Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

Section I. Arria GX Device Data Sheet

This section provides designers with the data sheet specifications for Arria® GX devices. They contain feature definitions of the transceivers, internal architecture, configuration, and JTAG boundary-scan testing information, DC operating conditions, AC timing parameters, a reference to power consumption, and ordering information for Arria GX devices.

This section includes the following chapters:

Revision History

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

1. Arria GX Device Family Overview

AGX51001-2.0

Introduction

The Arria® GX family of devices combines 3.125 Gbps serial transceivers with reliable packaging technology and a proven logic array. Arria GX devices include 4 to 12 high-speed transceiver channels, each incorporating clock data recovery (CDR) technology and embedded SERDES circuitry designed to support PCI-Express, Gigabit Ethernet, SDI, SerialLite II, XAUI, and Serial RapidIO protocols, along with the ability to develop proprietary, serial-based IP using its Basic mode. The transceivers build upon the success of the Stratix® II GX family. The Arria GX FPGA technology offers a 1.2-V logic array with the right level of performance and dependability needed to support these mainstream protocols.

Features

The key features of Arria GX devices include:

- Transceiver block features
  - High-speed serial transceiver channels with CDR support up to 3.125 Gbps.
  - Devices available with 4, 8, or 12 high-speed full-duplex serial transceiver channels
  - Support for the following CDR-based bus standards—PCI Express, Gigabit Ethernet, SDI, SerialLite II, XAUI, and Serial RapidIO, along with the ability to develop proprietary, serial-based IP using its Basic mode
  - Individual transmitter and receiver channel power-down capability for reduced power consumption during non-operation
  - 1.2- and 1.5-V pseudo current mode logic (PCML) support on transmitter output buffers
  - Receiver indicator for loss of signal (available only in PCI Express [PIPE] mode)
  - Hot socketing feature for hot plug-in or hot swap and power sequencing support without the use of external devices
  - Dedicated circuitry that is compliant with PIPE, XAUI, Gigabit Ethernet, Serial Digital Interface (SDI), and Serial RapidIO
  - 8B/10B encoder/decoder performs 8-bit to 10-bit encoding and 10-bit to 8-bit decoding
  - Phase compensation FIFO buffer performs clock domain translation between the transceiver block and the logic array
  - Channel aligner compliant with XAUI

- Main device features:
  - TriMatrix memory consisting of three RAM block sizes to implement true dual-port memory and first-in first-out (FIFO) buffers with performance up to 380 MHz
  - Up to 16 global clock networks with up to 32 regional clock networks per device
  - High-speed DSP blocks provide dedicated implementation of multipliers, multiply-accumulate functions, and finite impulse response (FIR) filters
  - Up to four enhanced phase-locked loops (PLLs) per device provide spread spectrum, programmable bandwidth, clock switch-over, and advanced multiplication and phase shifting
  - Support for numerous single-ended and differential I/O standards
  - High-speed source-synchronous differential I/O support on up to 47 channels
  - Support for source-synchronous bus standards, including SPI-4 Phase 2 (POS-PHY Level 4), SFI-4.1, XSBI, UTOPIA IV, NPSI, and CSIX-L1
  - Support for high-speed external memory including DDR and DDR2 SDRAM, and SDR SDRAM
  - Support for multiple intellectual property megafunctions from Altera® MegaCore® functions and Altera Megafunction Partners Program (AMPP℠)
  - Support for remote configuration updates

Table 1–1 lists Arria GX device features for FineLine BGA (FBGA) with flip chip packages.

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

Feature	EP1AGX20C	EP1AGX35C/D		EP1AGX50C/D		EP1AGX60C/D/E			EP1AGX90E
	C	C	D	C	D	C	D	E	E
Package	484-pin, 780-pin (Flip chip)	484-pin (Flip chip)	780-pin (Flip chip)	484-pin (Flip chip)	780-pin, 1152-pin (Flip chip)	484-pin (Flip chip)	780-pin (Flip chip)	1152-pin (Flip chip)	1152-pin (Flip chip)
ALMs	8,632	13,408		20,064		24,040			36,088
Equivalent logic elements (LEs)	21,580	33,520		50,160		60,100			90,220
Transceiver channels	4	4	8	4	8	4	8	12	12
Transceiver data rate	600 Mbps to 3.125 Gbps	600 Mbps to 3.125 Gbps		600 Mbps to 3.125 Gbps		600 Mbps to 3.125 Gbps			600 Mbps to 3.125 Gbps
Source-synchronous receive channels	31	31	31	31	31, 42	31	31	42	47

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

Feature	EP1AGX20C	EP1AGX35C/D		EP1AGX50C/D		EP1AGX60C/D/E			EP1AGX90E
	C	C	D	C	D	C	D	E	E
Source-synchronous transmit channels	29	29	29	29	29, 42	29	29	42	45
M512 RAM blocks (32 × 18 bits)	166	197		313		326			478
M4K RAM blocks (128 × 36 bits)	118	140		242		252			400
M-RAM blocks (4096 × 144 bits)	1	1		2		2			4
Total RAM bits	1,229,184	1,348,416		2,475,072		2,528,640			4,477,824
Embedded multipliers (18 × 18)	40	56		104		128			176
DSP blocks	10	14		26		32			44
PLLs	4	4		4	4, 8	4		8	8
Maximum user I/O pins	230, 341	230	341	229	350, 514	229	350	514	538

Arria GX devices are available in space-saving FBGA packages (refer to Table 1–2). All Arria GX devices support vertical migration within the same package. With vertical migration support, designers can migrate to devices whose dedicated pins, configuration pins, and power pins are the same for a given package across device densities. For I/O pin migration across densities, the designer must cross-reference the available I/O pins with the device pin-outs for all planned densities of a given package type to identify which I/O pins are migratable.

Table 1–2. Arria GX Package Options (Pin Counts and Transceiver Channels) (Part 1 of 2)

Device	Transceiver Channels	Source-Synchronous Channels		Maximum User I/O Pin Count		
		Receive	Transmit	484-Pin FBGA (23 mm)	780-Pin FBGA (29 mm)	1152-Pin FBGA (35 mm)
EP1AGX20C	4	31	29	230	341	—
EP1AGX35C	4	31	29	230	—	—
EP1AGX50C	4	31	29	229	—	—
EP1AGX60C	4	31	29	229	—	—
EP1AGX35D	8	31	29	—	341	—
EP1AGX50D	8	31, 42	29, 42	—	350	514

Table 1–2. Arria GX Package Options (Pin Counts and Transceiver Channels) (Part 2 of 2)

Device	Transceiver Channels	Source-Synchronous Channels		Maximum User I/O Pin Count		
		Receive	Transmit	484-Pin FBGA (23 mm)	780-Pin FBGA (29 mm)	1152-Pin FBGA (35 mm)
EP1AGX60D	8	31	29	—	350	—
EP1AGX60E	12	42	42	—	—	514
EP1AGX90E	12	47	45	—	—	538

Table 1–3 lists the Arria GX device package sizes.

Table 1–3. Arria GX FBGA Package Sizes

Dimension	484 Pins	780 Pins	1152 Pins
Pitch (mm)	1.00	1.00	1.00
Area (mm^2)	529	841	1225
Length × width (mm × mm)	23 × 23	29 × 29	35 × 35

Document Revision History

Table 1–4 lists the revision history for this chapter.

Table 1–4. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
December 2009, v2.0	■ Document template update. ■ Minor text edits.	—
May 2008, v1.2	Included support for SDI, SerialLite II, and XAUI.	—
June 2007, v1.1	Included GIGE information.	—
May 2007, v1.0	Initial Release	—

2. Arria GX Architecture

Transceivers

Arria® GX devices incorporate up to 12 high-speed serial transceiver channels that build on the success of the Stratix® II GX device family. Arria GX transceivers are structured into full-duplex (transmitter and receiver) four-channel groups called transceiver blocks located on the right side of the device. You can configure the transceiver blocks to support the following serial connectivity protocols (functional modes):

- PCI Express (PIPE)
- Gigabit Ethernet (GIGE)
- XAUI
- Basic (600 Mbps to 3.125 Gbps)
- SDI (HD, 3G)
- Serial RapidIO (1.25 Gbps, 2.5 Gbps, 3.125 Gbps)

Transceivers within each block are independent and have their own set of dividers. Therefore, each transceiver can operate at different frequencies. Each block can select from two reference clocks to provide two clock domains that each transceiver can select from.

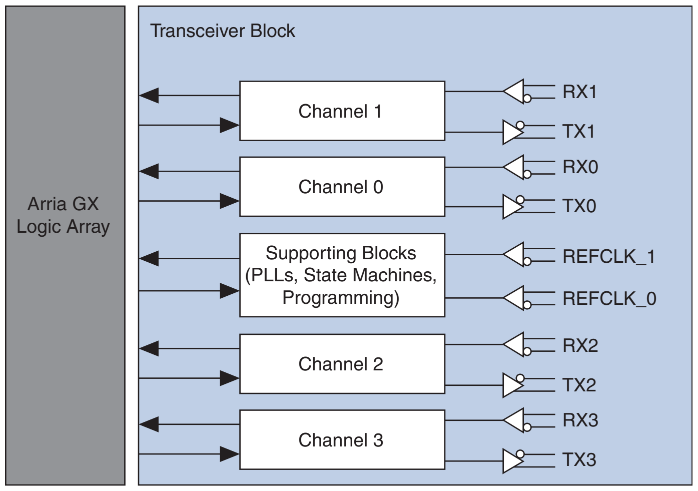
Table 2–1 lists the number of transceiver channels for each member of the Arria GX family.

Table 2–1. Arria GX Transceiver Channels

Device	Number of Transceiver Channels
EP1AGX20C	4
EP1AGX35C	4
EP1AGX35D	8
EP1AGX50C	4
EP1AGX50D	8
EP1AGX60C	4
EP1AGX60D	8
EP1AGX60E	12
EP1AGX90E	12

Figure 2–1 shows a high-level diagram of the transceiver block architecture divided into four channels.

Figure 2–1. Transceiver Block

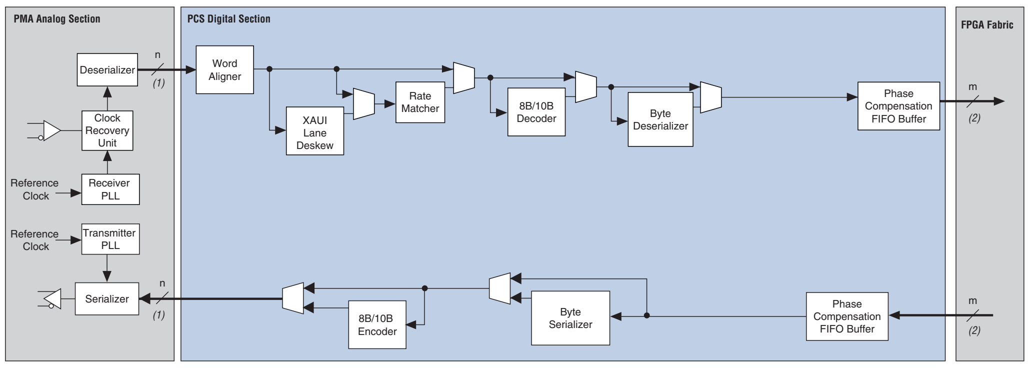


Each transceiver block has:

- Four transceiver channels with dedicated physical coding sublayer (PCS) and physical media attachment (PMA) circuitry
- One transmitter PLL that takes in a reference clock and generates high-speed serial clock depending on the functional mode
- Four receiver PLLs and clock recovery unit (CRU) to recover clock and data from the received serial data stream
- State machines and other logic to implement special features required to support each protocol

Figure 2–2 shows functional blocks that make up a transceiver channel.

Figure 2–2. Arria GX Transceiver Channel Block Diagram



Notes to Figure 2–2:

(1) "n" represents the number of bits in each word that must be serialized by the transmitter portion of the PMA. n = 8 or 10.

(2) "m" represents the number of bits in the word that passes between the FPGA logic and the PCS portion of the transceiver. m = 8, 10, 16, or 20.

Each transceiver channel is full-duplex and consists of a transmitter channel and a receiver channel.

The transmitter channel contains the following sub-blocks:

- Transmitter phase compensation first-in first-out (FIFO) buffer
- Byte serializer (optional)
- 8B/10B encoder (optional)
- Serializer (parallel-to-serial converter)
- Transmitter differential output buffer

The receiver channel contains the following:

- Receiver differential input buffer
- Receiver lock detector and run length checker
- CRU
- Deserializer
- Pattern detector
- Word aligner
- Lane deskew
- Rate matcher (optional)
- 8B/10B decoder (optional)
- Byte deserializer (optional)
- Receiver phase compensation FIFO buffer

You can configure the transceiver channels to the desired functional modes using the ALT2GXB MegaCore instance in the Quartus® II MegaWizard™ Plug-in Manager for the Arria GX device family. Depending on the selected functional mode, the Quartus II software automatically configures the transceiver channels to employ a subset of the sub-blocks listed above.

Transmitter Path

This section describes the data path through the Arria GX transmitter. The sub-blocks are described in order from the PLD-transmitter parallel interface to the serial transmitter buffer.

Clock Multiplier Unit

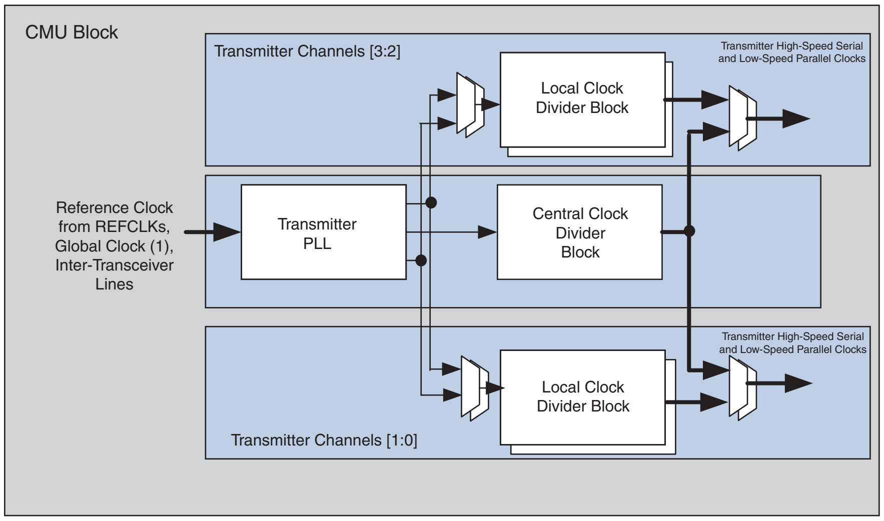
Each transceiver block has a clock multiplier unit (CMU) that takes in a reference clock and synthesizes two clocks: a high-speed serial clock to serialize the data and a low-speed parallel clock to clock the transmitter digital logic (PCS).

The CMU is further divided into three sub-blocks:

- One transmitter PLL
- One central clock divider block
- Four local clock divider blocks (one per channel)

Figure 2–3 shows the block diagram of the clock multiplier unit.

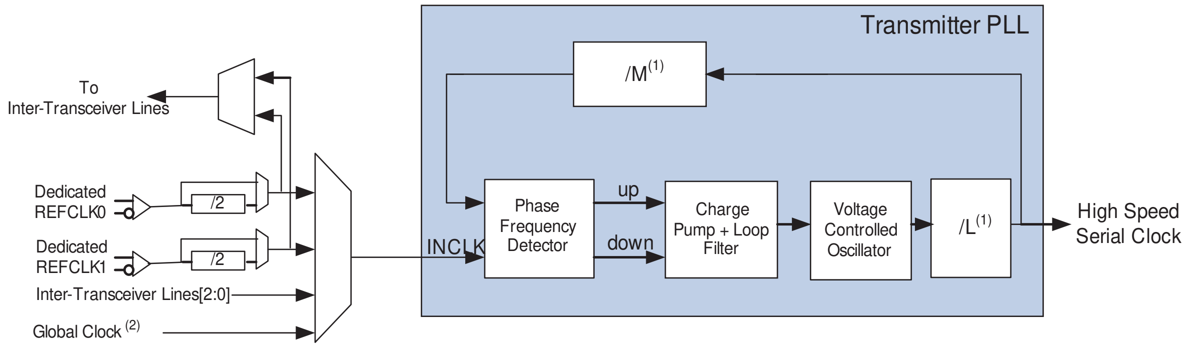
Figure 2–3. Clock Multiplier Unit


CMU Block
Transmitter Channels [3:2]
Transmitter High-Speed Serial and Low-Speed Parallel Clocks
Local Clock Divider Block
Reference Clock from REFCLKs, Global Clock (1), Inter-Transceiver Lines
Transmitter PLL
Central Clock Divider Block
Transmitter High-Speed Serial and Low-Speed Parallel Clocks
Local Clock Divider Block
Transmitter Channels [1:0]

The transmitter PLL multiplies the input reference clock to generate the high-speed serial clock required to support the intended protocol. It implements a half-rate voltage controlled oscillator (VCO) that generates a clock at half the frequency of the serial data rate for which it is configured.

Figure 2–4 shows the block diagram of the transmitter PLL.

Figure 2–4. Transmitter PLL



Notes to Figure 2–4:

(1) You only need to select the protocol and the available input reference clock frequency in the ALTGXB MegaWizard Plug-In Manager. Based on your selections, the MegaWizard Plug-In Manager automatically selects the necessary /M and /L dividers (clock multiplication factors).

(2) The global clock line must be driven from an input pin only.

The reference clock input to the transmitter PLL can be derived from:

- One of two available dedicated reference clock input pins (`REFCLK0` or `REFCLK1`) of the associated transceiver block
- PLD global clock network (must be driven directly from an input clock pin and cannot be driven by user logic or enhanced PLL)

- Inter-transceiver block lines driven by reference clock input pins of other transceiver blocks

Altera® recommends using the dedicated reference clock input pins (`REFCLK0` or `REFCLK1`) to provide reference clock for the transmitter PLL.

Table 2–2 lists the adjustable parameters in the transmitter PLL.

Table 2–2. Transmitter PLL Specifications

Parameter	Specifications
Input reference frequency range	50 MHz to 622.08 MHz
Data rate support	600 Mbps to 3.125 Gbps
Bandwidth	Low, medium, or high

The transmitter PLL output feeds the central clock divider block and the local clock divider blocks. These clock divider blocks divide the high-speed serial clock to generate the low-speed parallel clock for the transceiver PCS logic and PLD-transceiver interface clock.

Transmitter Phase Compensation FIFO Buffer

A transmitter phase compensation FIFO is located at each transmitter channel's logic array interface. It compensates for the phase difference between the transmitter PCS clock and the local PLD clock. The transmitter phase compensation FIFO is used in all supported functional modes. The transmitter phase compensation FIFO buffer is eight words deep in PCI Express (PIPE) mode and four words deep in all other modes.

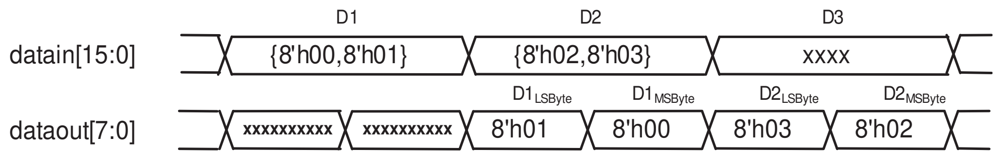
For more information about architecture and clocking, refer to the *Arria GX Transceiver Architecture* chapter.

Byte Serializer

The byte serializer takes in two-byte wide data from the transmitter phase compensation FIFO buffer and serializes it into a one-byte wide data at twice the speed. The transmit data path after the byte serializer is 8 or 10 bits. This allows clocking the PLD-transceiver interface at half the speed when compared with the transmitter PCS logic. The byte serializer is bypassed in GIGE mode. After serialization, the byte serializer transmits the least significant byte (LSByte) first and the most significant byte (MSByte) last.

Figure 2–5 shows byte serializer input and output. `datain[15:0]` is the input to the byte serializer from the transmitter phase compensation FIFO; `dataout[7:0]` is the output of the byte serializer.

Figure 2–5. Byte Serializer Operation *(Note 1)*



Note to Figure 2–5:

(1) `datain` may be 16 or 20 bits. `dataout` may be 8 or 10 bits.

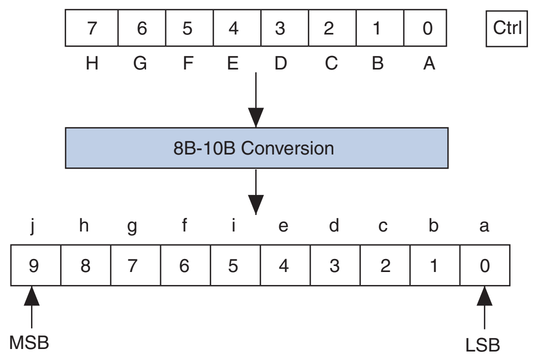
8B/10B Encoder

The 8B/10B encoder block is used in all supported functional modes. The 8B/10B encoder block takes in 8-bit data from the byte serializer or the transmitter phase compensation FIFO buffer. It generates a 10-bit code group with proper running disparity from the 8-bit character and a 1-bit control identifier (`tx_ctrlenable`). When `tx_ctrlenable` is low, the 8-bit character is encoded as data code group (`Dx.y`). When `tx_ctrlenable` is high, the 8-bit character is encoded as a control code group (`Kx.y`). The 10-bit code group is fed to the serializer. The 8B/10B encoder conforms to the IEEE 802.3 1998 edition standard.

For additional information regarding 8B/10B encoding rules, refer to the *Specifications and Additional Information* chapter.

Figure 2–6 shows the 8B/10B conversion format.

Figure 2–6. 8B/10B Encoder



During reset (`tx_digitalreset`), the running disparity and data registers are cleared and the 8B/10B encoder continously outputs a K28.5 pattern from the RD-column. After out of reset, the 8B/10B encoder starts with a negative disparity (RD-) and transmits three K28.5 code groups for synchronizing before it starts encoding the input data or control character.

Transmit State Machine

The transmit state machine operates in either PCI Express (PIPE) mode, XAUI mode, or GIGE mode, depending on the protocol used.

GIGE Mode

In GIGE mode, the transmit state machine converts all idle ordered sets (/K28.5/, /Dx.y/) to either /I1/ or /I2/ ordered sets. The /I1/ set consists of a negative-ending disparity /K28.5/ (denoted by /K28.5/-), followed by a neutral /D5.6/. The /I2/ set consists of a positive-ending disparity /K28.5/ (denoted by /K28.5/+) and a negative-ending disparity /D16.2/ (denoted by /D16.2/-). The transmit state machines do not convert any of the ordered sets to match /C1/ or /C2/, which are the configuration ordered sets. (/C1/ and /C2/ are defined by [/K28.5/, /D21.5/] and [/K28.5/, /D2.2/], respectively). Both the /I1/ and /I2/ ordered sets guarantee a negative-ending disparity after each ordered set.

XAUI Mode

The transmit state machine translates the XAUI XGMII code group to the XAUI PCS code group. Table 2–3 lists the code conversion.

Table 2–3. On-Chip Termination Support by I/O Banks

XGMII TXC	XGMII TXD	PCS Code-Group	Description
0	00 through FF	Dxx.y	Normal data
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	Refer to IEEE 802.3 reserved code groups	Refer to IEEE 802.3 reserved code groups	Reserved code groups
1	Other value	K30.7	Invalid XGMII character

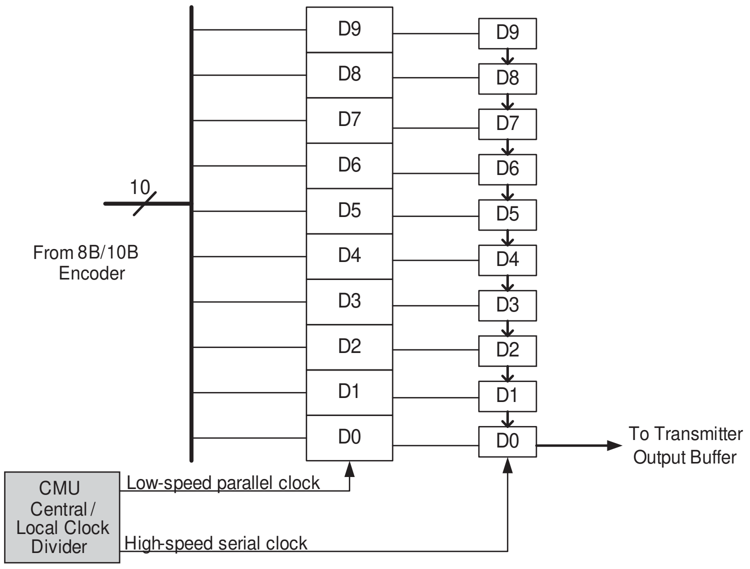
The XAUI PCS idle code groups, /K28.0/ (/R/) and /K28.5/ (/K/), are automatically randomized based on a PRBS7 pattern with an ×7 + ×6 + 1 polynomial. The /K28.3/ (/A/) code group is automatically generated between 16 and 31 idle code groups. The idle randomization on the /A/, /K/, and /R/ code groups is automatically done by the transmit state machine.

Serializer (Parallel-to-Serial Converter)

The serializer block clocks in 8- or 10-bit encoded data from the 8B/10B encoder using the low-speed parallel clock and clocks out serial data using the high-speed serial clock from the central or local clock divider blocks. The serializer feeds the data LSB to MSB to the transmitter output buffer.

Figure 2–7 shows the serializer block diagram.

Figure 2–7. Serializer



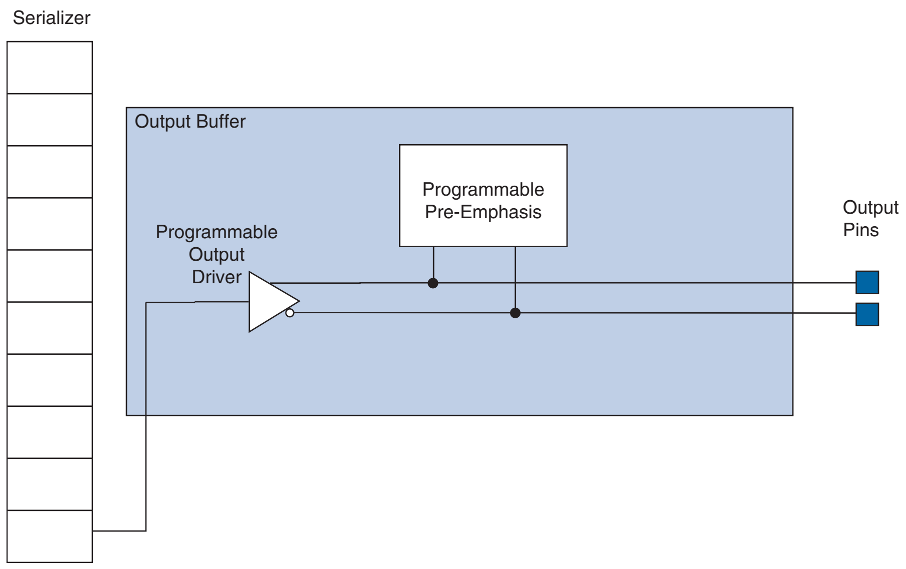
Transmitter Buffer

The Arria GX transceiver buffers support the 1.2- and 1.5-V PCML I/O standard at rates up to 3.125 Gbps. The common mode voltage (V_{CM}) of the output driver may be set to 600 or 700 mV.

For more information about the Arria GX transceiver buffers, refer to the *Arria GX Transceiver Architecture* chapter.

The output buffer, as shown in Figure 2–8, is directly driven by the high-speed data serializer and consists of a programmable output driver, a programmable pre-emphasis circuit, and OCT circuitry.

Figure 2–8. Output Buffer



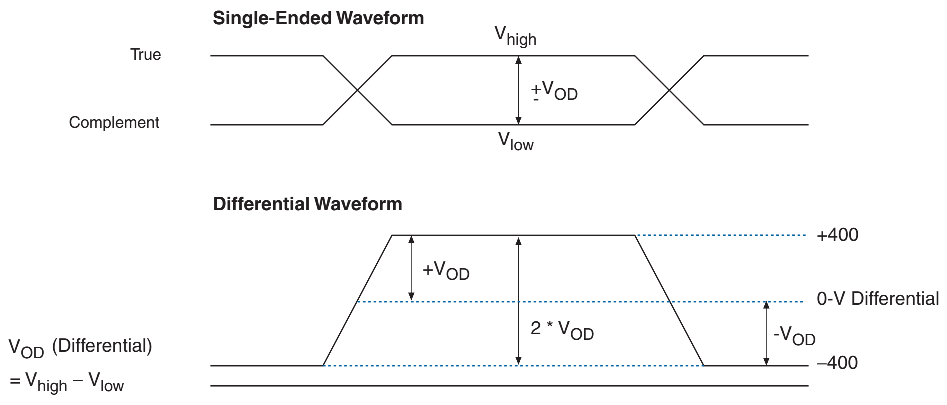
Programmable Output Driver

The programmable output driver can be set to drive out differentially from 400 to 1200 mV. The differential output voltage (V_{OD}) can be statically set by using the ALTGXB megafunction.

You can configure the output driver with 100-Ω OCT or external OCT.

Differential signaling conventions are shown in Figure 2–9. The differential amplitude represents the value of the voltage between the true and complement signals. Peak-to-peak differential voltage is defined as 2 (V_{HIGH} – V_{LOW}) = 2 single-ended voltage swing. The common mode voltage is the average of V_{HIGH} and V_{LOW}.

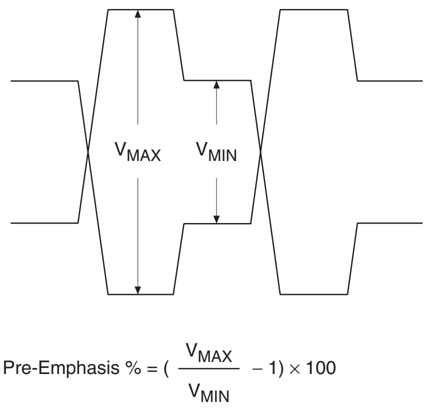
Figure 2–9. Differential Signaling



Programmable Pre-Emphasis

The programmable pre-emphasis module controls the output driver to boost high frequency components and compensate for losses in the transmission medium, as shown in Figure 2–10. Pre-emphasis is set statically using the ALTGXB megafunction.

Figure 2–10. Pre-Emphasis Signaling



Pre-emphasis percentage is defined as (V_{MAX}/V_{MIN} – 1) × 100, where V_{MAX} is the differential emphasized voltage (peak-to-peak) and V_{MIN} is the differential steady-state voltage (peak-to-peak).

PCI Express (PIPE) Receiver Detect

The Arria GX transmitter buffer has a built-in receiver detection circuit for use in PCI Express (PIPE) mode. This circuit provides the ability to detect if there is a receiver downstream by sending out a pulse on the channel and monitoring the reflection. This mode requires a tri-stated transmitter buffer (in electrical idle mode).

PCI Express (PIPE) Electric Idles (or Individual Transmitter Tri-State)

The Arria GX transmitter buffer supports PCI Express (PIPE) electrical idles. This feature is only active in PCI Express (PIPE) mode. The `tx_forceelecidle` port puts the transmitter buffer in electrical idle mode. This port is available in all PCI Express (PIPE) power-down modes and has specific usage in each mode.

Receiver Path

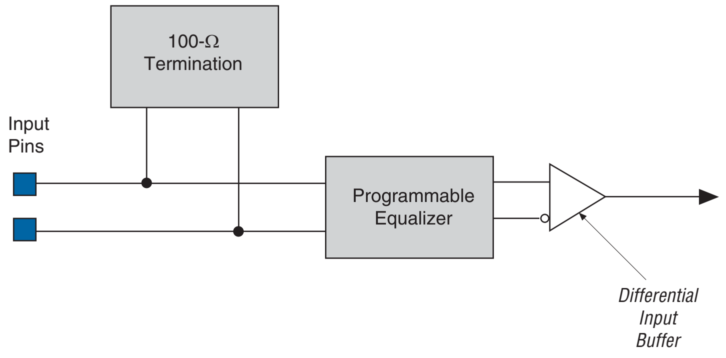
This section describes the data path through the Arria GX receiver. The sub-blocks are described in order from the receiver buffer to the PLD-receiver parallel interface.

Receiver Buffer

The Arria GX receiver input buffer supports the 1.2-V and 1.5-V PCML I/O standards at rates up to 3.125 Gbps. The common mode voltage of the receiver input buffer is programmable between 0.85 V and 1.2 V. You must select the 0.85 V common mode voltage for AC- and DC-coupled PCML links and 1.2 V common mode voltage for DC-coupled LVDS links.

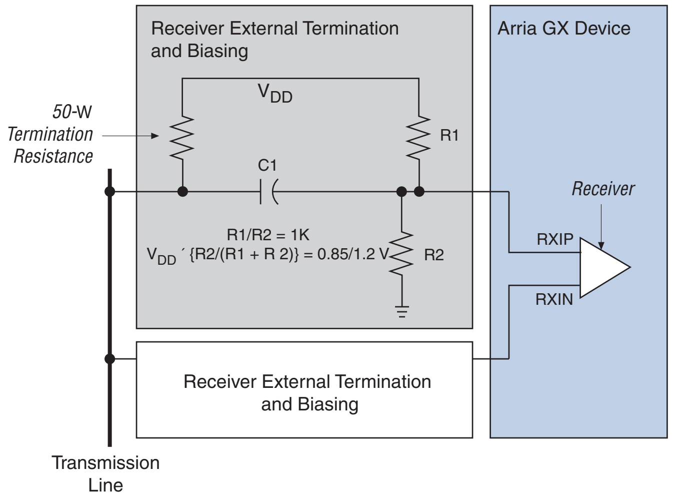
The receiver has 100-Ω on-chip differential termination (R_D OCT) for different protocols, as shown in Figure 2–11. You can disable the receiver's internal termination if external terminations and biasing are provided. The receiver and transmitter differential termination method can be set independently of each other.

Figure 2–11. Receiver Input Buffer



If a design uses external termination, the receiver must be externally terminated and biased to 0.85 V or 1.2 V. Figure 2–12 shows an example of an external termination and biasing circuit.

Figure 2–12. External Termination and Biasing Circuit



Programmable Equalizer

The Arria GX receivers provide a programmable receiver equalization feature to compensate for the effects of channel attenuation for high-speed signaling. PCB traces carrying these high-speed signals have low-pass filter characteristics. Impedance mismatch boundaries can also cause signal degradation. Equalization in the receiver diminishes the lossy attenuation effects of the PCB at high frequencies.

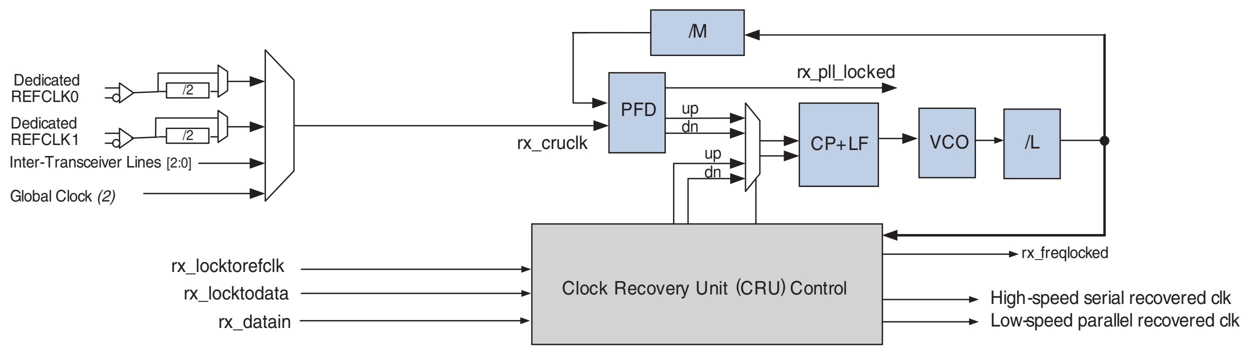
The receiver equalization circuit is comprised of a programmable amplifier. Each stage is a peaking equalizer with a different center frequency and programmable gain. This allows varying amounts of gain to be applied, depending on the overall frequency response of the channel loss. Channel loss is defined as the summation of all losses through the PCB traces, vias, connectors, and cables present in the physical link. The Quartus II software allows five equalization settings for Arria GX devices.

Receiver PLL and Clock Recovery Unit (CRU)

Each transceiver block has four receiver PLLs and CRU units, each of which is dedicated to a receiver channel. The receiver PLL is fed by an input reference clock. The receiver PLL, in conjunction with the CRU, generates two clocks: a high-speed serial recovered clock that clocks the deserializer and a low-speed parallel recovered clock that clocks the receiver's digital logic.

Figure 2–13 shows a block diagram of the receiver PLL and CRU circuits.

Figure 2–13. Receiver PLL and Clock Recovery Unit



Notes to Figure 2–13:

(1) You only need to select the protocol and the available input reference clock frequency in the ALTGXB MegaWizard Plug-In Manager. Based on your selections, the ALTGXB MegaWizard Plug-In Manager automatically selects the necessary /M and /L dividers.

(2) The global clock line must be driven from an input pin only.

The reference clock input to the receiver PLL can be derived from:

- One of the two available dedicated reference clock input pins (`REFCLK0` or `REFCLK1`) of the associated transceiver block
- PLD global clock network (must be driven directly from an input clock pin and cannot be driven by user logic or enhanced PLL)
- Inter-transceiver block lines driven by reference clock input pins of other transceiver blocks

All the parameters listed are programmable in the Quartus II software. The receiver PLL has the following features:

- Operates from 600 Mbps to 3.125 Gbps.
- Uses a reference clock between 50 MHz and 622.08 MHz.
- Programmable bandwidth settings: low, medium, and high.
- Programmable `rx_locktorefclk` (forces the receiver PLL to lock to reference clock) and `rx_locktodata` (forces the receiver PLL to lock to data).

- The voltage-controlled oscillator (V_{CO}) operates at half rate.
- Programmable frequency multiplication W of 1, 4, 5, 8, 10, 16, 20, and 25. Not all settings are supported for any particular frequency.
- Two lock indication signals are provided. They are found in PFD mode (lock-to-reference clock), and PD (lock-to-data).

The CRU controls whether the receiver PLL locks to the input reference clock (lock-to-reference mode) or the incoming serial data (lock-to data mode). You can set the CRU to switch between lock-to-data and lock-to-reference modes automatically or manually. In automatic lock mode, the phase detector and dedicated parts per million (PPM) detector within each receiver channel control the switch between lock-to-data and lock-to-reference modes based on some pre-set conditions. In manual lock mode, you can control the switch manually using the `rx_locktorefclk` and `rx_locktodata` signals.

For more information, refer to the "Clock Recovery Unit" section in the *Arria GX Transceiver Protocol Support and Additional Features* chapter.

Table 2–4 lists the behavior of the CRU block with respect to the `rx_locktorefclk` and `rx_locktodata` signals.

Table 2–4. CRU Manual Lock Signals

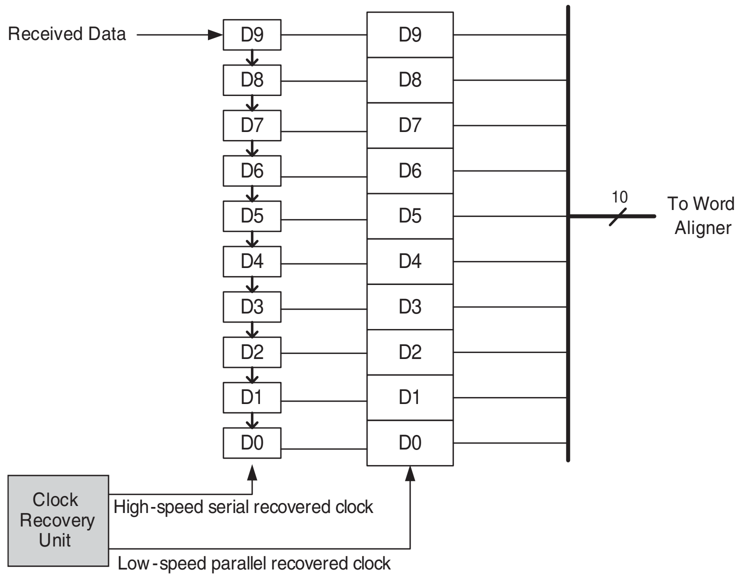
rx_locktorefclk	rx_locktodata	CRU Mode
1	0	Lock-to-reference clock
x	1	Lock-to-data
0	0	Automatic

If the `rx_locktorefclk` and `rx_locktodata` ports are not used, the default setting is automatic lock mode.

Deserializer

The deserializer block clocks in serial input data from the receiver buffer using the high-speed serial recovered clock and deserializes into 8- or 10-bit parallel data using the low-speed parallel recovered clock. The serial data is assumed to be received with LSB first, followed by MSB. It feeds the deserialized 8- or 10-bit data to the word aligner, as shown in Figure 2–14.

Figure 2–14. Deserializer *(Note 1)*



Note to Figure 2–14:

(1) This is a 10-bit deserializer. The deserializer can also convert 8 bits of data.

Word Aligner

The deserializer block creates 8- or 10-bit parallel data. The deserializer ignores protocol symbol boundaries when converting this data. Therefore, the boundaries of the transferred words are arbitrary. The word aligner aligns the incoming data based on specific byte or word boundaries. The word alignment module is clocked by the local receiver recovered clock during normal operation. All the data and programmed patterns are defined as “big-endian” (most significant word followed by least significant word). Most-significant-bit-first protocols should reverse the bit order of word align patterns programmed.

This module detects word boundaries for 8B/10B-based protocols. This module is also used to align to specific programmable patterns in PRBS7/23 test mode.

Pattern Detection

The programmable pattern detection logic can be programmed to align word boundaries using a single 7- or 10-bit pattern. The pattern detector can either do an exact match, or match the exact pattern and the complement of a given pattern. Once the programmed pattern is found, the data stream is aligned to have the pattern on the LSB portion of the data output bus.

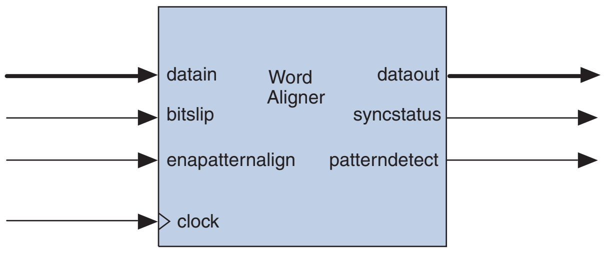
XAUI, GIGE, PCI Express (PIPE), and Serial RapidIO standards have embedded state machines for symbol boundary synchronization. These standards use K28.5 as their 10-bit programmed comma pattern. Each of these standards uses different algorithms before signaling symbol boundary acquisition to the FPGA.

Pattern detection logic searches from the LSB to the MSB. If multiple patterns are found within the search window, the pattern in the lower portion of the data stream (corresponding to the pattern received earlier) is aligned and the rest of the matching patterns are ignored.

Once a pattern is detected and the data bus is aligned, the word boundary is locked. The two detection status signals (`rx_syncstatus` and `rx_patterndetect`) indicate that an alignment is complete.

Figure 2–15 is a block diagram of the word aligner.

Figure 2–15. Word Aligner



Control and Status Signals

The `rx_enapatternalign` signal is the FPGA control signal that enables word alignment in non-automatic modes. The `rx_enapatternalign` signal is not used in automatic modes (PCI Express [PIPE], XAUI, GIGE, and Serial RapidIO).

In manual alignment mode, after the `rx_enapatternalign` signal is activated, the `rx_syncstatus` signal goes high for one parallel clock cycle to indicate that the alignment pattern has been detected and the word boundary has been locked. If `rx_enapatternalign` is deactivated, the `rx_syncstatus` signal acts as a re-synchronization signal to signify that the alignment pattern has been detected but not locked on a different word boundary.

When using the synchronization state machine, the `rx_syncstatus` signal indicates the link status. If the `rx_syncstatus` signal is high, link synchronization is achieved. If the `rx_syncstatus` signal is low, link synchronization has not yet been achieved, or there were enough code group errors to lose synchronization.

For more information about manual alignment modes, refer to the *Arria GX Device Handbook*.

The `rx_patterndetect` signal pulses high during a new alignment and whenever the alignment pattern occurs on the current word boundary.

Programmable Run Length Violation

The word aligner supports a programmable run length violation counter. Whenever the number of the continuous ‘0’ (or ‘1’) exceeds a user programmable value, the `rx_rlv` signal goes high for a minimum pulse width of two recovered clock cycles. The maximum run values supported are 128 UI for 8-bit serialization or 160 UI for 10-bit serialization.

Running Disparity Check

The running disparity error `rx_disperr` and running disparity value `rx_runningdisp` are sent along with aligned data from the 8B/10B decoder to the FPGA. You can ignore or act on the reported running disparity value and running disparity error signals.

Bit-Slip Mode

The word aligner can operate in either pattern detection mode or in bit-slip mode.

The bit-slip mode provides the option to manually shift the word boundary through the FPGA. This feature is useful for:

- Longer synchronization patterns than the pattern detector can accommodate
- Scrambled data stream
- Input stream consisting of over-sampled data

The word aligner outputs a word boundary as it is received from the analog receiver after reset. You can examine the word and search its boundary in the FPGA. To do so, assert the `rx_bitslip` signal. The `rx_bitslip` signal should be toggled and held constant for at least two FPGA clock cycles.

For every rising edge of the `rx_bitslip` signal, the current word boundary is slipped by one bit. Every time a bit is slipped, the bit received earliest is lost. If bit slipping shifts a complete round of bus width, the word boundary is back to the original boundary.

The `rx_syncstatus` signal is not available in bit-slipping mode.

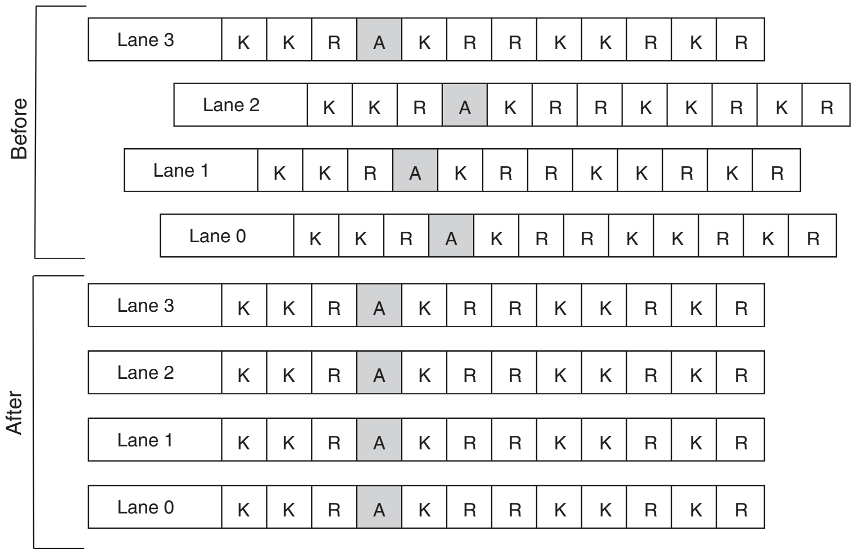
Channel Aligner

The channel aligner is available only in XAUI mode and aligns the signals of all four channels within a transceiver. The channel aligner follows the IEEE 802.3ae, clause 48 specification for channel bonding.

The channel aligner is a 16-word FIFO buffer with a state machine controlling the channel bonding process. The state machine looks for an /A/ (/K28.3/) in each channel and aligns all the /A/ code groups in the transceiver. When four columns of /A/ (denoted by //A//) are detected, the `rx_channelaligned` signal goes high, signifying that all the channels in the transceiver have been aligned. The reception of four consecutive misaligned /A/ code groups restarts the channel alignment sequence and sends the `rx_channelaligned` signal low.

Figure 2–16 shows misaligned channels before the channel aligner and the aligned channels after the channel aligner.

Figure 2–16. Before and After the Channel Aligner


Before
Lane 3 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 1 K K R A K R R K K R K R
Lane 0 K K R A K R R K K R K R
After
Lane 3 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 1 K K R A K R R K K R K R
Lane 0 K K R A K R R K K R K R

Rate Matcher

In asynchronous systems, the upstream transmitter and local receiver can be clocked with independent reference clock sources. Frequency differences in the order of a few hundred PPM can potentially corrupt the data at the receiver.

The rate matcher compensates for small clock frequency differences between the upstream transmitter and the local receiver clocks by inserting or removing skip characters from the inter packet gap (IPG) or idle streams. It inserts a skip character if the local receiver is running a faster clock than the upstream transmitter. It deletes a skip character if the local receiver is running a slower clock than the upstream transmitter. The Quartus II software automatically configures the appropriate skip character as specified in the IEEE 802.3 for GIGE mode and PCI-Express Base Specification for PCI Express (PIPE) mode. The rate matcher is bypassed in Serial RapidIO and must be implemented in the PLD logic array or external circuits depending on your system design.

Table 2–5 lists the maximum frequency difference that the rate matcher can tolerate in XAUI, PCI Express (PIPE), GIGE, and Basic functional modes.

Table 2–5. Rate Matcher PPM Tolerance

Function Mode	PPM
XAUI	± 100
PCI Express (PIPE)	± 300
GIGE	± 100
Basic	± 300

XAUI Mode

In XAUI mode, the rate matcher adheres to clause 48 of the IEEE 802.3ae specification for clock rate compensation. The rate matcher performs clock compensation on columns of /R/ (/K28.0/), denoted by //R//. An //R// is added or deleted automatically based on the number of words in the FIFO buffer.

PCI Express (PIPE) Mode Rate Matcher

In PCI Express (PIPE) mode, the rate matcher can compensate up to ± 300 PPM (600 PPM total) frequency difference between the upstream transmitter and the receiver. The rate matcher logic looks for skip ordered sets (SOS), which contains a /K28.5/ comma followed by three /K28.0/ skip characters. The rate matcher logic deletes or inserts /K28.0/ skip characters as necessary from/to the rate matcher FIFO.

The rate matcher in PCI Express (PIPE) mode has a FIFO buffer overflow and underflow protection. In the event of a FIFO buffer overflow, the rate matcher deletes any data after detecting the overflow condition to prevent FIFO pointer corruption until the rate matcher is not full. In an underflow condition, the rate matcher inserts 9'h1FE (/K30.7/) until the FIFO buffer is not empty. These measures ensure that the FIFO buffer can gracefully exit the overflow and underflow condition without requiring a FIFO reset. The rate matcher FIFO overflow and underflow condition is indicated on the `pipestatus` port.

You can bypass the rate matcher in PCI Express (PIPE) mode if you have a synchronous system where the upstream transmitter and local receiver derive their reference clocks from the same source.

GIGE Mode Rate Matcher

In GIGE mode, the rate matcher can compensate up to ± 100 PPM (200 PPM total) frequency difference between the upstream transmitter and the receiver. The rate matcher logic inserts or deletes /I2/ idle ordered sets to/from the rate matcher FIFO during the inter-frame or inter-packet gap (IFG or IPG). /I2/ is selected as the rate matching ordered set because it maintains the running disparity, unlike /I1/ that alters the running disparity. Because the /I2/ ordered-set contains two 10-bit code groups (/K28.5/, /D16.2/), 20 bits are inserted or deleted at a time for rate matching.

☞ The rate matcher logic has the capability to insert or delete /C1/ or /C2/ configuration ordered sets when 'GIGE Enhanced' mode is chosen as the sub-protocol in the MegaWizard Plug-In Manager.

If the frequency PPM difference between the upstream transmitter and the local receiver is high, or if the packet size is too large, the rate matcher FIFO buffer can face an overflow or underflow situation.

Basic Mode

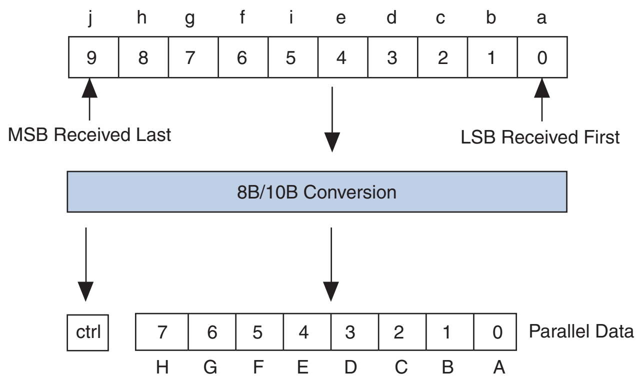
In basic mode, you can program the skip and control pattern for rate matching. There is no restriction on the deletion of a skip character in a cluster. The rate matcher deletes the skip characters as long as they are available. For insertion, the rate matcher inserts skip characters such that the number of skip characters at the output of rate matcher does not exceed five.

8B/10B Decoder

The 8B/10B decoder is used in all supported functional modes. The 8B/10B decoder takes in 10-bit data from the rate matcher and decodes it into 8-bit data + 1-bit control identifier, thereby restoring the original transmitted data at the receiver. The 8B/10B decoder indicates whether the received 10-bit character is a data or control code through the `rx_ctrldetect` port. If the received 10-bit code group is a control character (`Kx.y`), the `rx_ctrldetect` signal is driven high and if it is a data character (`Dx.y`), the `rx_ctrldetect` signal is driven low.

Figure 2–17 shows a 10-bit code group decoded to an 8-bit data and a 1-bit control indicator.

Figure 2–17. 10-Bit to 8-Bit Conversion



If the received 10-bit code is not a part of valid Dx.y or Kx.y code groups, the 8B/10B decoder block asserts an error flag on the `rx_errdetect` port. If the received 10-bit code is detected with incorrect running disparity, the 8B/10B decoder block asserts an error flag on the `rx_disperr` and `rx_errdetect` ports. The error flag signals (`rx_errdetect` and `rx_disperr`) have the same data path delay from the 8B/10B decoder to the PLD-transceiver interface as the bad code group.

Receiver State Machine

The receiver state machine operates in Basic, GIGE, PCI Express (PIPE), and XAUI modes. In GIGE mode, the receiver state machine replaces invalid code groups with K30.7. In XAUI mode, the receiver state machine translates the XAUI PCS code group to the XAUI XGMII code group.