `data1`, `tdi`, and `input`. Active-low signals are denoted by suffix `n`. For example, `resetn`. Indicates command line commands and anything that must be typed exactly as it appears. For example, `c:\qdesigns\tutorial\chiptrip.gdf`. Also indicates sections of an actual file, such as a Report File, references to parts of files (for example, the AHDL keyword `SUBDESIGN`), and logic function names (for example, `TRI`).
1., 2., 3., and a., b., c., and so on.	Numbered steps indicate a list of items when the sequence of the items is important, such as the steps listed in a procedure.
■ ■	Bullets indicate a list of items when the sequence of the items is not important.
☞	The hand points to information that requires special attention.
⚠ CAUTION	A caution calls attention to a condition or possible situation that can damage or destroy the product or your work.
⚠ WARNING	A warning calls attention to a condition or possible situation that can cause you injury.
↵	The angled arrow instructs you to press **Enter**.
👣	The feet direct you to more information about a particular topic.

Contents

Chapter Revision Dates v

Additional Information

About this Handbook . . . . Info-vii
How to Contact Altera . . . . Info-vii
Typographic Conventions . . . . Info-vii

Section I. Device Datasheet

Revision History I-1

Chapter 1. DC and Switching Characteristics

Electrical Characteristics 1-1
Operating Conditions 1-1
Absolute Maximum Ratings 1-1
Recommended Operating Conditions 1-3
DC Characteristics 1-4
I/O Standard Specifications 1-7
Power Consumption 1-10
Switching Characteristics 1-10
Transceiver Performance Specifications 1-10
Core Performance Specifications 1-23
Clock Tree Specifications 1-23
PLL Specifications 1-24
DSP Block Specifications 1-25
TriMatrix Memory Block Specifications 1-25
Configuration and JTAG Specifications 1-26
Temperature Sensing Diode Specifications 1-27
Periphery Performance 1-27
High-Speed I/O Specification 1-27
External Memory Interface Specifications 1-28
OCT Calibration Block Specifications 1-32
Duty Cycle Distortion (DCD) Specifications 1-32
I/O Timing 1-33
Timing Model 1-33
Preliminary and Final Timing 1-33
I/O Timing Measurement Methodology 1-33
I/O Default Capacitive Loading 1-37
Programmable IOE Delay 1-38
Programmable Output Buffer Delay 1-38
Glossary 1-38
Documents Referenced 1-42
Document Revision History 1-43

Chapter Revision Dates

The chapters in this book, *Stratix IV Device Handbook, Volume 4*, were revised on the following dates. Where chapters or groups of chapters are available separately, part numbers are listed.

Chapter 1	DC and Switching Characteristics	
	Revised:	*December 2008*
	Part Number:	*SIV54001-2.1*

Section I. Stratix IV Device Datasheet

This section includes the following chapters:

- Chapter 1, DC and Switching Characteristics

Revision History

Refer to each chapter for its own specific revision history. For information on when each chapter was updated, refer to the Chapter Revision Dates section, which appears in the full handbook.

1. DC and Switching Characteristics

SIV54001-2.1

Electrical Characteristics

This chapter covers the electrical characteristics for Stratix IV devices.

Operating Conditions

When Stratix® IV devices are implemented in a system, they are rated according to a set of defined parameters. To maintain the highest possible performance and reliability of Stratix IV devices, system designers must consider the following operating requirements. Stratix IV devices are offered in both commercial and industrial grades. Commercial devices are offered in -2 (fastest), -2x, -3, and -4 speed grades.

Absolute Maximum Ratings

Absolute maximum ratings define the maximum operating conditions for Stratix IV devices. The values are based on experiments conducted with the devices and theoretical modeling of breakdown and damage mechanisms. The functional operation of the device is not implied at these conditions.

☞ Conditions beyond those listed in Table 1–1 and Table 1–2 may cause permanent damage to the device. Additionally, device operation at the absolute maximum ratings for extended periods of time may have adverse effects on the device.

Table 1–1. Stratix IV Device Absolute Maximum Ratings

Symbol	Description	Minimum	Maximum	Unit
V_{CC}	Core voltage and periphery circuitry power supply	-0.5	1.35	V
V_{CCPT}	Power supply for programmable power technology	-0.5	2.25	V
V_{CCPGM}	Configuration pins power supply	-0.5	3.75	V
V_{CCAUX}	Auxiliary supply for the programmable power technology	-0.5	3.75	V
V_{CCBAT}	Battery back-up power supply for design security volatile key register	-0.5	3.75	V
V_{CCPD}	I/O pre-driver power supply	-0.5	3.75	V
V_{CCIO}	I/O power supply	-0.5	3.9	V
V_{CC_CLKIN}	Differential clock input power supply	-0.5	3.75	V
V_{CCD_PLL}	PLL digital power supply	-0.5	1.35	V
V_{CCA_PLL}	PLL analog power supply	-0.5	3.75	V
V_I	DC input voltage	-0.5	4.0	V
T_J	Operating junction temperature	-40	100	C
T_{STG}	Storage temperature (No bias)	-65	150	C

Table 1–2. Stratix IV GX Transceiver Power Supply Absolute Maximum Ratings

Symbol	Description	Minimum	Maximum	Unit
V_{CCA_L}	Transceiver high voltage power (left side)	—	3.15 / 2.625	V
V_{CCA_R}	Transceiver high voltage power (right side)	—	3.15 / 2.625	
V_{CCHIP_L}	Transceiver HIP digital power (left side)	—	0.99	V
V_{CCHIP_R}	Transceiver HIP digital power (right side)	—	0.99	
V_{CCR_L}	Receiver power (left side)	—	1.21	V
V_{CCR_R}	Receiver power (right side)	—	1.21	
V_{CCT_L}	Transmitter power (left side)	—	1.21	V
V_{CCT_R}	Transmitter power (right side)	—	1.21	
V_{CCL_GXBLn} *(2)*	Transceiver clock power (left side)	—	1.21	V
V_{CCL_GXBRn} *(2)*	Transceiver clock power (right side)	—	1.21	V
V_{CCH_GXBLn} *(2)*	Transmitter output buffer power (left side)	—	1.54 / 1.65	V
V_{CCH_GXBRn} *(2)*	Transmitter output buffer power (right side)	—	1.54 / 1.65	V

Note to Table 1–2:

(1) n=0, 1, 2, 3

Maximum Allowed Overshoot/Undershoot Voltage

During transitions, input signals may overshoot to the voltage shown in Table 1–3 and undershoot to -2.0 V for input currents less than 100 mA and periods shorter than 20 ns.

Table 1–3 lists the maximum allowed input overshoot voltage and the duration of the overshoot voltage as a percentage of device lifetime. The maximum allowed overshoot duration is specified as a percentage of high-time over the lifetime of the device. A DC signal is equivalent to 100% duty cycle. For example, a signal that overshoots to 4.3 V can only be at 4.3 V for ~5% over the lifetime of the device; for a device lifetime of 10 years, this amounts to 5/10ths of a year.

Table 1–3. Maximum Allowed Overshoot During Transitions

Symbol	Description	Condition	Overshoot Duration as % of High Time	Unit
Vi (AC)	AC input voltage	4.0 V	100.000	%
		4.05 V	79.330	%
		4.1 V	46.270	%
		4.15 V	27.030	%
		4.2 V	15.800	%
		4.25 V	9.240	%
		4.3 V	5.410	%
		4.35 V	3.160	%
		4.4 V	1.850	%
		4.45 V	1.080	%
		4.5 V	0.630	%
		4.55 V	0.370	%
		4.6 V	0.220	%

Recommended Operating Conditions

This section lists the functional operation limits for AC and DC parameters for Stratix IV devices. The steady-state voltage and current values expected from Stratix IV devices are provided in Table 1–4. All supplies must be strictly monotonic, without plateaus.

Table 1–4. Stratix IV Device Recommended Operating Conditions (Part 1 of 2)

Symbol	Description	Condition	Minimum	Typical	Maximum	Unit
V_{CC}	Core voltage and periphery circuitry power supply	—	0.87	0.90	0.93	V
V_{CCPT}	Power supply for programmable power technology	—	1.45	1.50	1.55	V
V_{CCAUX}	Auxiliary supply for the programmable power technology	—	2.375	2.5	2.625	V
V_{CCPD}	I/O pre-driver (3.0 V) power supply	—	2.85	3	3.15	V
	I/O pre-driver (2.5 V) power supply	—	2.375	2.5	2.625	V
V_{CCIO}	I/O buffers (3.0-V) power supply	—	2.85	3	3.15	V
	I/O buffers (2.5-V) power supply	—	2.375	2.5	2.625	V
	I/O buffers (1.8-V) power supply	—	1.71	1.8	1.89	V
	I/O buffers (1.5-V) power supply	—	1.425	1.5	1.575	V
	I/O buffers (1.2-V) power supply	—	1.14	1.2	1.26	V
V_{CCPGM}	Configuration pins (3.0-V) power supply	—	2.85	3	3.15	V
	Configuration pins (2.5-V) power supply	—	2.375	2.5	2.625	V
	Configuration pins (1.8-V) power supply	—	1.71	1.8	1.89	V
V_{CCA_PLL}	PLL analog voltage regulator power supply	—	2.375	2.5	2.625	V
V_{CCD_PLL}	PLL digital voltage regulator power supply	—	0.87	0.90	0.93	V

Table 1–4. Stratix IV Device Recommended Operating Conditions (Part 2 of 2)

Symbol	Description	Condition	Minimum	Typical	Maximum	Unit
V_{CC_CLKIN}	Differential clock input power supply	—	2.375	2.5	2.625	V
V_{CCBAT}	Battery back-up power supply (For design security volatile key register)	—	1.2	3.0	3.3	V
V_I	DC input voltage	—	–0.5	—	3.6	V
V_O	Output voltage	—	0	—	V_{CCIO}	V
T_J	Operating junction temperature	Commercial	0	—	85	C
		Industrial	–40	—	100	C
t_{RAMP}	Power supply ramp time	Normal POR	0.05	—	100	ms
		Fast POR *(1)*	0.05	—	12	ms

Note to Table 1–4:

(1) If the PORSEL pin is connected to V_{CC}, all supplies must ramp up within 12 ms.

Table 1–5 shows the transceiver power supply recommended operating conditions.

Table 1–5. Stratix IV GX Transceiver Power Supply Recommended Operating Conditions

Symbol	Description	Minimum	Typical	Maximum	Unit
V_{CCA_L}	Transceiver high voltage power (left side)	2.85/2.375	3.0/2.5	3.15/2.625	V
V_{CCA_R}	Transceiver high voltage power (right side)				
V_{CCHIP_L} *(1)*	Transceiver HIP digital power (left side)	0.855	0.9	0.945	V
V_{CCHIP_R} *(1)*	Transceiver HIP digital power (right side)				
V_{CCR_L}	Receiver power (left side)	1.045	1.1	1.155	V
V_{CCR_R}	Receiver power (right side)				
V_{CCT_L}	Transmitter power (left side)	1.045	1.1	1.155	V
V_{CCT_R}	Transmitter power (right side)				
V_{CCL_GXBLn} *(2)*	Transceiver clock power (left side)	1.045	1.1	1.155	V
V_{CCL_GXBRn} *(2)*	Transceiver clock power (right side)				V
V_{CCH_GXBLn} *(2)*	Transmitter output buffer power (left side)	1.33/1.425	1.4/1.5	1.47/1.575	V
V_{CCH_GXBRn} *(2)*	Transmitter output buffer power (right side)				V

Note to Table 1–5:

(1) If V_{CCHIP_L} is connected to the same power supply source as V_{CC}, the tighter V_{CC} recommended operating conditions need to be met.

(2) n=0, 1, 2, 3

DC Characteristics

This section lists the supply current, I/O pin leakage current, input pin capacitance, on-chip termination tolerance, and hot socketing specifications.

Supply Current

Standby current is the current the device draws after the device is configured, with no inputs or outputs toggling and no activity in the device. Since these currents vary largely with resources used, use the Excel-based Early Power Estimator (EPE) to get supply current estimates for your design.

Table 1–6 lists supply current specifications for V_{CC_CLKIN}, V_{CCPGM}, and V_{CCAUX}. Use the EPE to get supply current estimates for remaining power supplies.

Table 1–6. Supply Current Specifications for V_{CC_CLKIN}, V_{CCPGM}, and V_{CCAUX}

Symbol	Parameter	Min	Max	Unit
I_{CLKIN}	V_{CC_CLKIN} current specifications	0	250	mA
I_{PGM}	V_{CCPGM} current specifications	0	250	mA
I_{AUX}	V_{CCAUX} current specification	0	250	mA

I/O Pin Leakage Current

Table 1–7 defines the Stratix IV I/O pin leakage current specifications.

Table 1–7. Stratix IV I/O Pin Leakage Current

Symbol	Description	Conditions	Min	Typ	Max	Unit
I_I	Input pin	V_I = 0V to $V_{CCIOMAX}$	-10	—	10	µA
I_{OZ}	Tri-stated I/O pin	V_O = 0V to $V_{CCIOMAX}$	-10	—	10	µA

On-Chip Termination (OCT) Specifications

Table 1–8 lists the Stratix IV series and parallel OCT calibration accuracy.

Table 1–8. On-Chip Termination With Calibration Specification for I/Os - Preliminary

			Calibration Accuracy	
Symbol	Description	Conditions	Commercial	Unit
25-Ω R_S 3.0/2.5	25-Ω series termination	V_{CCIO} = 3.0/2.5 V	± 5	%
50-Ω R_S 3.0/2.5	50-Ω series termination	V_{CCIO} = 3.0/2.5 V	± 5	%
50-Ω R_T 2.5	50-Ω parallel termination	V_{CCIO} = 2.5 V	± 10	%
25-Ω R_S 1.8	25-Ω series termination	V_{CCIO} = 1.8 V	± 5	%
50-Ω R_S 1.8	50-Ω series termination	V_{CCIO} = 1.8 V	± 5	%
50-Ω R_T 1.8	50-Ω parallel termination	V_{CCIO} = 1.8 V	± 10	%
50-Ω R_S 1.5	50-Ω series termination	V_{CCIO} = 1.5 V	± 8	%
50-Ω R_T 1.5	50-Ω parallel termination	V_{CCIO} = 1.5 V	± 10	%
50-Ω R_S 1.2	50-Ω series termination	V_{CCIO} = 1.2 V	± 8	%
50-Ω R_T 1.2	50-Ω series termination	V_{CCIO} = 1.2 V	± 10	%

The calibration accuracy for calibrated series and parallel OCTs are applicable at the moment of calibration. When PVT conditions change after calibration, the tolerance may change. Table 1–9 lists the Stratix IV OCT resistance tolerance to PVT changes.

Table 1–9. I/O On-Chip Termination Resistance Tolerance - Preliminary

Symbol	Description	Resistance Tolerance	
		Commercial	Unit
R_{OCT_UNCAL}	Internal series/parallel OCT with calibration	± 5	%
R_{OCT_CAL}	Internal series/parallel OCT without calibration	± 30	%

OCT calibration is automatically performed at power-up for OCT-enabled I/Os. Table 1–10 lists OCT variation with temperature and voltage after power-up calibration. Use Equation 1–1 to determine the OCT variation when voltage and temperature vary after power-up calibration.

Equation 1–1. OCT Variation Without Re-Calibration *(Note 1)*

$$R_{OCT} = R_{CAL}\left(1 + \frac{dR}{dT} \times \Delta T + \frac{dR}{dV} \times \Delta V\right)$$

Note to Equation 1–1:

(1) R_{CAL} is calibrated on-chip termination at power up. ΔT and ΔV are variations in temperature and voltage with respect to temperature and V_{CCIO} values, respectively, at power up.

Table 1–10. On-Chip Termination Variation after Power-Up Calibration - Preliminary

Symbol	Description	V_{CCIO} (V)	Commercial Typical	Industrial Typical	Unit
dR/dV	OCT variation with voltage without re-calibration	3.0	0.029	—	Ω/V
		2.5	0.036	—	
		1.8	0.033	—	
		1.5	0.033	—	
		1.2	0.033	—	
dR/dT	OCT variation with temperature without re-calibration	3.0	0.294	—	Ω/C
		2.5	0.301	—	
		1.8	0.355	—	
		1.5	0.344	—	
		1.2	0.348	—	

Pin Capacitance

Table 1–11 shows the Stratix IV device family pin capacitance.

Table 1–11. Stratix IV Device Capacitance *(Note 1)* - Preliminary (Part 1 of 2)

Symbol	Description	Typical	Unit
C_{IOTB}	Input capacitance on top/bottom I/O pins	8	pF
C_{IOLR}	Input capacitance on left/right I/O pins	8	pF
C_{CLKTB}	Input capacitance on top/bottom dedicated clock input pins	5	pF
C_{CLKLR}	Input capacitance on left/right dedicated clock input pins	5	pF

Table 1–11. Stratix IV Device Capacitance *(Note 1)* - Preliminary (Part 2 of 2)

Symbol	Description	Typical	Unit
C_{OUTFB}	Input capacitance on dual-purpose clock output/feedback pins	8	pF

Note to Table 1–11:

(1) Pending silicon characterization.

Hot Socketing

Table 1–12 defines the hot socketing specification for Stratix IV devices.

Table 1–12. Stratix IV Hot Socketing Specifications - Preliminary

Symbol	Description	Maximum
$I_{IIOPIN(DC)}$	DC current per I/O pin	300 µA
$I_{IOPIN(AC)}$	AC current per I/O pin	8 mA for Trise > 10 ns

I/O Standard Specifications

Table 1–13 through Table 1–18 list input voltage (V_{IH} and V_{IL}), output voltage (V_{OH} and V_{OL}), and current drive characteristics (I_{OH} and I_{OL}) for various I/O standards supported by Stratix IV devices. These tables also show the Stratix IV device family I/O standard specifications. Refer to the "Glossary" on page 1–34 for an explanation of terms used in Table 1–13 through Table 1–18. V_{OL} and V_{OH} values are valid at the corresponding I_{OH} and I_{OL}, respectively.

Table 1–13. Single-Ended I/O Standards

I/O Standard	V_{CCIO} (V)			V_{IL} (V)		V_{IH} (V)		V_{OL} (V)	V_{OH} (V)	I_{OL} (mA)	I_{OH} (mA)
	Min	Typ	Max	Min	Max	Min	Max	Max	Min		
LVTTL	2.85	3	3.15	-0.3	0.8	1.7	V_{CCIO} + 0.3	0.4	2.4	2	-2
LVCMOS	2.85	3	3.15	-0.3	0.8	1.7	V_{CCIO} + 0.3	0.2	V_{CCIO} - 0.2	0.1	-0.1
2.5 V	2.375	2.5	2.625	-0.3	0.7	1.7	V_{CCIO} + 0.3	0.2	2.1	0.1	-0.1
								0.4	2	1	-1
								0.7	1.7	2	-2
1.8 V	1.71	1.8	1.89	-0.3	0.35 * V_{CCIO}	0.65 * V_{CCIO}	V_{CCIO} + 0.3	0.45	V_{CCIO} -0.45	2	-2
1.5 V	1.425	1.5	1.575	-0.3	0.35 * V_{CCIO}	0.65 * V_{CCIO}	V_{CCIO} + 0.3	0.25 * V_{CCIO}	0.75 * V_{CCIO}	2	-2
1.2 V	1.14	1.2	1.26	-0.3	0.35 * V_{CCIO}	0.65 * V_{CCIO}	V_{CCIO} + 0.3	0.25 * V_{CCIO}	0.75 * V_{CCIO}	2	-2
3.0-V PCI	2.85	3	3.15	-	0.3 * V_{CCIO}	0.5 * V_{CCIO}	3.6	0.1 * V_{CCIO}	0.9 * V_{CCIO}	1.5	-0.5
3.0-V PCI-X	2.85	3	3.15	-	0.35 * V_{CCIO}	0.5 * V_{CCIO}	-	0.1 * V_{CCIO}	0.9 * V_{CCIO}	1.5	-0.5

Table 1–14. Single-Ended SSTL and HSTL I/O Reference Voltage Specifications

	V_{CCIO}(V)			V_{REF}(V)			V_{TT}(V)		
I/O Standard	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**
SSTL-2 Class I, II	2.375	2.5	2.625	0.49 * V_{CCIO}	0.5 * V_{CCIO}	0.51 * V_{CCIO}	V_{REF} - 0.04	V_{REF}	V_{REF} + 0.04
SSTL-18 Class I, II	1.71	1.8	1.89	0.49 * V_{CCIO}	0.5 * V_{CCIO}	0.51 * V_{CCIO}	V_{REF} - 0.04	V_{REF}	V_{REF} + 0.04
SSTL-15 Class I, II	1.425	1.5	1.575	0.49 * V_{CCIO}	0.5 * V_{CCIO}	0.51 * V_{CCIO}	V_{REF} - 0.04	V_{REF}	V_{REF} + 0.04
HSTL-18 Class I, II	1.71	1.8	1.89	0.85	0.9	0.95	—	V_{CCIO}/2	—
HSTL-15 Class I, II	1.425	1.5	1.575	0.68	0.75	0.9	—	V_{CCIO}/2	—
HSTL-12 Class I, II	1.14	1.2	1.26	0.48 * V_{CCIO}	0.5 * V_{CCIO}	0.52 * V_{CCIO}	—	V_{CCIO}/2	—

Table 1–15. Single-Ended SSTL and HSTL I/O Standards Signal Specifications

	$V_{IL(DC)}$(V)		$V_{IH(DC)}$(V)		$V_{IL(AC)}$(V)	$V_{IH(AC)}$(V)	V_{OL}(V)	V_{OH}(V)		
I/O Standard	**Min**	**Max**	**Min**	**Max**	**Max**	**Min**	**Max**	**Min**	**I_{ol} (mA)**	**I_{oh} (mA)**
SSTL-2 Class I	-0.3	V_{REF} - 0.15	V_{REF}+ 0.15	V_{CCIO} + 0.3	V_{REF} - 0.31	V_{REF}+ 0.31	V_{TT} - 0.57	V_{TT} + 0.57	8.1	-8.1
SSTL-2 Class II	-0.3	V_{REF} - 0.15	V_{REF}+ 0.15	V_{CCIO} + 0.3	V_{REF} - 0.31	V_{REF}+ 0.31	V_{TT} - 0.76	V_{TT} + 0.76	16.2	-16.2
SSTL-18 Class I	-0.3	V_{REF} - 0.125	V_{REF} + 0.125	V_{CCIO} + 0.3	V_{REF} -0.25	V_{REF} + 0.25	V_{TT} - 0.475	V_{TT} + 0.475	6.7	-6.7
SSTL-18 Class II	-0.3	V_{REF} - 0.125	V_{REF} + 0.125	V_{CCIO} + 0.3	V_{REF} -0.25	V_{REF} + 0.25	0.28	V_{CCIO} - 0.28	13.4	-13.4
SSTL-15 Class I	-0.3	V_{REF} -0.1	V_{REF} + 0.1	V_{CCIO} + 0.3	V_{REF} - 0.175	V_{REF} + 0.175	0.2 * V_{CCIO}	0.8 * V_{CCIO}	8	-8
SSTL-15 Class II	-0.3	V_{REF} -0.1	V_{REF} + 0.1	V_{CCIO} + 0.3	V_{REF} - 0.175	V_{REF} + 0.175	0.2 * V_{CCIO}	0.8 * V_{CCIO}	16	-16
HSTL-18 Class I	-0.3	V_{REF} -0.1	V_{REF} + 0.1	V_{CCIO} + 0.3	V_{REF} -0.2	V_{REF} + 0.2	0.4	V_{CCIO} -0.4	8	-8
HSTL-18 Class II	-0.3	V_{REF} -0.1	V_{REF} + 0.1	V_{CCIO} + 0.3	V_{REF} -0.2	V_{REF} + 0.2	0.4	V_{CCIO} -0.4	16	-16
HSTL-15 Class I	-0.3	V_{REF} -0.1	V_{REF} + 0.1	V_{CCIO} + 0.3	V_{REF} -0.2	V_{REF} + 0.2	0.4	V_{CCIO} -0.4	8	-8
HSTL-15 Class II	-0.3	V_{REF} -0.1	V_{REF} + 0.1	V_{CCIO} + 0.3	V_{REF} -0.2	V_{REF} + 0.2	0.4	V_{CCIO} -0.4	16	-16
HSTL-12 Class I	-0.15	V_{REF} - 0.08	V_{REF} + 0.08	V_{CCIO} + 0.15	V_{REF} -0.15	V_{REF}+ 0.15	0.25* V_{CCIO}	0.75* V_{CCIO}	8	-8
HSTL-12 Class II	-0.15	V_{REF} - 0.08	V_{REF} + 0.08	V_{CCIO} + 0.15	V_{REF} -0.15	V_{REF}+ 0.15	0.25* V_{CCIO}	0.75* V_{CCIO}	16	-16

Table 1–16. Differential SSTL I/O Standards

I/O Standard	V_{CCIO}(V)			$V_{CSWING(DC)}$(V)		$V_{X(AC)}$(V)			$V_{SWING(AC)}$(V)		$V_{OX(AC)}$(V)		
	Min	Typ	Max	Min	Max	Min	Typ	Max	Min	Max	Min	Typ	Max
SSTL-2 Class I, II	2.375	2.5	2.625	0.3	V_{CCIO} + 0.6	V_{CCIO}/2 - 0.2	—	V_{CCIO}/2 + 0.2	0.6	V_{CCIO} + 0.6	V_{CCIO}/2 - 0.15	—	V_{CCIO}/2 + 0.15
SSTL-18 Class I, II	1.71	1.8	1.89	0.3	V_{CCIO} + 0.6	V_{CCIO}/2 - 0.175	—	V_{CCIO}/2 + 0.175	0.5	V_{CCIO} + 0.6	V_{CCIO}/2 - 0.125	—	V_{CCIO}/2 + 0.125
SSTL-15 Class I, II	1.425	1.5	1.575	0.2	—	—	V_{CCIO}/2	—	0.4	—	—	V_{CCIO}/2	—

Table 1–17. Differential HSTL I/O Standards

I/O Standard	V_{CCIO}(V)			$V_{DIF(DC)}$(V)		$V_{X(AC)}$(V)			$V_{CM(DC)}$(V)			$V_{DIF(AC)}$(V)	
	Min	Typ	Max	Min	Max	Min	Typ	Max	Min	Typ	Max	Min	Max
HSTL-18 Class I	1.71	1.8	1.89	0.2	—	0.78	—	1.12	0.8	—	1.12	0.4	—
HSTL-15 Class I, II	1.425	1.5	1.575	0.2	—	0.68	—	0.9	0.7	—	0.9	0.4	—
HSTL-12 Class I, II	1.14	1.2	1.26	0.2	—	—	0.5* V_{CCIO}	—	0.4* V_{CCIO}	0.5* V_{CCIO}	0.6* V_{CCIO}	0.3	—

Table 1–18. Differential I/O Standard Specifications (Part 1 of 2) *(Note 1)*, *(2)*

I/O Standard	V_{CCIO}(V)			V_{ID}(mV)			$V_{ICM(DC)}$(V)			V_{OD}(V) *(3)*			V_{OCM}(V) *(3)*		
	Min	Typ	Max	Min	Condition	Max	Min	Condition	Max	Min	Typ	Max	Min	Typ	Max
2.5V LVDS (HIO)	2.375	2.5	2.625	100	V_{CM} = 1.25V	—	0.05	D_{max}<= 700Mbps	1.8	0.247	—	0.6	1.125	1.25	1.375
						—	1.05	D_{max} > 700Mbps	1.55	—	—	—	—	—	—
2.5V LVDS (VIO)	2.375	2.5	2.625	100	V_{CM} = 1.25V	—	0.05	D_{max}<= 700Mbps	1.8	0.247	-	0.6	1	1.25	1.5
						—	1.05	D_{max} > 700Mbps	1.55	—	—	—	—	—	1.5
RSDS (HIO)	2.375	2.5	2.625	100	V_{CM} = 1.25V	—	0.3	—	1.4	0.1	0.2	0.6	0.5	1.2	1.4
RSDS (VIO)	2.375	2.5	2.625	100	V_{CM} = 1.25V	—	0.3	—	1.4	0.1	0.2	0.6	0.5	1.2	1.5
Mini-LVDS (HIO)	2.375	2.5	2.625	200	—	600	0.4	—	1.325	0.25	—	0.6	0.5	1.2	1.4
Mini-LVDS (VIO)	2.375	2.5	2.625	200	—	600	0.4	—	1.325	0.25	—	0.6	0.5	1.2	1.5

Table 1–18. Differential I/O Standard Specifications (Part 2 of 2) *(Note 1)*, *(2)*

I/O Standard	V_{CCIO}(V)			V_{ID}(mV)			$V_{ICM(DC)}$(V)			V_{OD}(V) *(3)*			V_{OCM}(V) *(3)*		
	Min	**Typ**	**Max**	**Min**	**Condition**	**Max**	**Min**	**Condition**	**Max**	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**
LVPECL (VIO) *(4)*	2.375	2.5	2.625	300	—	—	0.6	D_{max}<= 700Mbps	1.8 *(5)*	—	—	—	—	—	—
	—	—	—	—	—	—	0.6	D_{max} > 700Mbps	1.6 *(5)*	—	—	—	—	—	—

Notes to ***Table 1–18:***

(1) VIO (vertical I/O) is top and bottom I/Os; HIO (horizontal I/O) is left and right I/Os.

(2) 1.4V/1.5V PCML transceiver I/O standard specifications are described in the section "Transceiver Performance Specifications" on page 1–10.

(3) RL range: 90 <= RL <= 110 Ω

(4) LVPECL specifications apply only to CLK input pins on column I/Os.

(5) For D_{MAX} > 700 Mbps, the minimum input voltage is 0.85 V; the maximum input voltage is 1.75 V. For F_{MAX} <=700Mbps, the minimum input voltage is 0.45 V; the maximum input voltage is 1.95 V.

Power Consumption

Altera® offers two ways to estimate power consumption for a design: the Excel-based Early Power Estimator and the Quartus® II PowerPlay Power Analyzer feature.

The interactive Excel-based Early Power Estimator is typically used prior to designing the FPGA in order to get a magnitude estimate of the device power. The Quartus II PowerPlay Power Analyzer provides better quality estimates based on the specifics of the design after place-and-route is complete. The PowerPlay Power Analyzer can apply a combination of user-entered, simulation-derived, and estimated signal activities that, combined with detailed circuit models, can yield very accurate power estimates.

For more information about power estimation tools, refer to the *PowerPlay Early Power Estimator User Guide for Stratix III and Stratix IV FPGAs* and the *PowerPlay Power Analysis* chapter in the *Quartus II Handbook*.

Switching Characteristics

This section provides performance characteristics of Stratix IV core and periphery blocks for commercial grade devices.

These characteristics can be designated as Preliminary and Final. Preliminary characteristics are created using simulation results, process data, and other known parameters. Final numbers are based on actual silicon characterization and testing. These numbers reflect the actual performance of the device under worst-case silicon process, voltage, and junction temperature conditions. The upper-right hand corner of a table shows the designation as "Preliminary" or "Final".

Transceiver Performance Specifications

This sections describes transceiver performance specifications.

Table 1–19 lists Stratix IV GX transceiver specifications.

Table 1–19. Stratix IV GX Transceiver Specification (Part 1 of 4)

Symbol/ Description	Conditions	-2 Speed Commercial Speed Grade			-3 Commercial/Industrial and -2x Commercial Speed Grade *(1)*			-4 Speed Commercial Speed Grade			Unit
		Min	**Typ**	**Max**	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**	
Reference Clock											
Input frequency from REFCLK input pins	—	50	—	637.5	50	—	637.5	50	—	622.08	MHz
Phase frequency detector (CMU PLL and receiver CDR)	—	50	—	425	50	—	325	50	—	325	MHz
Absolute V_{MAX} for a REFCLK pin	—	—	—	1.6	—	—	1.6	—	—	1.6	V
Operational V_{MAX} for a REFCLK pin	—	—	—	1.5	—	—	1.5	—	—	1.5	V
Absolute V_{MIN} for a REFCLK pin	—	-0.3	—	—	-0.3	—	—	-0.3	—	—	V
Rise/fall time	—	—	—	0.2	—	—	0.2	—	—	0.2	UI
Duty cycle	—	45	—	55	45	—	55	45	—	55	%
Peak-to-peak differential input voltage	—	200	—	1600	200	—	1600	200	—	1600	mV
Spread-spectrum modulating clock frequency	PCI Express	30	—	33	30	—	33	30	—	33	kHz
Spread-spectrum downspread	PCI Express	—	0 to -0.5%	—	—	0 to -0.5%	—	—	0 to -0.5%	—	—
On-chip termination resistors	—	—	100	—	—	100	—	—	100	—	Ω
V_{ICM} (AC coupled)	—	—	1100	—	—	1100	—	—	1100	—	mV
V_{ICM} (DC coupled)	HCSL I/O standard for PCI Express reference clock	250	—	550	250	—	550	250	—	550	mV
R_{REF}	— —	—	2000 ± 1%	—	—	2000 ± 1%	—	—	2000 ± 1%	—	Ω
Transceiver Clocks											
Calibration block clock frequency	—	10	—	125	10	—	125	10	—	125	MHz
fixedclk clock frequency	PCI Express Receiver Detect	—	125	—	—	125	—	—	125	—	MHz
reconfig_clk clock frequency	Dynamic reconfiguration clock frequency	2.5/ 37.5 *(2)*	—	50	2.5/ 37.5 *(2)*	—	50	2.5/ 37.5 *(2)*	—	50	—

Table 1–19. Stratix IV GX Transceiver Specification (Part 2 of 4)

Symbol/ Description	Conditions	-2 Speed Commercial Speed Grade			-3 Commercial/Industrial and -2x Commercial Speed Grade *(1)*			-4 Speed Commercial Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Transceiver block minimum power-down pulse width	—	—	1	—	—	1	—	—	1	—	µs
Receiver											
Data rate	—	600	—	8500	600	—	6500	600	—	5000	Mbps
Absolute V_{MAX} for a receiver pin *(3)*	—	—	—	1.6	—	—	1.6	—	—	1.6	V
Operational V_{MAX} for a receiver pin	—	—	—	1.5	—	—	1.5	—	—	1.5	V
Absolute V_{MIN} for a receiver pin	—	-0.4	—	—	-0.4	—	—	-0.4	—	—	V
Maximum peak-to-peak differential input voltage V_{ID} (diff p-p)	V_{ICM} = 0.82 V setting	—	—	2.7	—	—	2.7	—	—	2.7	V
	V_{ICM} =1 .1 V setting *(4)*	—	—	1.6	—	—	1.6	—	—	1.6	V
Minimum peak-to-peak differential input voltage V_{ID} (diff p-p)	Data Rate = 600 Mbps to 5 Gbps.	100	—	—	100	—	—	165	—	—	mV
	Data Rate > 5Gbps.	165	—	—	165	—	—	—	—	—	mV
V_{ICM}	V_{ICM} = 0.82 V setting	—	820	—	—	820	—	—	820	—	mV
	V_{ICM} =1 .1 V setting *(4)*	—	1100	—	—	1100	—	—	1100	—	mV
Differential on-chip termination resistors	85-Ω setting	—	85	—	—	85	—	—	85	—	Ω
	100-Ω setting	—	100	—	—	100	—	—	100	—	Ω
	120-Ω setting	—	120	—	—	120	—	—	120	—	Ω
	150-Ω setting	—	150	—	—	150	—	—	150	—	Ω
Return loss differential mode	PCI Express	50 MHz to 1.25 GHz: -10dB									
	XAUI	100 MHz to 2.5 GHz: -10dB									
	(OIF) CEI	100 MHz to 4.875 GHz: -8dB 4.875 GHz to 10 GHz: 16.6 dB/decade slope									
Return loss common mode	PCI Express	50 MHz to 1.25 GHz: -6dB									
	XAUI	100 MHz to 2.5 GHz: -6dB									
	(OIF) CEI	100 MHz to 4.875 GHz: -6dB 4.875 GHz to 10 GHz: 16.6 dB/decade slope									

Table 1–19. Stratix IV GX Transceiver Specification (Part 3 of 4)

Symbol/ Description	Conditions	-2 Speed Commercial Speed Grade			-3 Commercial/Industrial and -2x Commercial Speed Grade *(1)*			-4 Speed Commercial Speed Grade			Unit
		Min	**Typ**	**Max**	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**	
Programmable PPM detector *(5)*	—	± 62.5, 100, 125, 200, 250, 300, 500, 1000									ppm
Run length	—	—	80	—	—	80	—	—	80	—	UI
Programmable equalization	—	—	—	16	—	—	16	—	—	16	dB
Signal detect/loss threshold	PCI Express (PIPE) Mode	65	—	175	65	—	175	65	—	175	mV
CDR LTR time *(6)*	—	—	—	75	—	—	75	—	—	75	µs
CDR minimum T1b *(7)*	—	15	—		15	—	—	15	—	—	µs
LTD lock time *(8)*	—	0	100	4000	0	100	4000	0	100	4000	ns
Data lock time from `rx_freqlocked` *(9)*	—	—	—	4000	—	—	4000	—	—	4000	ns
Programmable DC gain	DC Gain Setting = 0	—	0	—	—	0	—	—	0	—	dB
	DC Gain Setting = 1	—	3	—	—	3	—	—	3	—	dB
	DC Gain Setting = 2	—	6	—	—	6	—	—	6	—	dB
	DC Gain Setting = 3	—	9	—	—	9	—	—	9	—	dB
	DC Gain Setting = 4	—	12	—	—	12	—	—	12	—	dB
Transmitter											
Data rate	—	600	—	8500	600	—	6500	600	—	5000	Mbps
V_{OCM}	0.65 V setting —	—	650	—	—	650	—	—	650	—	mV
Differential on-chip termination resistors	85-Ω setting	—	85	—	—	85	—	—	85	—	Ω
	100-Ω setting	—	100	—	—	100	—	—	100	—	Ω
	120-Ω setting	—	120	—	—	120	—	—	120	—	Ω
	150-Ω setting	—	150	—	—	150	—	—	150	—	Ω
Return loss differential mode	PCI Express	50 MHz to 1.25 GHz: -10dB									
	XAUI	312 MHz to 625 MHz: -10dB 625 MHz to 3.125 GHz: -10dB/decade slope									
	(OIF) CEI	100 MHz to 4.875 GHz: -8dB 4.875 GHz to 10 GHz: 16.6 dB/decade slope									

Table 1–19. Stratix IV GX Transceiver Specification (Part 4 of 4)

Symbol/ Description	Conditions	-2 Speed Commercial Speed Grade			-3 Commercial/Industrial and -2x Commercial Speed Grade *(1)*			-4 Speed Commercial Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Return loss common mode	PCI Express	50 MHz to 1.25 GHz: -6dB									
	(OIF) CEI	100 MHz to 4.875 GHz: -6dB 4.875 GHz to 10 GHz: 16.6 dB/decade slope									
Rise time	—	50	—	200	50	—	200	50	—	200	ps
Fall time	—	50	—	200	50	—	200	50	—	200	ps
Intra differential pair skew	—	—	—	15	—	—	15	—	—	15	ps
Intra-transceiver block skew	—	—	—	120	—	—	120	—	—	120	ps
Inter-transceiver block skew	—	—	—	300	—	—	300	—	—	300	ps
CMU PLL0 and CMU PLL1											
CMU PLL lock time from `CMUPLL_reset` deassertion	—	—	—	100	—	—	100	—	—	100	µs
PLD-Transceiver Interface											
Interface speed	—	25	—	250	25	—	250	25	—	250	MHz
Digital reset pulse width	—	Minimum is 2 parallel clock cycles									—

Notes to Table 1–19:

(1) The -2x speed grade is the fastest speed grade offered in the following Stratix IV GX devices: EP4SGX70DF29, EP4SGX110DF29, EP4SGX110FF35, EP4SGX230DF29, EP4SGX110FF35, EP4SGX230DF29, EP4SGX230FF35, EP4SGX290FF35, EP4SGX290FH29, EP4SGX360FF35, and EPSGX360FH29.

(2) The minimum `reconfig_clk` frequency is 2.5 MHz if the transceiver channel is configured in transmitter only mode. The minimum `reconfig_clk` frequency is 37.5MHz if the transceiver channel is configured in receiver only or receiver and transmitter mode. For more details, refer to the *Stratix IV Dynamic Reconfiguration* chapter in volume 1 of the *Stratix IV Device Handbook*.

(3) The device cannot tolerate prolonged operation at this absolute maximum.

(4) The 1.1-V RX V_{ICM} setting must be used if the input serial data standard is LVDS and the link is DC coupled.

(5) The rate matcher supports only up to +/-300 ppm.

(6) Time taken to `rx_pll_locked` goes high from `rx_analogreset` deassertion. Refer to Figure 1–1.

(7) Time for which the CDR must be kept in lock-to-reference mode after `rx_pll_locked` goes high and before `rx_locktodata` is asserted in manual mode. Refer to Figure 1–1.

(8) Time taken to recover valid data after the `rx_locktodata` signal is asserted in manual mode. Refer to Figure 1–1.

(9) Time taken to recover valid data after the `rx_freqlocked` signal goes high in automatic mode. Refer to Figure 1–2.

Figure 1–1 shows the lock time parameters in manual mode. Figure 1–2 shows the lock time parameters in automatic mode.

☞ LTD = Lock-To-Data LTR = Lock-To-Reference

Figure 1–1. Lock Time Parameters for Manual Mode

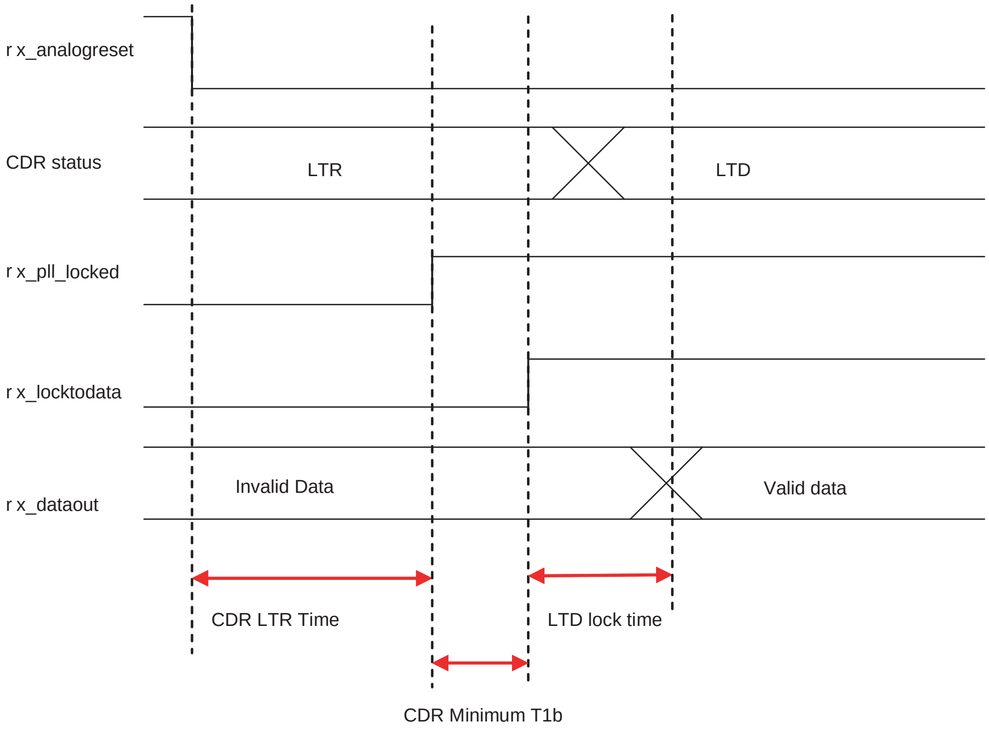


Figure 1–2. Lock Time Parameters for Automatic Mode

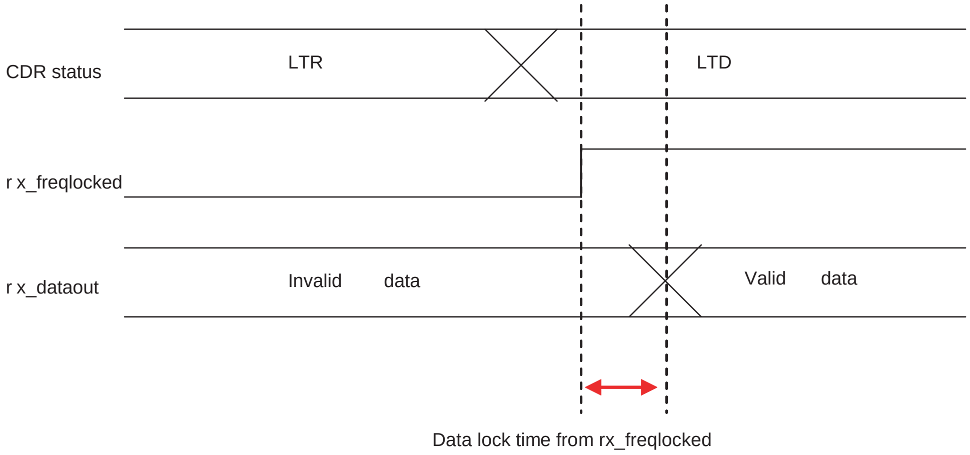


Table 1–20 through Table 1–23 show the typical V_{OD} for various differential termination settings.

Table 1–20. Typical V_{OD} Setting, TX Term = 85 Ω

Symbol	V_{OD} Setting (mV)							
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
V_{OD} Typical (mV)	170	340	510	595	680	765	850	1020

Table 1–21. Typical V_{OD} Setting, TX Term = 100 W

Symbol	V_{OD} Setting (mV)							
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
V_{OD} Typical (mV)	200	400	600	700	800	900	1000	1200

Table 1–22. Typical V_{OD} Setting, TX Term = 120 Ω

Symbol	V_{OD} Setting (mV)						
	0	**1**	**2**	**3**	**4**	**5**	**6**
V_{OD} Typical (mV)	240	480	720	840	960	1080	1200

Table 1–23. Typical V_{OD} Setting, TX Term = 150 Ω

Symbol	V_{OD} Setting (mV)					
	0	**1**	**2**	**3**	**4**	**5**
V_{OD} Typical (mV)	300	600	900	1050	1200	1350

Table 1–24 shows the Stratix IV GX transceiver block AC specifications.

Table 1–24. Stratix IV GX Transceiver Block AC Specification *(Note 1)*, *(2)* (Part 1 of 8)

Symbol/ Description	Conditions	-2 Speed Commercial Speed Grade			-3 Speed Commercial and Industrial Speed Grade			-4 Speed Commercial Speed Grade			Unit
		Min	**Typ**	**Max**	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**	
SONET/SDH Transmit Jitter Generation *(3)*											
Peak-to-peak jitter at 622.08 Mbps	Pattern = PRBS23	—	—	0.1	—	—	0.1	—	—	0.1	UI
RMS jitter at 622.08 Mbps	Pattern = PRBS23	—	—	0.01	—	—	0.01	—	—	0.01	UI
Peak-to-peak jitter at 2488.32 Mbps	Pattern = PRBS23	—	—	0.1	—	—	0.1	—	—	0.1	UI
RMS jitter at 2488.32 Mbps	Pattern = PRBS23	—	—	0.01	—	—	0.01	—	—	0.01	UI
SONET/SDH Receiver Jitter Tolerance *(3)*											

Table 1–24. Stratix IV GX Transceiver Block AC Specification *(Note 1)*, *(2)* (Part 2 of 8)

Symbol/ Description	**Conditions**	**-2 Speed Commercial Speed Grade**			**-3 Speed Commercial and Industrial Speed Grade**			**-4 Speed Commercial Speed Grade**			**Unit**
		Min	**Typ**	**Max**	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**	
Jitter tolerance at 622.08 Mbps	Jitter frequency = 0.03 KHz Pattern = PRBS23	> 15			> 15			> 15			UI
	Jitter frequency = 25 KHZ Pattern = PRBS23	> 1.5			> 1.5			> 1.5			UI
	Jitter frequency = 250 KHz Pattern = PRBS23	> 0.15			> 0.15			> 0.15			UI
Jitter tolerance at 2488.32 MBps	Jitter frequency = 0.06 KHz Pattern = PRBS23	> 15			> 15			> 15			UI
	Jitter frequency = 100 KHZ Pattern = PRBS23	> 1.5			> 1.5			> 1.5			UI
	Jitter frequency = 1 MHz Pattern = PRBS23	> 0.15			> 0.15			> 0.15			UI
	Jitter frequency = 10 MHz Pattern = PRBS23	> 0.15			> 0.15			> 0.15			UI
Fibre Channel Transmit Jitter Generation *(4)*, *(12)*											
Total jitter FC-1	Pattern = CRPAT	—	—	0.23	—	—	0.23	—	—	0.23	UI
Deterministic jitter FC-1	Pattern = CRPAT	—	—	0.11	—	—	0.11	—	—	0.11	UI
Total jitter FC-2	Pattern = CRPAT	—	—	0.33	—	—	0.33	—	—	0.33	UI
Deterministic jitter FC-2	Pattern = CRPAT	—	—	0.2	—	—	0.2	—	—	0.2	UI
Total jitter FC-4	Pattern = CRPAT	—	—	0.52	—	—	0.52	—	—	0.52	UI
Deterministic jitter FC-4	Pattern = CRPAT	—	—	0.33	—	—	0.33	—	—	0.33	UI
Fibre Channel Receiver Jitter Tolerance *(4)*, *(13)*											
Deterministic jitter FC-1	Pattern = CJTPAT	> 0.37			> 0.37			> 0.37			UI
Random jitter FC-1	Pattern = CJTPAT	> 0.31			> 0.31			> 0.31			UI
Sinusoidal jitter FC-1	Fc/25000	> 1.5			> 1.5			> 1.5			UI
	Fc/1667	> 0.1			> 0.1			> 0.1			UI
Deterministic jitter FC-2	Pattern = CJTPAT	> 0.33			> 0.33			> 0.33			UI
Random jitter FC-2	Pattern = CJTPAT	> 0.29			> 0.29			> 0.29			UI
Sinusoidal jitter FC-2	Fc/25000	> 1.5			> 1.5			> 1.5			UI
	Fc/1667	> 0.1			> 0.1			> 0.1			UI

Table 1–24. Stratix IV GX Transceiver Block AC Specification *(Note 1)*, *(2)* (Part 3 of 8)

Symbol/ Description	Conditions	-2 Speed Commercial Speed Grade			-3 Speed Commercial and Industrial Speed Grade			-4 Speed Commercial Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Deterministic jitter FC-4	Pattern = CJTPAT	> 0.33			> 0.33			> 0.33			UI
Random jitter FC-4	Pattern = CJTPAT	> 0.29			> 0.29			> 0.29			UI
Sinusoidal jitter FC-4	Fc/25000	> 1.5			> 1.5			> 1.5			UI
	Fc/1667	> 0.1			> 0.1			> 0.1			UI
XAUI Transmit Jitter Generation *(5)*											
Total jitter at 3.125 Gbps	Pattern = CJPAT	—	—	0.3	—	—	0.3	—	—	0.3	UI
Deterministic jitter at 3.125 Gbps	Pattern = CJPAT	—	—	0.17	—	—	0.17	—	—	0.17	UI
XAUI Receiver Jitter Tolerance *(5)*											
Total jitter	—	> 0.65			> 0.65			> 0.65			UI
Deterministic jitter	—	> 0.37			> 0.37			> 0.37			UI
Peak-to-peak jitter	Jitter frequency = 22.1 KHz	> 8.5			> 8.5			> 8.5			UI
Peak-to-peak jitter	Jitter frequency = 1.875 MHz	> 0.1			> 0.1			> 0.1			UI
Peak-to-peak jitter	Jitter frequency = 20 MHz	> 0.1			> 0.1			> 0.1			UI
PCI Express Transmit Jitter Generation *(6)*											
Total jitter at 2.5 Gbps (Gen1)	Compliance pattern	—	—	0.25	—	—	0.25	—	—	0.25	UI
Total jitter at 5 Gbps (Gen2)	Compliance pattern	—	—	—	—	—	—	—	—	—	UI
PCI Express Receiver Jitter Tolerance *(6)*											
Total jitter at 2.5 Gbps (Gen1)	Compliance pattern	> 0.6			> 0.6			> 0.6			UI
Total jitter at 2.5 Gbps (Gen2)	Compliance pattern	—	—	—	—	—	—	—	—	—	UI
Serial RapidIO Transmit Jitter Generation *(7)*											
Deterministic jitter (peak-to-peak)	Data Rate = 1.25, 2.5, 3.125 Gbps Pattern = CJPAT	—	—	0.17	—	—	0.17	—	—	0.17	UI
Total jitter (peak-to-peak)	Data Rate = 1.25, 2.5, 3.125 Gbps Pattern = CJPAT	—	—	0.35	—	—	0.35	—	—	0.35	UI
Serial RapidIO Receiver Jitter Tolerance *(7)*											
Deterministic jitter tolerance (peak-to-peak)	Data Rate = 1.25, 2.5, 3.125 Gbps Pattern = CJPAT	> 0.37			> 0.37			> 0.37			UI

Table 1–24. Stratix IV GX Transceiver Block AC Specification *(Note 1)*, *(2)* (Part 4 of 8)

Symbol/ Description	Conditions	-2 Speed Commercial Speed Grade			-3 Speed Commercial and Industrial Speed Grade			-4 Speed Commercial Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Combined deterministic and random jitter tolerance (peak-to-peak)	Data Rate = 1.25, 2.5, 3.125 Gbps Pattern = CJPAT	> 0.55			> 0.55			> 0.55			UI
Sinusoidal jitter tolerance (peak-to-peak)	Jitter Frequency = 22.1 KHz Data Rate = 1.25, 2.5, 3.125 Gbps Pattern = CJPAT	> 8.5			> 8.5			> 8.5			UI
	Jitter Frequency = 1.875 MHz Data Rate = 1.25, 2.5, 3.125 Gbps Pattern = CJPAT	> 0.1			> 0.1			> 0.1			UI
	Jitter Frequency = 20 MHz Data Rate = 1.25, 2.5, 3.125 Gbps Pattern = CJPAT	> 0.1			> 0.1			> 0.1			UI
GIGE Transmit Jitter Generation *(8)*											
Deterministic jitter (peak-to-peak)	Pattern = CRPAT	—	—	0.14	—	—	0.14	—	—	0.14	UI
Total jitter (peak-to-peak)	Pattern = CRPAT	—	—	0.279	—	—	0.279	—	—	0.279	UI
GIGE Receiver Jitter Tolerance *(8)*											
Deterministic jitter tolerance (peak-to-peak)	Pattern = CJPAT	> 0.4			> 0.4			> 0.4			UI
Combined deterministic and random jitter tolerance (peak-to-peak)	Pattern = CJPAT	> 0.66			> 0.66			> 0.66			UI
HiGig Transmit Jitter Generation *(9)*											
Deterministic jitter (peak-to-peak)	Data Rate = 3.75 Gbps Pattern = CJPAT	—	—	0.17	—	—	—	—	—	—	UI
Total jitter (peak-to-peak)	Data Rate = 3.75 Gbps Pattern = CJPAT	—	—	0.35	—	—	—	—	—	—	UI
HiGig Receiver Jitter Tolerance *(9)*											
Deterministic jitter tolerance (peak-to-peak)	Data Rate = 3.75 Gbps Pattern = CJPAT	> 0.37			—	—	—	—	—	—	UI
Combined deterministic and random jitter tolerance (peak-to-peak)	Data Rate = 3.75 Gbps Pattern = CJPAT	> 0.65			—	—	—	—	—	—	UI

Table 1–24. Stratix IV GX Transceiver Block AC Specification *(Note 1)*, *(2)* (Part 5 of 8)

Symbol/ Description	Conditions	-2 Speed Commercial Speed Grade			-3 Speed Commercial and Industrial Speed Grade			-4 Speed Commercial Speed Grade			Unit
		Min	**Typ**	**Max**	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**	
Sinusoidal jitter tolerance (peak-to-peak)	Jitter Frequency = 22.1 KHz Data Rate = 3.75 Gbps Pattern = CJPAT	> 8.5			—	—	—	—	—	—	UI
	Jitter Frequency = 1.875MHz Data Rate = 3.75 Gbps Pattern = CJPAT	> 0.1			—	—	—	—	—	—	UI
	Jitter Frequency = 20 MHz Data Rate = 3.75 Gbps Pattern = CJPAT	> 0.1			—	—	—	—	—	—	UI
(OIF) CEI Transmitter Jitter Generation *(10)*											
Total jitter (peak-to-peak)	Data Rate = 6.375 Gbps Pattern = PRBS15 BER = 10^{-12}	—	—	0.3	—	—	N/A	—	—	N/A	UI
(OIF) CEI Receiver Jitter Tolerance *(10)*											
Deterministic jitter tolerance (peak-to-peak)	Data Rate = 6.375 Gbps Pattern = PRBS31 BER = 10^{-12}	> 0.675			N/A	—	—	N/A	—	—	UI
Combined deterministic and random jitter tolerance (peak-to-peak)	Data Rate = 6.375 Gbps Pattern=PRBS31 BER = 10^{-12}	> 0.988			N/A	—	—	N/A	—	—	UI

Table 1–24. Stratix IV GX Transceiver Block AC Specification *(Note 1)*, *(2)* (Part 6 of 8)

Symbol/ Description	Conditions	-2 Speed Commercial Speed Grade			-3 Speed Commercial and Industrial Speed Grade			-4 Speed Commercial Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Sinusoidal jitter tolerance (peak-to-peak)	Jitter Frequency = 38.2 KHz Data Rate = 6.375 Gbps Pattern = PRBS31 BER = 10^{-12}	> 5			N/A	—	—	N/A	—	—	UI
	Jitter Frequency = 3.82 MHz Data Rate = 6.375 Gbps Pattern = PRBS31 BER = 10^{-12}	> 0.05			N/A	—	—	N/A	—	—	UI
	Jitter Frequency = 20 MHz Data Rate= 6.375 Gbps Pattern = PRBS31 BER = 10^{-12}	> 0.05			N/A	—	—	N/A	—	—	UI
SDI Transmitter Jitter Generation *(11)*											
Alignment jitter (peak-to-peak)	Data Rate = 1.485 Gbps (HD) Pattern = Color Bar Low-Frequency Roll-Off = 100 KHz	0.2	—	—	0.2	—	—	0.2	—	—	UI
	Data Rate = 2.97 Gbps (3G) Pattern = Color Bar Low-Frequency Roll-Off = 100 KHz	0.3	—	—	0.3	—	—	0.3	—	—	UI
SDI Receiver Jitter Tolerance *(11)*											

Table 1–24. Stratix IV GX Transceiver Block AC Specification *(Note 1)*, *(2)* (Part 7 of 8)

Symbol/ Description	Conditions	-2 Speed Commercial Speed Grade			-3 Speed Commercial and Industrial Speed Grade			-4 Speed Commercial Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Sinusoidal jitter tolerance (peak-to-peak)	Jitter Frequency = 15 KHz Data Rate = 2.97 Gbps (3G) Pattern = Single Line Scramble Color Bar	> 2			> 2			> 2			UI
	Jitter Frequency = 100 KHz Data Rate = 2.97 Gbps (3G) Pattern = Single Line Scramble Color Bar	> 0.3			> 0.3			> 0.3			UI
	Jitter Frequency = 148.5 MHz Data Rate = 2.97 Gbps (3G) Pattern = Single Line Scramble Color Bar	> 0.3			> 0.3			> 0.3			UI

Table 1–24. Stratix IV GX Transceiver Block AC Specification *(Note 1)*, *(2)* (Part 8 of 8)

Symbol/ Description	Conditions	-2 Speed Commercial Speed Grade			-3 Speed Commercial and Industrial Speed Grade			-4 Speed Commercial Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Sinusoidal jitter tolerance (peak-to-peak)	Jitter Frequency = 20 KHz Data Rate = 1.485 Gbps (HD) Pattern = 75% Color Bar		> 1			> 1			> 1		UI
	Jitter Frequency = 100 KHz Data Rate = 1.485 Gbps (HD) Pattern = 75% Color Bar		> 0.2			> 0.2			> 0.2		UI
	Jitter Frequency = 148.5 MHz Data Rate = 1.485 Gbps (HD) Pattern =75% Color Bar		> 0.2			> 0.2			> 0.2		UI

Notes to Table 1–24:

(1) Dedicated `refclk` pins were used to drive the input reference clocks.
(2) Jitter numbers specified are valid for the stated conditions only.
(3) The jitter numbers for SONET/SDH are compliant to the GR-253-CORE Issue 3 Specification.
(4) The jitter numbers for Fibre Channel are compliant to the FC-PI-4 Specification revision 6.10.
(5) The jitter numbers for XAUI are compliant to the IEEE802.3ae-2002 Specification.
(6) The jitter numbers for PCI Express are compliant to the PCIe Base Specification 2.0.
(7) The jitter numbers for Serial RapidIO are compliant to the RapidIO Specification 1.3.
(8) The jitter numbers for GIGE are compliant to the IEEE802.3-2002 Specification.
(9) The jitter numbers for HiGig are compliant to the IEEE802.3ae-2002 Specification.
(10) The jitter numbers for (OIF) CEI are compliant to the OIF-CEI-02.0 Specification.
(11) The HD-SDI and 3G-SDI jitter numbers are compliant to the SMPTE292M and SMPTE424M Specifications.
(12) The fibre channel transmitter jitter generation numbers are compliant to the specification at δ_T interoperability point.
(13) The fibre channel receiver jitter tolerance numbers are compliant to the specification at δ_R interoperability point.

Core Performance Specifications

This sections describes the clock tree, PLL, DSP, TriMatrix, and configuration and JTAG specifications.

Clock Tree Specifications

Table 1–25 lists the clock tree specifications for Stratix IV devices.

Table 1–25. Stratix IV Clock Tree Performance - Preliminary

Device	Performance			Unit
	-2/-2x Speed Grade	-3 Speed Grade	-4 Speed Grade	
EP4SE110	600	500	450	MHz
EP4SE230	600	500	450	MHz
EP4SE290	600	500	450	MHz
EP4SE360	600	500	450	MHz
EP4SE530	600	500	450	MHz
EP4SE680	600	500	450	MHz
EP4SGX70	600	500	450	MHz
EP4SGX110	600	500	450	MHz
EP4SGX230	600	500	450	MHz
EP4SGX290	600	500	450	MHz
EP4SGX360	600	500	450	MHz
EP4SGX530	600	500	450	MHz

PLL Specifications

Table 1–26 describes the Stratix IV PLL specifications when operating in both the commercial junction temperature range (0 to 85×C) and the industrial junction temperature range (-40 to 100×C).

Table 1–26. Stratix IV PLL Specifications - Preliminary (Part 1 of 2)

Symbol	Description	Min	Typ	Max	Unit
f_{IN}	Input clock frequency	5	—	720 *(1)*	MHz
f_{INPFD}	Input frequency to the PFD	5	—	325	MHz
f_{VCO} *(2)*	PLL VCO operating Range	600	—	1300	MHz
f_{INDUTY}	Input clock duty cycle	40	—	60	%
$f_{EINDUTY}$	External feedback clock input duty cycle	40	—	60	%
t_{INCCJ}	Input clock cycle to cycle jitter	—	—	*(4)*	ps
f_{OUT}	Output frequency for internal global or regional clock	—	—	717 *(3)*	MHz
f_{OUT_EXT}	Output frequency for external clock output	—	—	717 *(3)*	MHz
$t_{OUTDUTY}$	Duty cycle for external clock output (when set to 50%)	45	50	55	%
t_{OUTPJ_DC}	Dedicated clock output period jitter	—	—	*(4)*	ps
t_{OUTPJ_IO}	Regular I/O clock output period jitter	—	—	*(4)*	ps
t_{FCOMP}	External feedback clock compensation time	—	—	10	ns
$t_{CONFIGPLL}$	Time required to reconfigure PLL scan chains	—	*(4)*	—	SCANCLK cycles
$t_{CONFIGPHASE}$	Time required to reconfigure phase shift	1	—	1	SCANCLK cycles
$f_{SCANCLK}$	scanclk frequency	—	—	100	MHz
t_{LOCK}	Time required to lock from end of device configuration	—	—	*(4)*	ms

Table 1–26. Stratix IV PLL Specifications - Preliminary (Part 2 of 2)

Symbol	Description	Min	Typ	Max	Unit
t_{DLOCK}	Time required to lock dynamically (after switchover or reconfiguring any non-post-scale counters/delays)	—	—	*(4)*	ms
$f_{CL\ B\ W}$	PLL closed-loop low bandwidth range	—	*(4)*	—	MHz
	PLL closed-loop medium bandwidth range	—	*(4)*	—	MHz
	PLL closed-loop high bandwidth range	—	*(4)*	—	MHz
t_{PLL_PSERR}	Accuracy of PLL phase shift	—	*(4)*	—	ps
t_{ARESET}	Minimum pulse width on *areset* signal	10	—	—	ns

Notes to Table 1–26:

(1) F_{IN} is limited by I/O F_{MAX}.

(2) The VCO frequency reported by Quartus II software is after the post scale divider (k) and may be outside the VCO min and max range.

(3) This specification is limited by the lower of the two: I/O F_{MAX} or F_{OUT} of the PLL.

(4) Pending silicon characterization.

DSP Block Specifications

Table 1–27 describes the Stratix IV DSP block performance specifications.

Table 1–27. Stratix IV DSP Block Performance Specifications ***(Note 1)*** **- Preliminary**

Mode	Resources Used	Performance			Unit
	Number of Multipliers	-2/-2x Speed Grade	-3 Speed Grade	-4 Speed Grade	
9×9-bit multiplier	1	490	405	375	MHz
12×12-bit multiplier	1	490	405	375	MHz
18×18-bit multiplier	1	550	455	420	MHz
36×36-bit multiplier	1	440	365	335	MHz
18×18-bit multiply accumulator	4	490	405	375	MHz
18×18-bit multiply adder	4	490	405	375	MHz
18×18-bit multiply adder-signed full precision	2	490	405	375	MHz
18×18-bit multiply adder with loopback *(2)*	2	390	320	300	MHz
36-bit shift (32 bit data)	1	440	365	335	MHz
Double mode	1	440	365	335	MHz

Notes to Table 1–27:

(1) Maximum is for fully pipelined block with **Round** and **Saturation** disabled.

(2) Maximum is for non-pipelined block with loopback input registers disabled and **Round** and **Saturation** disabled.

TriMatrix Memory Block Specifications

Table 1–28 describes the Stratix IV TriMatrix memory block specifications.

Table 1–28. Stratix IV TriMatrix Memory Block Performance Specifications Preliminary

Memory	Mode	Resources Used		Performance			Unit
		ALUTs	TriMatrix Memory	-2 /-2x Speed Grade	-3 Speed Grade	-4 Speed Grade	
MLAB	Single port 64×10	0	1	600	500	450	MHz
	Simple dual-port 32×20 single clock	0	1	600	500	450	MHz
	Simple dual-port 64×10 single clock	0	1	600	500	450	MHz
M9K Block	Single-port 256×36	0	1	600	500	450	MHz
	Simple dual-port 256×36 single CLK	0	1	600	500	450	MHz
	True dual port 512×18 single CLK	0	1	600	500	450	MHz
M144K	Single-port 2K×72	0	1	600	500	450	MHz
	Simple dual-port 2K×72 dual CLK	0	1	600	500	450	MHz
	Simple dual-port 2K×64 dual CLK (with ECC)	0	1	333	275	250	MHz
	True dual-port 4K×36 dual CLK	0	1	600	500	450	MHz

Configuration and JTAG Specifications

Table 1–29 lists the Stratix IV configuration mode specifications.

Table 1–29. Stratix IV Configuration Mode Specifications - Preliminary

Programming Mode	DCLK Fmax	Unit
Passive serial	125	MHz
Fast passive parallel	125	MHz
Fast active serial	40	MHz
Remote update only in fast AS mode	10	MHz

Table 1–30 shows the JTAG timing parameters and values for Stratix IV devices.

Table 1–30. Stratix IV JTAG Timing Parameters and Values - Preliminary

Symbol	Description	Min	Max	Unit
t_{JCP}	TCK clock period	30	—	ns
t_{JCH}	TCK clock high time	14	—	ns
t_{JCL}	TCK clock low time	14	—	ns
$t_{JPSU\ (TDI)}$	TDI JTAG port setup time	1	—	ns
$t_{JPSU\ (TMS)}$	TMS JTAG port setup time	3	—	ns
t_{JPH}	JTAG port hold time	5	—	ns
t_{JPCO}	JTAG port clock to output	—	11 *(1)*	ns
t_{JPZX}	JTAG port high impedance to valid output	—	14 *(1)*	ns
t_{JPXZ}	JTAG port valid output to high impedance	—	14 *(1)*	ns

Note to Table 1–30:

(1) A 1 ns adder is required for each V_{CCIO} voltage step down from 3.3 V. For example, t_{JPCO} = 12 ns if V_{CCIO} of the TDO I/O bank = 2.5 V, or 13 ns if it equals 1.8 V.

Temperature Sensing Diode Specifications

Table 1–31 lists the specifications for the Stratix IV temperature sensing diode.

Table 1–31. Temperature Sensing Diode Specifications - Preliminary

Symbol	Description	Min	Max	Unit
f_{TSD_INCLK}	TSD Input Clock Frequency (without CLK divider)	0.25	1.01	MHz
	TSD Input Clock Frequency (with CLK divider)	38	42	MHz
$t_{DUTY\_TSD\_INCLK}$	Duty Cycle of TSD Input Clock	45	55	%

Periphery Performance

This section describes periphery performance including high-speed I/O, external memory interface, and OCT calibration block specifications.

High-Speed I/O Specification

Table 1–32 shows the high-speed I/O timing for Stratix IV devices.

Table 1–32. High-Speed I/O Specifications for Fastest Speed Grade - Preliminary *(Note 1), (2), (3), (4)* (Part 1 of 2)

Symbol	Conditions	-2/-2x Speed Grade			Unit
		Min	Typ	Max	
f_{IN} (input reference clock frequency) = f_{HSDR} / W	Clock boost factor, W = 1 to 40	5	—	717	MHz
f_{HSCLK} (source synchronous output clock frequency)	—	5 *(5)*	—	717	MHz
f_{HSDR} (data rate)	Serdes factor, J = 3 to 10	150	—	1600	Mbps
	Serdes factor, J = 2, Uses DDR Registers	*(6)*	—	1250	Mbps
	Serdes factor, J = 1, Uses SDR Register	*(6)*	—	717 *(7)*	Mbps
$f_{HSDRDPA}$ (DPA data rate)	Serdes factor, J = 3 to 10	150	—	1600	Mbps
Transmitter channel-to-channel skew (TCCS)	All differential standards	—	—	*(5)*	ps
Receiver sampling window (SW)	All differential standards	—	—	*(5)*	ps
$t_{OUTJITTER\_DC}$	—	—	—	*(5)*	ps
$t_{OUTJITTER\_IO}$	—	—	—	*(5)*	ps
Output t_{RISE}	All differential I/O standards	—	—	*(5)*	ps
Output t_{FALL}	All differential I/O standards	—	—	*(5)*	ps
t_{DUTY}	Tx output clock duty cycle	45	50	55	%
DPA run length	—	—	—	*(5)*	UI

Table 1–32. High-Speed I/O Specifications for Fastest Speed Grade - Preliminary *(Note 1), (2), (3), (4)* (Part 2 of 2)

		-2/-2x Speed Grade			
Symbol	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
DPA jitter tolerance	Data channel peak-to-peak jitter tolerance	*(5)*	—	—	UI

Notes to Table 1–32:

(1) When J = 4 to 10, the SERDES block is used.

(2) When J = 1 or 2, the SERDES block is bypassed.

(3) The input clock frequency and the W factor must satisfy the following Left/Right PLL output specification:
150 <= input clock frequency × W <= 1600 MHz.

(4) Specifications for -3 and -4 speed grades will be available after silicon characterization.

(5) Pending silicon characterization.

(6) The minimum specification is dependent on the clock source (for example, PLL or clock pin) and the clock routing resource (global, regional, or local) utilized. The I/O differential buffer and input register does not have a minimum toggle rate.

(7) Same as device clock tree F_{MAX}.

Table 1–33 shows the DPA lock time specifications for Stratix IV devices.

Table 1–33. DPA Lock Time Specifications - Preliminary

Standard	Training Pattern	Transition Density	Min	Unit
SPI-4	00000000001111111111	10%	*(1)*	Number of repetitions
Parallel Rapid I/O	00001111	25%	*(1)*	Number of repetitions
	10010000	50%	*(1)*	Number of repetitions
Miscellaneous	10101010	100%	*(1)*	Number of repetitions
	01010101	—	*(1)*	Number of repetitions

Note to Table 1–33:

(1) Pending silicon characterization.

External Memory Interface Specifications

Table 1–34 through Table 1–43 list the external memory interface specifications for the Stratix IV device family. Use these tables for memory interface timing analysis.

Table 1–34. Stratix IV Maximum Clock Rate Support for External Memory Interfaces with Half-Rate Controller *(Note 1), (2)* (Part 1 of 2)

	Stratix IV GX Devices with 1152-Pin (with 24 Transceivers), 1517-Pin, and 1932-Pin Packages						Stratix IV GX Devices with 780-Pin and 1152-Pin (with 16 Transceivers) Packages					
	–2 Speed Grade (MHz)		–3 Speed Grade (MHz)		–4 Speed Grade (MHz)		-2x Speed Grade (MHz)		-3 Speed Grade (MHz)		-4 Speed Grade (MHz)	
Memory Standards	**Column I/O Banks**	**Row I/O Banks** *(3)*	**Column I/O Banks**	**Row I/O Banks** *(3)*	**Column I/O Banks**	**Row I/O Banks** *(3)*	**Column I/O Banks**	**Row I/O Banks** *(3)*	**Column I/O Banks**	**Row I/O Banks** *(3)*	**Column I/O Banks**	**Row I/O Banks** *(3)*
DDR3 SDRAM *(4)*	533	333	400	333	333	333	333	333	333	333	333	333
DDR2 SDRAM *(4)*	400	333	333	333	333	333	333	333	333	333	333	333
DDR SDRAM *(4)*	200	200	200	200	200	200	200	200	200	200	200	200

Table 1–34. Stratix IV Maximum Clock Rate Support for External Memory Interfaces with Half-Rate Controller *(Note 1)*, *(2)* (Part 2 of 2)

Memory Standards	Stratix IV GX Devices with 1152-Pin (with 24 Transceivers), 1517-Pin, and 1932-Pin Packages						Stratix IV GX Devices with 780-Pin and 1152-Pin (with 16 Transceivers) Packages					
	–2 Speed Grade (MHz)		–3 Speed Grade (MHz)		–4 Speed Grade (MHz)		-2x Speed Grade (MHz)		-3 Speed Grade (MHz)		-4 Speed Grade (MHz)	
	Column I/O Banks	Row I/O Banks *(3)*	Column I/O Banks	Row I/O Banks *(3)*	Column I/O Banks	Row I/O Banks *(3)*	Column I/O Banks	Row I/O Banks *(3)*	Column I/O Banks	Row I/O Banks *(3)*	Column I/O Banks	Row I/O Banks *(3)*
QDRII+SRAM (2.5 clock cycle latency only) *(5)*, *(6)*	400	300	350	300	300	300	300	300	300	300	300	300
QDRII SRAM (1.5-V and 1.8-V HSTL) *(6)*	350	300	300	300	300	300	300	300	300	300	300	300
RLDRAM II (1.5-V and 1.8-V HSTL)	400	333	333	333	333	333	333	333	333	333	333	333

Notes to Table 1–34:

(1) Numbers are preliminary pending characterization. The supported operating frequencies listed here are memory interface maximums for the FPGA device family. Your design's actual achievable performance is based on design and system-specific factors, as well as static timing analysis of the completed design.

(2) Column I/Os refer to top and bottom I/Os. Row I/Os refer to left and right I/Os.

(3) The row I/O banks do not support 1.5-V HSTL and SSTL Class II I/O standards.

(4) This applies for interfaces with both modules and components.

(5) The QDRII+ SRAM devices with 2.0 clock cycle latency are not supported due to hardware limitations.

(6) Stratix IV devices in the 780- and 1152-pin packages support ×36 QDRII+/QDRII SRAM at a lower maximum frequency as detailed in the *External Memory Interfaces in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook*.

Table 1–35. Stratix IV Maximum Clock Rate Support for External Memory Interfaces with Full-Rate Controller *(Note 1)*, *(2)*, *(3)*

Memory Standards	-2/-2x Speed Grade (MHz)		-3 Speed Grade (MHz)		-4 Speed Grade (MHz)	
	Column I/O Banks	Row I/O Banks *(4)*	Column I/O Banks	Row I/O Banks *(4)*	Column I/O Banks	Row I/O Banks *(4)*
DDR2 SDRAM	267	267	233	233	200	200
DDR SDRAM	200	200	200	200	200	200

Notes to Table 1–35:

(1) Numbers are preliminary until characterization is final. The supported operating frequencies listed here are memory interface maximums for the FPGA device family. Your design's actual achievable performance is based on design and system-specific factors, as well as static timing analysis of the completed design.

(2) Column I/Os refer to top and bottom I/Os. Row I/Os refer to left and right I/Os.

(3) This applies for interfaces with both modules and components.

(4) The row I/O banks do not support 1.5 V HSTL and SSTL Class II I/O standards.

Table 1–36. Stratix IV Maximum Clock Rate Support with the ×36 Mode Emulation *(Note 1)*, *(2)*, *(3)*

Memory Standards	-2/-2x Speed Grade (MHz)		-3/-3x Speed Grade (MHz)		–4 Speed Grade (MHz)	
	Column I/O Banks	Row I/O Banks *(4)*	Column I/O Banks	Row I/O Banks *(4)*	Column I/O Banks	Row I/O Banks *(4)*
QDRII+SRAM (2.5 clock cycle latency only) *(4)*	300	250	250	167	250	167
QDRII SRAM (1.5-V and 1.8-V HSTL)	300	250	250	167	250	167

Notes to Table 1–36:

(1) Numbers, based on using the half-rate controller, are preliminary until characterization is final. The supported operating frequencies listed here are memory interface maximums for the FPGA device family. Your design's actual achievable performance is based on design and system-specific factors as well as static timing analysis of the completed design.

(2) The performance listed in this table is lower than the performance listed in Table 1–34 due to double loading of the CQ/CQn pins. Double loading causes degradation in the signal slew rate which affects FPGA delay. Furthermore, due to the difference in slew rate, there is a shift in the setup and hold time window. You can perform an IBIS simulation to illustrate the shift in the clock signals.

(3) Column I/Os refer to top and bottom I/Os. Row I/Os refer to left and right I/Os.

(4) The QDRII+ SRAM devices with 2.0 clock cycle latency are not supported due to hardware limitations.

External Memory I/O Timing Specifications

Table 1–37 and Table 1–38 list Stratix IV device timing uncertainties on the read and write data paths. Use these specifications to determine timing margins for source synchronous paths between a Stratix IV FPGA and an external memory device.

Table 1–37. Sampling Window (SW) - Read Side - Preliminary

Memory Type	I/O Standard	Width	Sampling window (ps)					
			-2/-2x Speed Grade		-3 Speed Grade		-4 Speed Grade	
			Setup	Hold	Setup	Hold	Setup	Hold
DDR3	1.5 SSTL	×4	250	250	300	300	374	374
		×8	250	250	300	300	374	374
DDR2 Differential	1.8 V SSTL	×4	181	306	234	326	257	326
		×8	181	306	234	326	257	326
DDR2 SEIO	1.8 V SSTL	×4	231	256	284	276	307	276
		×8	231	256	284	276	307	276
DDR1 SEIO	2.5 V SSTL	×4	231	256	284	261	307	261
		×8	231	256	284	261	307	261
QDRII/II+	1.5 V HSTL	×9	231	256	284	261	307	261
		×18	261	286	314	291	337	291
		×36	261	286	314	291	337	291
QDRII/II+ Emulation	1.5 V HSTL	×36	261	328	314	337	337	350
QDRII	1.8 V HSTL	×9	231	256	284	261	307	261
		×18	261	286	314	291	337	291
		×36	261	286	314	291	337	291

Table 1–37. Sampling Window (SW) - Read Side - Preliminary

Memory Type	I/O Standard	Width	Sampling window (ps)					
			-2/-2x Speed Grade		-3 Speed Grade		-4 Speed Grade	
			Setup	Hold	Setup	Hold	Setup	Hold
RLDRAM II	1.5 V HSTL	×9	181	306	234	326	257	326
		×18	211	336	264	356	287	356
		×9	181	306	234	326	257	326
		×18	211	336	264	356	287	356

Table 1–38. Transmitter Channel-to-Channel Skew (TCCS) - Write Side - Preliminary

Memory Type	I/O Standard	Width	TCCS (ps)					
			-2/-2x Speed Grade		-3 Speed Grade		-4 Speed Grade	
			Lead	Lag	Lead	Lag	Lead	Lag
DDR3	1.5 SSTL	×4	260	260	290	290	310	310
		×8	260	260	290	290	310	310
DDR2 Differential	1.8 V SSTL	×4	229	246	230	355	250	388
		×8	229	246	230	355	250	388
DDR2 SEIO	1.8 V SSTL	×4	316	168	318	239	346	260
		×8	316	168	318	239	346	260
DDR1 SEIO	2.5 V SSTL	×4	313	157	315	222	343	242
		×8	313	157	315	222	343	242
QDRII/II+	1.5 V HSTL	×9	260	248	262	358	285	391
		×18	290	278	292	388	315	421
		×36	290	278	292	388	315	421
QDRII/II+ Emulation	1.5 V HSTL	×36	310	298	312	408	335	441
QDRII	1.8 V HSTL	×9	229	246	230	355	250	388
		×18	259	276	260	385	280	418
		×36	259	276	260	385	280	418
RLDRAM II	1.5	×9	260	248	262	358	285	391
		×18	290	278	292	388	315	421
		×9	229	246	230	355	250	388
		×18	259	276	260	385	280	418

DLL and DQS Logic Block Specifications

Table 1–39 describes the DLL frequency range specifications for Stratix IV devices.

Table 1–39. Stratix IV DLL Frequency Range Specifications - Preliminary

Frequency Mode	Frequency Range (MHz)			Resolution
	-2/-2x Speed Grade	-3 Speed Grade	-4 Speed Grade	
0	90 - 150	90 - 140	90 - 120	22.5
1	120 - 200	120 - 190	120 - 170	30
2	150 - 240	150 - 230	150 - 200	36
3	180 - 300	180 - 290	180 - 250	45
4	240 - 370	240 - 350	240 - 310	30
5	290 - 450	290 - 420	290 - 370	36
6	360 - 560	360 - 530	360 - 460	45

Table 1–40 describes the DQS phase offset delay per stage for Stratix IV devices.

Table 1–40. DQS Phase Offset Delay Per Setting *(Note 1), (2), (3)*

Speed Grade	Min	Max	Unit
-2/-2x	7	13	ps
-3	8	14	ps
-4	8.5	15.5	ps

Notes to Table 1–40:

(1) The valid settings for phase offset are -64 to +63 for frequency mode 0 to 3 and -32 to +31 for frequency modes 4 to 6.

(2) The typical value equals the average of the minimum and maximum values.

(3) The delay settings are linear, with a cumulative delay variation of 40 ps for all speed grades. For example, when using a -2 speed grade and applying a 10 phase offset settings to a 90° phase shift at 400 MHz, the expected average cumulative delay is [625 ps + (10 * 10.5 ps) ± 20 ps] = 730 ps ± 20 ps

OCT Calibration Block Specifications

Table 1–41 describes the OCT calibration block specifications for Stratix IV devices.

Table 1–41. OCT Calibration Block Specifications — Preliminary

Symbol	Description	Min	Typ	Max	Unit
OCTUSRCLK	Clock required by OCT calibration blocks	—	—	20	MHz
T_{OCTCAL}	Number of OCTUSRCLK clock cycles required for OCT R_S/R_T calibration	—	1000	—	Cycles
$T_{OCTSHIFT}$	Number of OCTUSRCLK clock cycles required for OCT code to shift out	—	28	—	Cycles
T_{RS_RT}	Time required to dynamically switch from R_S to R_T	—	2.5	—	ns

Duty Cycle Distortion (DCD) Specifications

Table 1–42 lists the worst case DCD for Stratix IV devices.

Table 1–42. DCD on Stratix IV I/O Pins *(Note 1)*, *(2)* — Preliminary

Symbol	-2/2x Speed Grade		-3 Speed Grade		-4 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	
Output Duty Cycle *(2)*	45	55	45	55	45	55	%

Notes to Table 1–42:

(1) Preliminary DCD specification applies to clock outputs from PLLs, global clock tree, and IOE driving dedicated and general purpose I/O pins.

(2) Detailed DCD specifications pending silicon characterization.

I/O Timing

Altera offers two ways to determine I/O timing: the Excel-based I/O timing and the Quartus II Timing Analyzer.

The Excel-based I/O timing provides pin timing performance for each device density and speed grade. The data is typically used prior to designing the FPGA to get an estimate of the timing budget as part of the link timing analysis. The Quartus II Timing Analyzer provides a more accurate and precise I/O timing data based on the specifics of the design after place-and-route is complete.

☞ The Excel-based I/O timing spreadsheet can be downloaded from the Stratix IV Device Literature webpage.

Programmable IOE Delay

Table 1–43 shows Stratix IV IOE programmable delay settings.

Table 1–43. Stratix IV IOE Programmable Delay

Parameter	Available Settings	-3 Speed Grade	
		Min Delay (ps)	Max Delay (ps)
D1	16	150	900
D2	8	330	700
D3	8	155	2581
D9	16	123	897
D10	7	118	377

Programmable Output Buffer Delay

Table 1–44 lists the delay chain settings that control the rising and falling edge delays of the output buffer. Default delay is 0 ps.

Table 1–44. Programmable Output Buffer Delay

Symbol	Parameter	Typ	Unit
D_{OUTBUF}	Rising and/or falling edge delay	0 (default)	ps
		50	ps
		100	ps
		150	ps

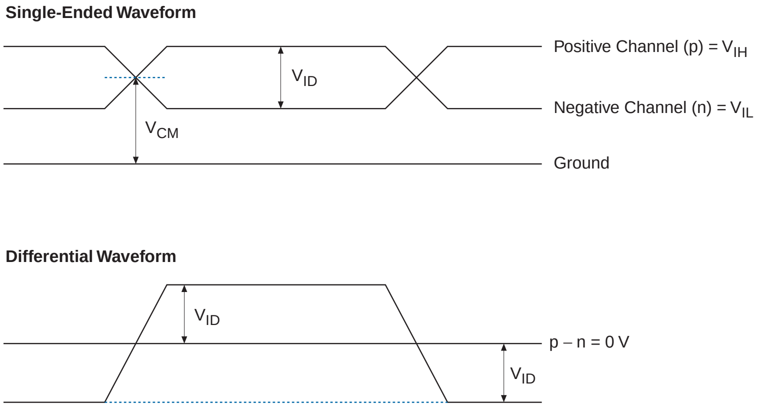
Glossary

Table 1–45 shows the glossary for this chapter.

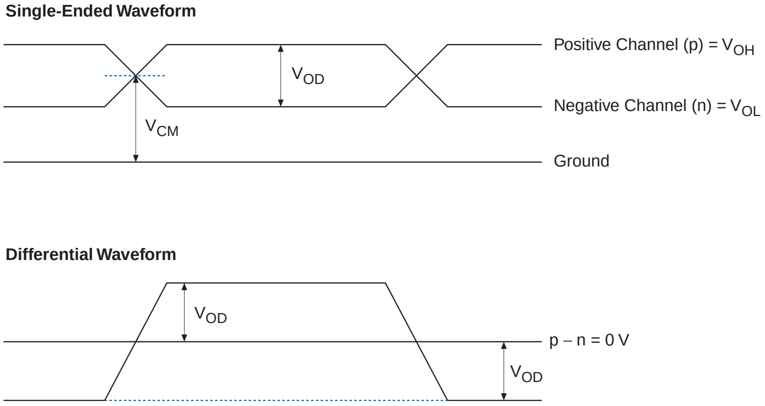
Table 1–45. Glossary Table

<table>
<tr><th>Letter</th><th>Subject</th><th>Definitions</th></tr>
<tr><td>A</td><td>—</td><td>—</td></tr>
<tr><td>B</td><td>—</td><td>—</td></tr>
<tr><td>C</td><td>—</td><td>—</td></tr>
<tr><td>D</td><td>Differential I/O Standards</td><td>

Receiver Input Waveforms



Transmitter Output Waveforms



</td></tr>
<tr><td>E</td><td>—</td><td>—</td></tr>
</table>

Table 1–45. Glossary Table

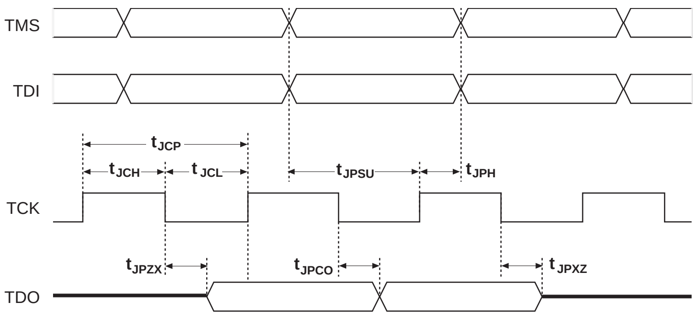
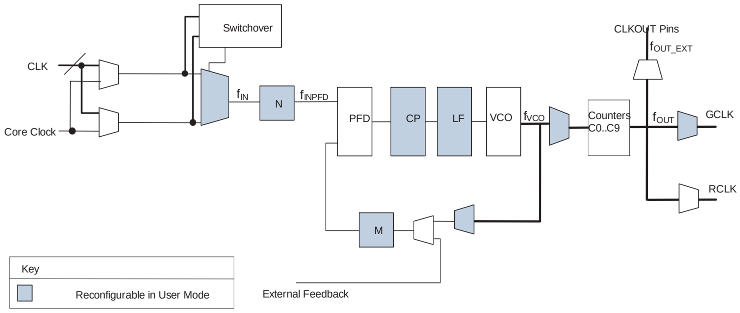
Letter	Subject	Definitions
F	f_{HSCLK}	Left/Right PLL input clock frequency.
	f_{HSDR}	HIGH-SPEED I/O Block: Maximum/minimum LVDS data transfer rate (f_{HSDR} = 1/TUI), non-DPA.
	$f_{HSDRDPA}$	HIGH-SPEED I/O Block: Maximum/minimum LVDS data transfer rate ($f_{HSDRDPA}$ = 1/TUI), DPA.
G	—	—
H	—	—
I	—	—
J	J	HIGH-SPEED I/O Block: Deserialization factor (width of parallel data bus).
	JTAG Timing Specifications	JTAG Timing Specifications are in the following figure:  TMS TDI t_{JCP} t_{JCH} t_{JCL} t_{JPSU} t_{JPH} TCK t_{JPZX} t_{JPCO} t_{JPXZ} TDO
K	—	—
L	—	—
M	—	—
N	—	—
O	—	—
P	**PLL Specifications**	The block diagram shown in the following figure highlights the PLL Specification parameters: **Diagram of PLL Specifications** *(1)*  Switchover CLKOUT Pins f_{OUT_EXT} CLK Core Clock f_{IN} N f_{INPFD} PFD CP LF VCO f_{VCO} Counters C0..C9 f_{OUT} GCLK RCLK M Key Reconfigurable in User Mode External Feedback **Note:** (1) CoreClock can only be fed by dedicated clock input pins or PLL outputs.
Q	—	—
R	R_L	Receiver differential input discrete resistor (external to Stratix IV device).

Table 1–45. Glossary Table

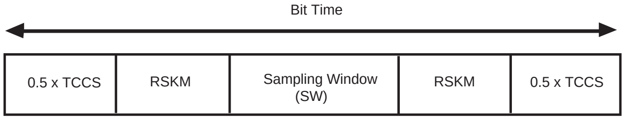
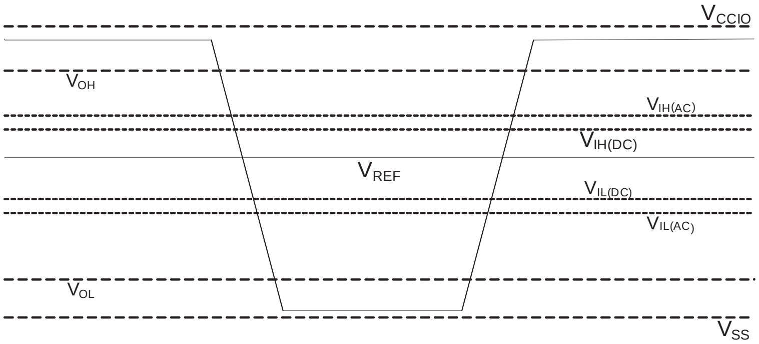
Letter	Subject	Definitions
S	**SW (sampling window)**	The period of time during which the data must be valid in order to capture it correctly. The setup and hold times determine the ideal strobe position within the sampling window as shown in (the following figure):  Bit Time 0.5 x TCCS RSKM Sampling Window (SW) RSKM 0.5 x TCCS *Timing Diagram*
	Single-ended voltage referenced I/O standard	The JEDEC standard for SSTl and HSTL I/O defines both the AC and DC input signal values. The AC values indicate the voltage levels at which the receiver must meet its timing specifications. The DC values indicate the voltage levels at which the final logic state of the receiver is unambiguously defined. Once the receiver input has crossed the AC value, the receiver changes to the new logic state. The new logic state is then maintained as long as the input stays beyond the AC threshold. This approach is intended to provide predictable receiver timing in the presence of input waveform ringing as shown in the following figure: *Single-Ended Voltage Referenced I/O Standard*  V_{CCIO} V_{OH} $V_{IH(AC)}$ $V_{IH(DC)}$ V_{REF} $V_{IL(DC)}$ $V_{IL(AC)}$ V_{OL} V_{SS}
T	t_C	High-speed receiver/transmitter input and output clock period.
	TCCS (channel-to-channel-skew)	The timing difference between the fastest and the slowest output edges, including t_{CO} variation and clock skew, across channels driven by the same PLL. The clock is included in the TCCS measurement (refer to the *Timing Diagram* figure under **S** in this table)
	t_{DUTY}	HIGH-SPEED I/O Block: Duty cycle on high-speed transmitter output clock. **Timing Unit Interval (TUI)** The timing budget allowed for skew, propagation delays, and data sampling window. (TUI = 1/(Receiver Input Clock Frequency Multiplication Factor) = t_C/w)
	t_{FALL}	Signal high-to-low transition time (80-20%)
	t_{INCCJ}	Cycle-to-cycle jitter tolerance on PLL clock input
	t_{OUTPJ_IO}	Period jitter on general purpose I/O driven by a PLL
	t_{OUTPJ_DC}	Period jitter on dedicated clock output driven by a PLL
	t_{RISE}	Signal Low-to-high transition time (20-80%)
U	—	—

Table 1–45. Glossary Table

Letter	Subject	Definitions
V	$V_{CM(DC)}$	DC Common Mode Input Voltage.
	V_{ICM}	Input Common Mode Voltage: The common mode of the differential signal at the receiver.
	V_{ID}	Input differential Voltage Swing: The difference in voltage between the positive and complementary conductors of a differential transmission at the receiver.
	$V_{DIF(AC)}$	AC differential Input Voltage: Minimum AC input differential voltage required for switching.
	$V_{DIF(DC)}$	DC differential Input Voltage: Minimum DC input differential voltage required for switching.
	V_{IH}	Voltage Input High: The minimum positive voltage applied to the input which will be accepted by the device as a logic high.
	$V_{IH(AC)}$	High-level AC input voltage
	$V_{IH(DC)}$	High-level DC input voltage
	V_{IL}	Voltage Input Low: The maximum positive voltage applied to the input which will be accepted by the device as a logic low.
	$V_{IL(AC)}$	Low-level AC input voltage
	$V_{IL(DC)}$	Low-level DC input voltage
	V_{OCM}	Output Common Mode Voltage: The common mode of the differential signal at the transmitter.
	V_{OD}	Output differential Voltage Swing: The difference in voltage between the positive and complementary conductors of a differential transmission at the transmitter.
W	W	HIGH-SPEED I/O BLOCK: Clock Boost Factor
X	—	—
Y	—	—
Z	—	—

Documents Referenced

This chapter references the following documents:

- *External Memory Interfaces in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook*
- *PowerPlay Early Power Estimator User Guide for Stratix III and Stratix IV FPGAs*

PowerPlay Power Analysis chapter in the *Quartus II Handbook*

Document Revision History

Table 1–46 shows the revision history for this document.

Table 1–46. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2008 v2.1	■ Edited "I/O Timing" section	—
November 2008 v2.0	■ Minor text edits. ■ Updated Table 1–19, Table 1–32, Table 1–34 - Table 1–39	Minor text edits.
August 2008 v1.1	■ Updated Table 1–1, Table 1–2, Table 1–4, Table 1–5, and Table 1–26. ■ Removed figures from "Transceiver Performance Specifications" on page 1–10 that are repeated in the glossary.	Minor text edits and an additional note to Table 1–26.
May 2008 v1.0	Initial release.	—