V-Series Transceiver PHY IP Core User Guide

Subscribe

Send Feedback

UG-01080
2020.06.02

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

ALTERA® now part of Intel

Contents

Introduction to the Protocol-Specific and Native Transceiver PHYs

1

2020.06.02

UG-01080

Subscribe Send Feedback

The Arria V, Cyclone V, and Stratix V support three types of transceiver PHY implementations or customization.

The three types of transceiver PHY implementations are the following:

- Protocol-specific PHY
- Non-protocol-specific PHY
- Native transceiver PHY

The protocol-specific transceiver PHYs configure the PMA and PCS to implement a specific protocol. In contrast, the native PHY provides broad access to the low-level hardware, allowing you to configure the transceiver to meet your design requirements. Examples of protocol-specific PHYs include XAUI and Interlaken.

You must also include the reconfiguration and reset controllers when you implement a transceiver PHY in your design.

Protocol-Specific Transceiver PHYs

The protocol-specific transceiver PHYs configure many PCS to meet the requirements of a specific protocol, leaving fewer parameters for you to specify.

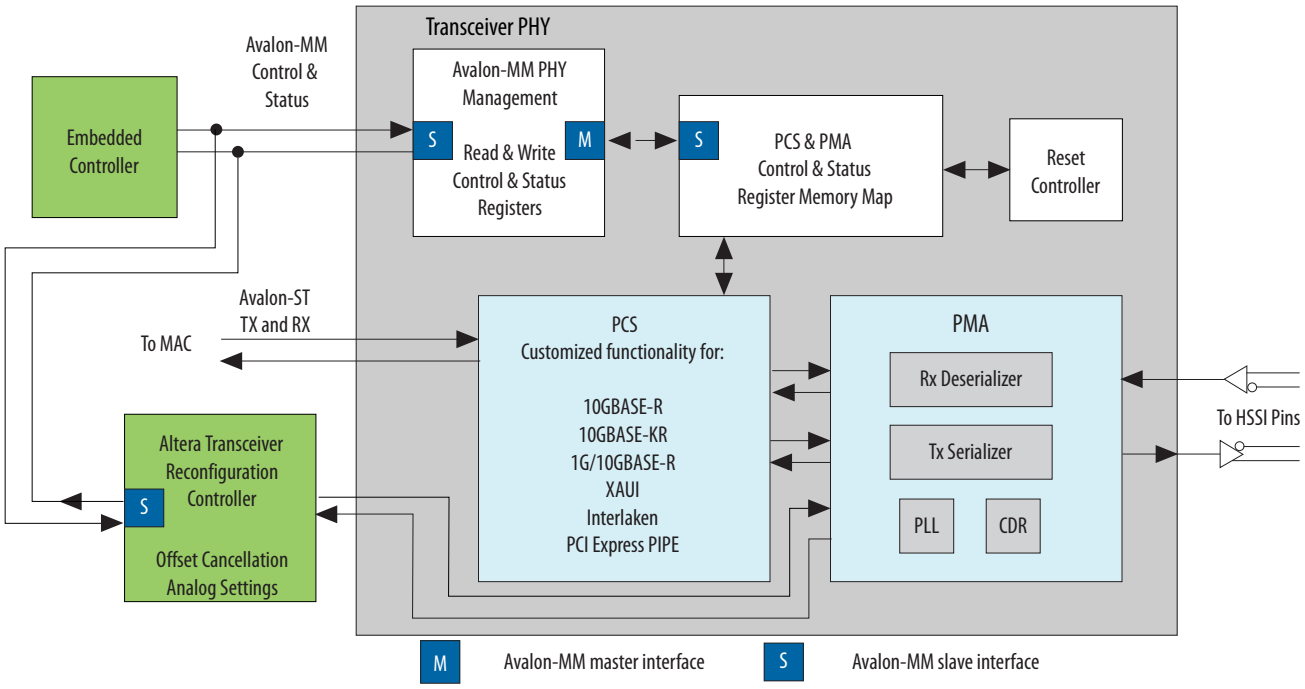
Altera offers the following protocol-specific transceiver PHYS:

- 1G/10 Gbps Ethernet
- 10GBASE-R
- Backplane Ethernet 10GBASE-KR PHY
- Interlaken
- PHY IP Core for PCI Express (PIPE)
- XAUI

These transceiver PHYs include an Avalon® Memory-Mapped (Avalon-MM) interface to access control and status registers and an Avalon Streaming (Avalon-ST) interface to connect to the MAC for data transfer.

The following figure illustrates the top level modules that comprise the protocol-specific transceiver PHY IP cores. As illustrated, the Altera Transceiver Reconfiguration Controller IP Core is instantiated separately.

Figure 1-1: Transceiver PHY Top-Level Modules

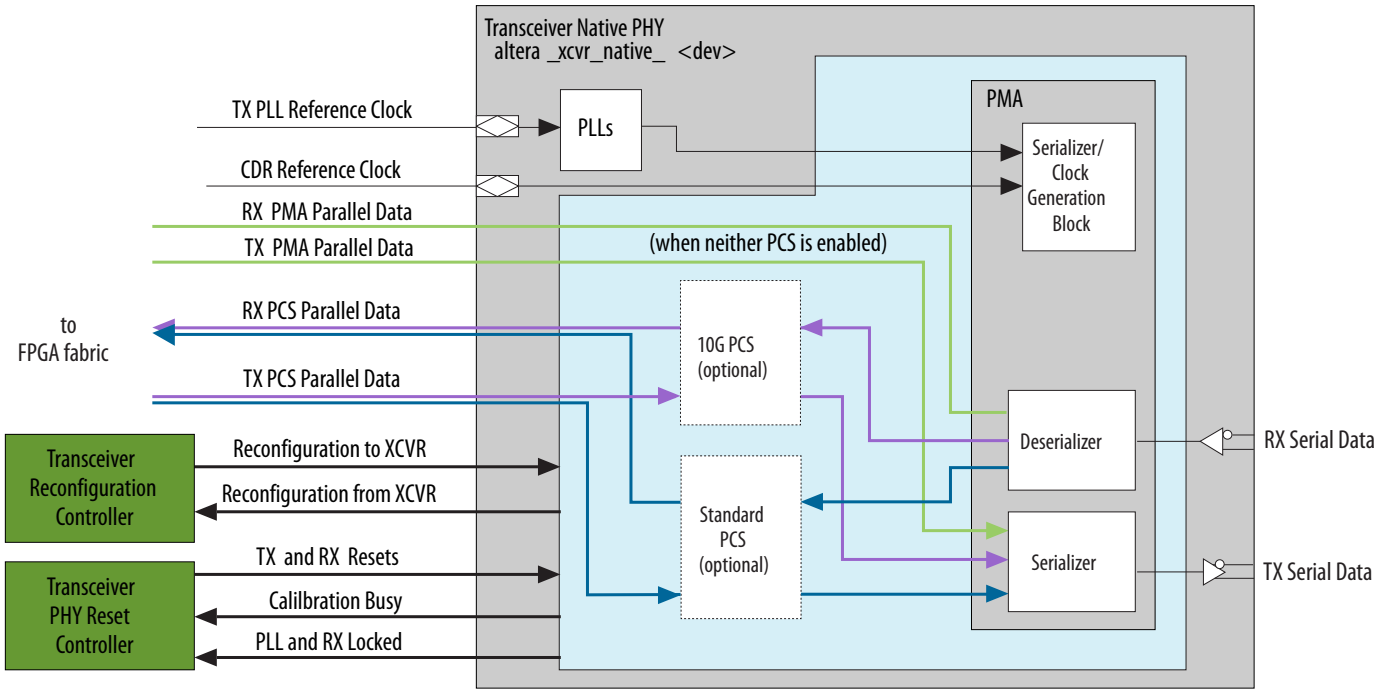


Native Transceiver PHYs

Each device family, beginning with Series V devices offers a separate Native PHY IP core to provide low-level access to the hardware. There are separate IP Cores for Arria V, Arria V GZ, Cyclone V, and Stratix V devices.

The Native PHYs allow you to customize the transceiver settings to meet your requirements. You can also use the Native PHYs to dynamically reconfigure the PCS datapath. Depending on protocol mode selected, built-in rules validate the options you specify. The following figure illustrates the Stratix V Native PHY.

Figure 1-2: Stratix V Transceiver Native PHY IP Core



As shown, the Stratix V Native PHY connects to the separately instantiated Transceiver Reconfiguration Controller and Transceiver PHY Reset Controller.

Table 1-1: Native Transceiver PHY Datapaths

Datapaths	Stratix V	Arria V	Arria V GZ	Cyclone V
PMA Direct: This datapath connects the FPGA fabric directly to the PMA, minimizing latency. You must implement any required PCS functions in the FPGA fabric. (1)	Yes	Yes	Yes	-

(1) PMA Direct mode is supported for Arria V GT, ST, and GZ devices, and for Stratix V GT devices only.

Datapaths	Stratix V	Arria V	Arria V GZ	Cyclone V
Standard: This datapath provides a complete PCS and PMA for the TX and RX channels. You can customize the Standard datapath by enabling or disabling individual modules and specifying data widths.	Yes	Yes	Yes	Yes
10G: This is a high performance datapath. It provides a complete PCS and PMA for the TX and RX channels. You can customize the 10G datapath by enabling or disabling individual modules and specifying data widths.	Yes	-	Yes	-

Related Information

- **Analog Settings for Arria V Devices** on page 20-2
- **Analog Settings for Arria V GZ Devices** on page 20-11
- **Analog Settings for Cyclone V Devices** on page 20-26
- **Analog Settings for Stratix V Devices** on page 20-35

Non-Protocol-Specific Transceiver PHYs

Non-protocol specific transceiver PHYs provide more flexible settings than the protocol-specific transceiver PHYs. They include the Custom PHY, Low Latency PHY, and Deterministic Latency PHY IP Cores.

These PHYs include an Avalon® Memory-Mapped (Avalon-MM) interface to access control and status registers and an Avalon Streaming (Avalon-ST) interface to connect to the MAC for data transfer.

Related Information

- **Custom PHY IP Core** on page 10-1
- **Deterministic Latency PHY IP Core** on page 12-1
- **Low Latency PHY IP Core** on page 11-1

Transceiver PHY Modules

The following sections provide a brief introduction to the modules included in the transceiver PHYs.

PCS

The PCS implements part of the physical layer specification for networking protocols. Depending upon the protocol that you choose, the PCS may include many different functions. Some of the most commonly included functions are: 8B/10B, 64B/66B, or 64B/67B encoding and decoding, rate matching and clock compensation, scrambling and descrambling, word alignment, phase compensation, error monitoring, and gearbox.

PMA

The PMA receives and transmits differential serial data on the device external pins. The transmit (TX) channel supports programmable pre-emphasis and programmable output differential voltage (VOD). It converts parallel input data streams to serial data. The receive (RX) channel supports offset cancellation to correct for process variation and programmable equalization. It converts serial data to parallel data for processing in the PCS. The PMA also includes a clock data recovery (CDR) module with separate CDR logic for each RX channel.

Avalon-MM PHY Management Interface

You can use the Avalon-MM PHY Management module to read and write the control and status registers in the PCS and PMA for the protocol-specific transceiver PHY. The Avalon-MM PHY Management module includes both Avalon-MM master and slave ports and acts as a bridge. It transfers commands received from an embedded controller on its slave port to its master port. The Avalon-MM PHY management master interface connects the Avalon-MM slave ports of PCS and PMA registers and the Transceiver Reconfiguration module, allowing you to manage these Avalon-MM slave components through a simple, standard interface. (Refer to Transceiver PHY Top-Level Modules.)

Transceiver Reconfiguration Controller

Altera Transceiver Reconfiguration Controller dynamically reconfigures analog settings in Arria V, Cyclone V, and Stratix V devices.

Reconfiguration allows you to compensate for variations due to process, voltage, and temperature (PVT) in 28-nm devices. It is required for Arria V, Cyclone V, and Stratix V devices that include transceivers. For more information about the Transceiver Reconfiguration Controller, refer to Transceiver Reconfiguration Controller IP Core. The reset controller may be included in the transceiver PHY or may be a separately instantiated component as described in Transceiver PHY Reset Controller.

Related Information

Transceiver Reconfiguration Controller IP Core Overview on page 17-1

Resetting the Transceiver PHY

This section provides an overview of the embedded reset controller and the separately instantiated Transceiver PHY Reset Controller IP Core.

The embedded reset controller ensures reliable transceiver link initialization. The reset controller initializes both the TX and RX channels. You can disable the automatic reset controller in the Custom, Low Latency Transceiver, and Deterministic Latency PHYs. If you disable the embedded reset controller, the powerdown, analog and digital reset signals for both the TX and RX channels are top-level ports of the transceiver PHY. You can use these ports to design a custom reset sequence, or you can use the Altera-provided Transceiver Reset Controller IP Core.

The Transceiver PHY Reset Controller IP Core handles all reset sequencing of the transceiver to enable successful operation. Because the Transceiver PHY Reset Controller IP is available in clear text, you can also modify it to meet your requirements. For more information about the Transceiver PHY Reset Controller, refer to Transceiver Reconfiguration Controller IP Core.

To accommodate different reset requirements for different transceivers in your design, instantiate multiple instances of a PHY IP core. For example, if your design includes 20 channels of the Custom PHY IP core with 12 channels running a custom protocol using the automatic reset controller and 8 channels requiring manual control of RX reset, instantiate 2 instances of the Custom PHY IP core and customize one to use automatic mode and the other to use your own reset logic. For more information, refer to "Enable embedded reset control" in Custom PHY General Options.

For more information about reset control in Stratix V devices, refer to *Transceiver Reset Control in Stratix V Devices* in volume 3 of the *Stratix V Device Handbook*. For Stratix IV devices, refer to *Reset Control and Power Down* in volume 4 of the *Stratix IV Device Handbook*. For Arria V devices, refer to *Transceiver Reset Control and Power-Down in Arria V Devices*. For Cyclone V devices refer to *Transceiver Reset Control and Power Down in Cyclone V Devices*.

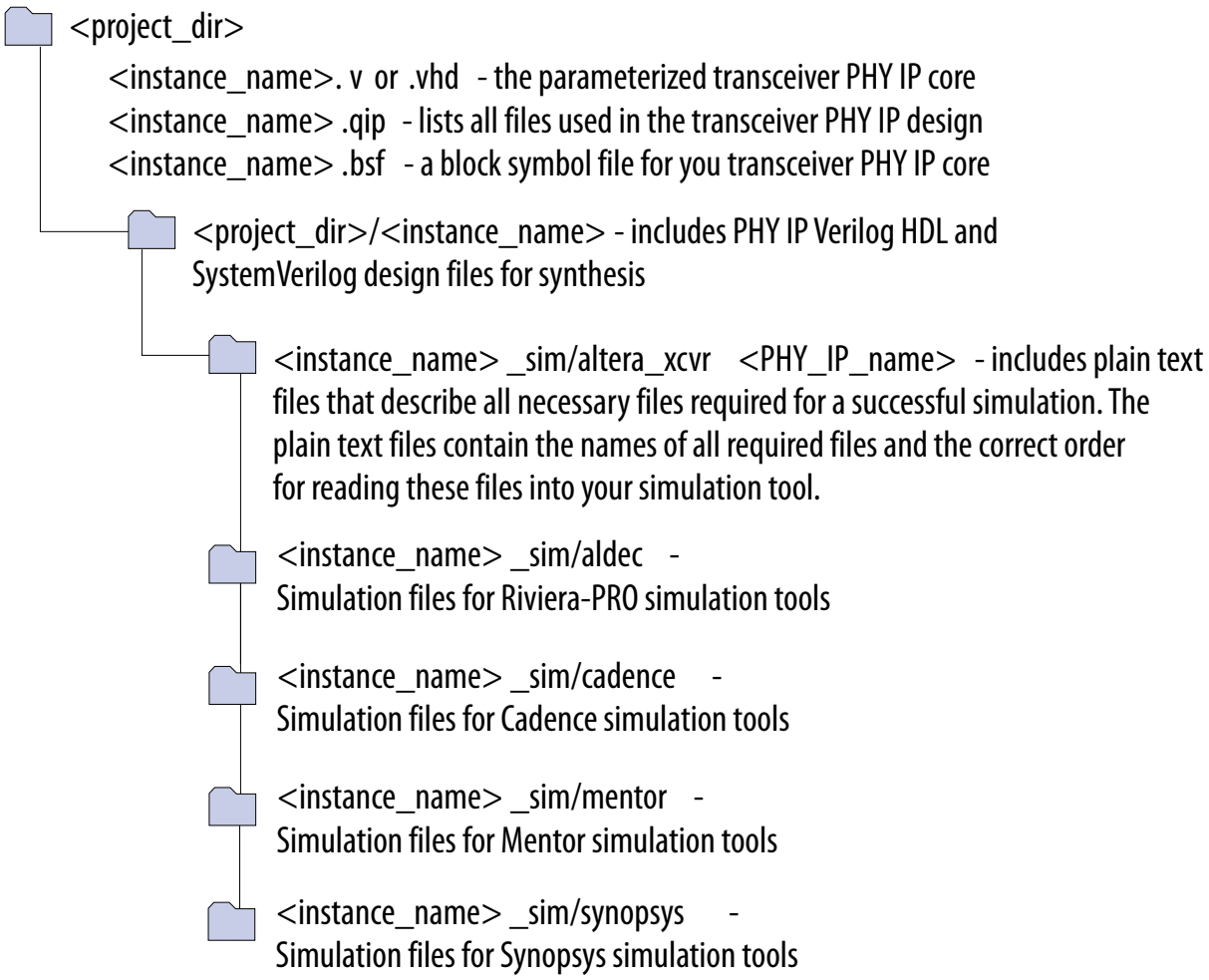
Related Information

- **General Options Parameters** on page 10-3
- **Transceiver PHY Reset Controller IP Core** on page 18-1
- **Transceiver Reset Control in Stratix V Devices**
- **Reset Control and Power Down**
- **Transceiver Reset Control and Power-Down in Arria V Devices**
- **Transceiver Reset Control and Power Down in Cyclone V Devices**

Running a Simulation Testbench

When you generate your transceiver PHY IP core, the Intel® Quartus® Prime software generates the HDL files that define your parameterized IP core. In addition, the Intel Quartus Prime software generates an example Tcl script to compile and simulate your design in ModelSim.

Figure 1-3: Directory Structure for Generated Files



The following table describes the key files and directories for the parameterized transceiver PHY IP core and the simulation environment which are in clear text.

Table 1-2: Transceiver PHY Files and Directories

File Name	Description
<project_dir>	The top-level project directory.
<instance_name> **.v** or **.vhd**	The top-level design file.
<instance_name> **.qip**	A list of all files necessary for Intel Quartus Prime compilation.
<instance_name> **.bsf**	A Block Symbol File (.bsf) for your transceiver PHY.
<project_dir>/<instance_name>/	The directory that stores the HDL files that define the protocol-specific PHY IP core. These files are used for synthesis.

File Name	Description
sv_xcvr_native.sv	Defines the transceiver. It includes instantiations of the PCS and PMA modules and Avalon-MM PHY management interface.
stratixv_hssi_ <*module_name*> **_rbc. sv**	These files perform rule based checking for the module specified. For example, if the PLL type, data rate, and FPGA fabric transceiver interface width are not compatible, the checker reports an error.
altera_wait_generate.v	Generates waitrequest for protocol-specific transceiver PHY IP core that includes backpressure.
alt_reset_ctrl_tgx_cdrauto.sv	Includes the reset controller logic.
<*instance_name*> **_phy_assignments.qip**	Includes an example of the PLL_TYPE assignment statement required to specify the PLL type for each PLL in the design. The available types are clock multiplier unit (CMU) and auxiliary transmit (ATX).
<*project_dir*>/<*instance_name*> **_sim/ altera_xcvr_** <*PHY_IP_name*>/	The simulation directory.
<*project_dir*>/<*instance_name*>*_sim/* **aldec**	Simulation files for Riviera-PRO simulation tools.
<*project_dir*>/<*instance_name*>*_sim/* **cadence**	Simulation files for Cadence simulation tools.
<*project_dir*>/<*instance_name*>*_sim/* **mentor**	Simulation files for Mentor simulation tools.
<*project_dir*>/<*instance_name*>*_sim/* **synopsys**	Simulation files for Synopsys simulation tools.

The Verilog and VHDL transceiver PHY IP cores have been tested with the following simulators:

- ModelSim SE
- Synopsys VCS MX
- Cadence NCSim

If you select VHDL for your transceiver PHY, only the wrapper generated by the Intel Quartus Prime software is in VHDL. All the underlying files are written in Verilog or System Verilog.

For more information about simulating with ModelSim, refer to the *Mentor Graphics ModelSim and QuestaSim Support* chapter of the *Intel Quartus Prime Handbook*.

The transceiver PHY IP cores do not support the NativeLink feature in the Intel Quartus Prime software.

Generating Custom Simulation Scripts for Multiple Transceiver PHYs with ip-make-simscript

Use the `ip-make-simscript` utility to generate simulation command scripts for multiple transceiver PHYs or Qsys systems. Specify all Simulation Package Descriptor files (**.spd**). The **.spd** files list the required simulation files for the corresponding IP core. The MegaWizard Plug-In Manager and Qsys generate the **.spd** files.

When you specify multiple **.spd** files, the `ip-make-simscript` utility generates a single simulation script containing all required simulation information. The default value of `TOP_LEVEL_NAME` is the

TOP_LEVEL_NAME defined in the IP core or Qsys **.spd** file. If this is not the top-level instance in your design, specify the top-level instance of your testbench or design.

You can set appropriate variables in the script or edit the variable assignments directly in the script. If the simulation script is a Tcl file that can be sourced in the simulator, set the variables before sourcing the script. If the simulation script is a shell script, pass in the variables as command-line arguments to shell script.

To run `ip-make-simscript` , type the following at the command prompt:

```
<ACDS installation path>\quartus\sopc_builder\bin\ip-make-simscript
```

The following tables lists some of the options available with this utility.

Table 1-3: Options for the `ip-make-simscript` Utility

Option	Description	Status
`--spd=<file>`	Describes the list of compiled files and memory model hierarchy. If your design includes multiple IP cores or Qsys systems that include **.spd** files, use this option for each file. For example: `ip-make-simscript --spd=ip1.spd --spd=ip2.spd`	Required
`--output-directory=<directory>`	Directory path specifying the location of output files. If unspecified, the default setting is the directory from which `ip-make-simscript` is run.	Optional
`--compile-to-work`	Compiles all design files to the default work library. Use this option only if you encounter problems managing your simulation with multiple libraries.	Optional
`--use-relative-paths`	Uses relative paths whenever possible	Optional

To learn about all options for the `ip-make-simscript` , type the following at the command prompt:

```
<ACDS installation path>\quartus\sopc_builder\bin\ip-make-simscript --help
```

Related Information

- **Mentor Graphics ModelSim Support**
- **Simulating Altera Designs**

Unsupported Features

The protocol-specific and native transceiver PHYs are not supported in Qsys in the current release.

Getting Started Overview

2

2020.06.02

UG-01080

Subscribe Send Feedback

This chapter provides a general overview of the Altera IP core design flow to help you quickly get started with any Altera IP core.

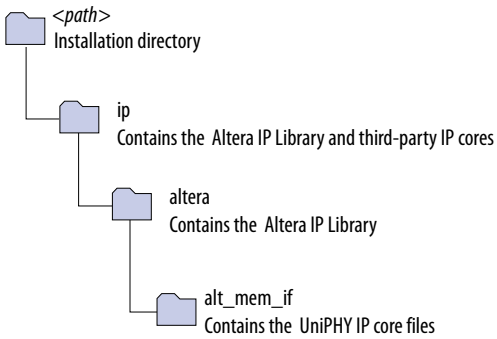
The Altera IP Library is installed as part of the Intel Quartus Prime installation process. You can select and parameterize any Altera IP core from the library. Altera provides an integrated parameter editor that allows you to customize IP cores to support a wide variety of applications. The parameter editor guides you through the setting of parameter values and selection of optional ports. The following sections describe the general design flow and use of Altera IP cores.

Installation and Licensing of IP Cores

The Altera IP Library is distributed with the Intel Quartus Prime software and downloadable from the Altera website.

The following figure shows the directory structure after you install an Altera IP core, where <*path*> is the installation directory. The default installation directory on Windows is **C:\altera\<version number>**; on Linux it is **/opt/altera<version number>**.

Figure 2-1: IP Core Directory Structure



You can evaluate an IP core in simulation and in hardware until you are satisfied with its functionality and performance. Some IP cores require that you purchase a license for the IP core when you want to take your design to production. After you purchase a license for an Altera IP core, you can request a license file from

the Altera Licensing page of the Altera website and install the license on your computer. For additional information, refer to *Altera Software Installation and Licensing*.

Related Information

- **Altera**
- **Altera Licensing**
- **Altera Software Installation and Licensing**

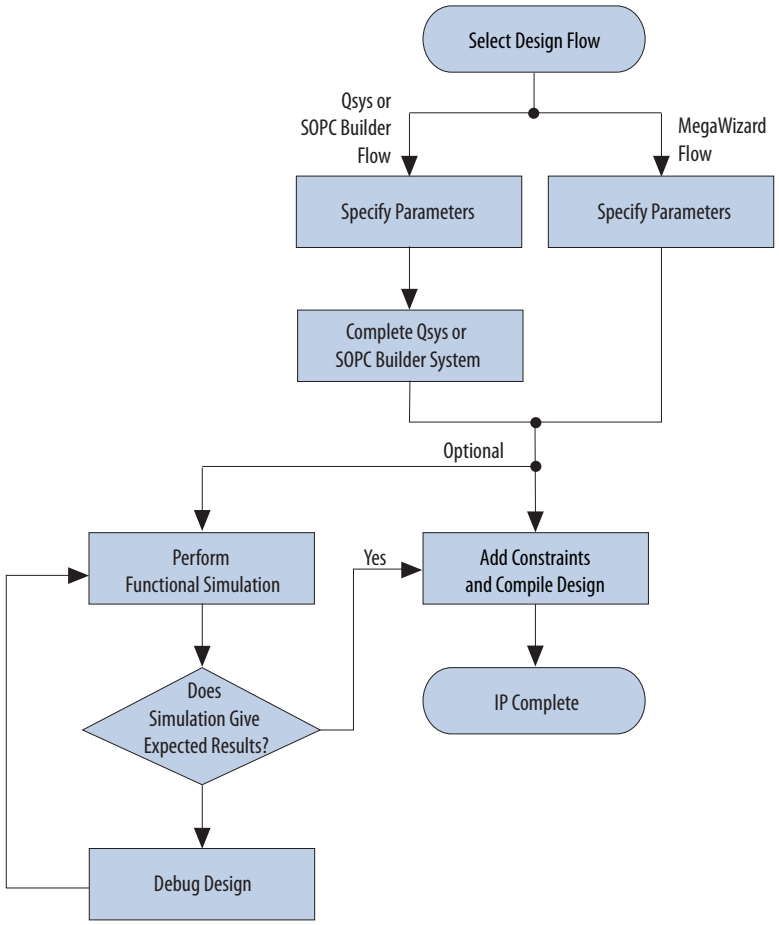
Design Flows

This section describes how to parameterize Altera IP cores.

You can use the following flow(s) to parameterize Altera IP cores:

Figure 2-2: Design Flows

(2)



(2) Altera IP cores may or may not support the Qsys and SOPC Builder design flows.

The MegaWizard Plug-In Manager flow offers the following advantages:

- Allows you to parameterize an IP core variant and instantiate into an existing design
- For some IP cores, this flow generates a complete example design and testbench

MegaWizard Plug-In Manager Flow

This section describes how to specify parameters and simulate your IP core with the MegaWizard Plug-In Manager.

The MegaWizard™ Plug-In Manager flow allows you to customize your IP core and manually integrate the function into your design.

Specifying Parameters

To specify IP core parameters, follow these steps:

1. Create a Intel Quartus Prime project using the **New Project Wizard** available from the File menu.
2. In the Intel Quartus Prime software, launch the **IP Catalog**.
3. You can select the IP core for your protocol implementation from the **IP Catalog**.
4. Specify the parameters on the **Parameter Settings** pages. For detailed explanations of these parameters, refer to the "*Parameter Settings*" chapter in this document or the "*Documentation*" button in the MegaWizard parameter editor.

   Note: Some IP cores provide preset parameters for specific applications. If you wish to use preset parameters, click the arrow to expand the **Presets** list, select the desired preset, and then click **Apply**. To modify preset settings, in a text editor modify the **<installation directory>/ip/altera/alt_mem_if_interfaces/alt_mem_if_<memory_protocol>_emif/alt_mem_if_<memory_protocol>_mem_model.qprs** file.
5. If the IP core provides a simulation model, specify appropriate options in the wizard to generate a simulation model.

   Note:
   - Altera IP supports a variety of simulation models, including simulation-specific IP functional simulation models and encrypted RTL models, and plain text RTL models. These are all cycle-accurate models. The models allow for fast functional simulation of your IP core instance using industry-standard VHDL or Verilog HDL simulators. For some cores, only the plain text RTL model is generated, and you can simulate that model.
   - For more information about functional simulation models for Altera IP cores, refer to *Simulating Altera Designs* in volume 3 of the *Intel Quartus Prime Handbook*.

   Caution: Use the simulation models only for simulation and not for synthesis or any other purposes. Using these models for synthesis creates a nonfunctional design.
6. If the parameter editor includes **EDA** and **Summary** tabs, follow these steps:
   a. Some third-party synthesis tools can use a netlist that contains the structure of an IP core but no detailed logic to optimize timing and performance of the design containing it. To use this feature if your synthesis tool and IP core support it, turn on **Generate netlist**.
   b. On the **Summary** tab, if available, select the files you want to generate. A gray checkmark indicates a file that is automatically generated. All other files are optional.

Note: If file selection is supported for your IP core, after you generate the core, a generation report (**<variation name>.html**)appears in your project directory. This file contains information about the generated files.

7. Click the **Finish** button, the parameter editor generates the top-level HDL code for your IP core, and a simulation directory which includes files for simulation.

   Note: The **Finish** button may be unavailable until all parameterization errors listed in the messages window are corrected.

8. Click **Yes** if you are prompted to add the Intel Quartus Prime IP File (**.qip**) to the current Intel Quartus Prime project. You can also turn on **Automatically add Intel Quartus Prime IP Files to all projects.**

You can now integrate your custom IP core instance in your design, simulate, and compile. While integrating your IP core instance into your design, you must make appropriate pin assignments. You can create a virtual pin to avoid making specific pin assignments for top-level signals while you are simulating and not ready to map the design to hardware.

For some IP cores, the generation process also creates complete example designs. An example design for hardware testing is located in the **< variation_name > _example_design/example_project/** directory. An example design for RTL simulation is located in the **< variation_name > _example_design/simulation/** directory.

Note: For information about the Intel Quartus Prime software, including virtual pins, refer to Intel Quartus Prime Help.

Related Information

- **Simulating Altera Designs**
- **Intel Quartus Prime Help**

Simulate the IP Core

This section describes how to simulate your IP core.

You can simulate your IP core variation with the functional simulation model and the testbench or example design generated with your IP core. The functional simulation model and testbench files are generated in a project subdirectory. This directory may also include scripts to compile and run the testbench.

For a complete list of models or libraries required to simulate your IP core, refer to the scripts provided with the testbench.

For more information about simulating Altera IP cores, refer to *Simulating Altera Designs* in volume 3 of the *Intel Quartus Prime Handbook*.

Related Information

Simulating Altera Designs

10GBASE-R PHY IP Core 3

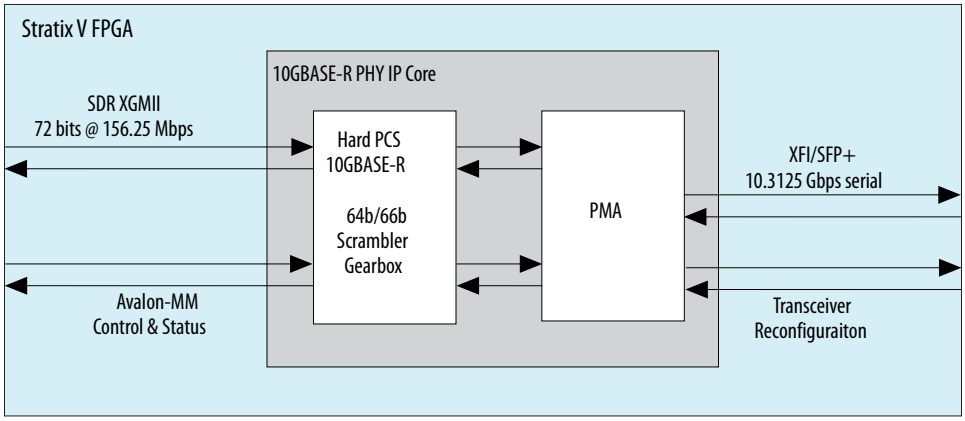
2020.06.02

UG-01080

The Altera 10GBASE-R PHY IP Core implements the functionality described in *IEEE Standard 802.3 Clause 45*.

It delivers serialized data to an optical module that drives optical fiber at a line rate of 10.3125 gigabits per second (Gbps). In a multi-channel implementation of 10GBASE-R, each channel of the 10GBASE-R PHY IP Core operates independently. Both the PCS and PMA of the 10GBASE-R PHY are implemented as hard IP blocks in Stratix V devices, saving FPGA resources.

Figure 3-1: 10GBASE-R PHY with Hard PCS with PMA in Stratix V Devices



Note: For a 10-Gbps Ethernet solution that includes both the Ethernet MAC and the 10GBASE-R PHY, refer to the *10-Gbps Ethernet MAC MegaCore Function User Guide*.

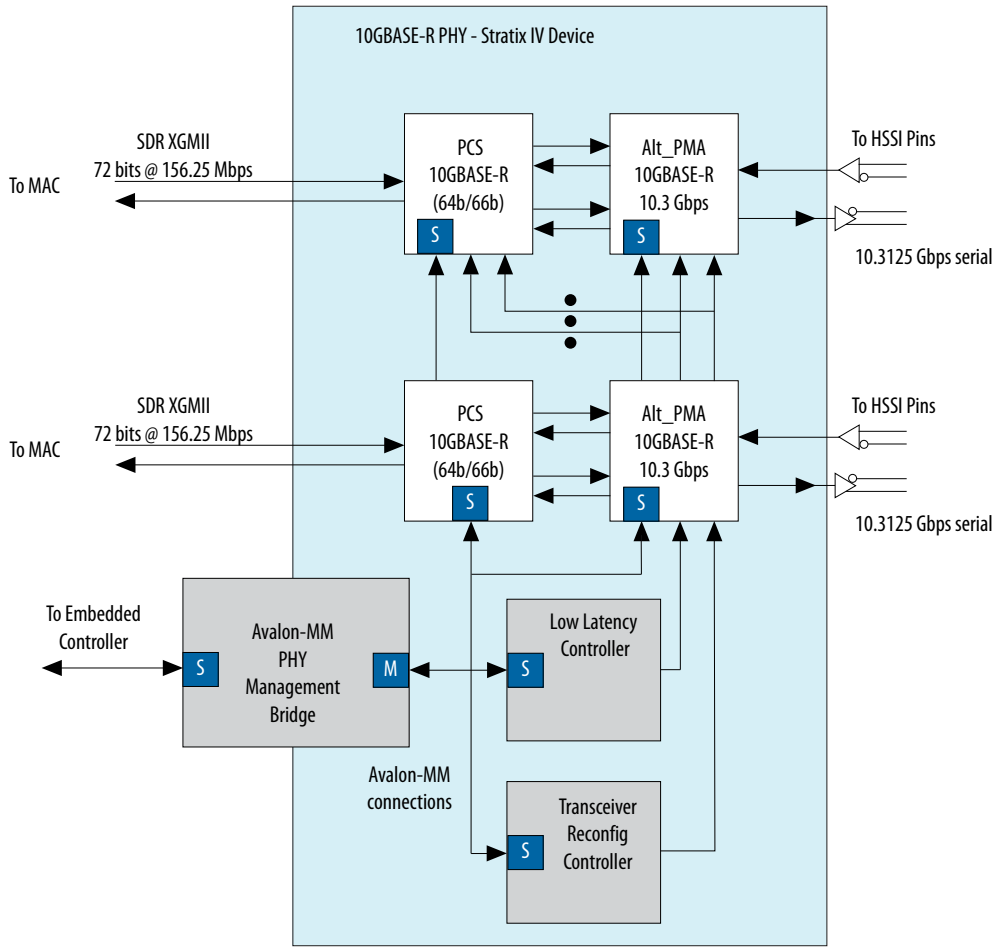
Note: For more detailed information about the 10GBASE-R transceiver channel datapath, clocking, and channel placement, refer to the "*10GBASE-R*" section in the *Transceiver Configurations in Stratix V Devices* chapter of the *Stratix V Device Handbook*.

The following figure illustrates a multiple 10 GbE channel IP core in a Stratix IV GT device. To achieve higher bandwidths, you can instantiate multiple channels. The PCS is available in soft logic for Stratix IV GT devices; it connects to a separately instantiated hard PMA. The PCS connects to an Ethernet MAC via single data rate (SDR) XGMII running at 156.25 megabits per second (Mbps) and transmits data to a 10 Gbps transceiver PMA running at 10.3125 Gbps in a Stratix IV GT device.

To make the most effective use of this soft PCS and PMA configuration for Stratix IV GT devices, you can group up to four channels in a single quad and control their functionality using one Avalon-MM PHY management bridge, transceiver reconfiguration module, and low controller. As this figure illustrates, the Avalon-MM bridge Avalon-MM master port connects to the Avalon-MM slave port of the transceiver reconfiguration and low latency controller modules so that you can update analog settings using the standard Avalon-MM interface.

Note: This configuration does not require that all four channels in a quad run the 10GBASE-R protocol.

Figure 3-2: Complete 10GBASE-R PHY Design in Stratix IV GT Device



The following figures illustrate the 10GBASE-R in Arria V GT, Arria V GZ, and Stratix V GX devices.

Figure 3-3: 10GBASE-R PHY IP Core In Arria V GT Devices

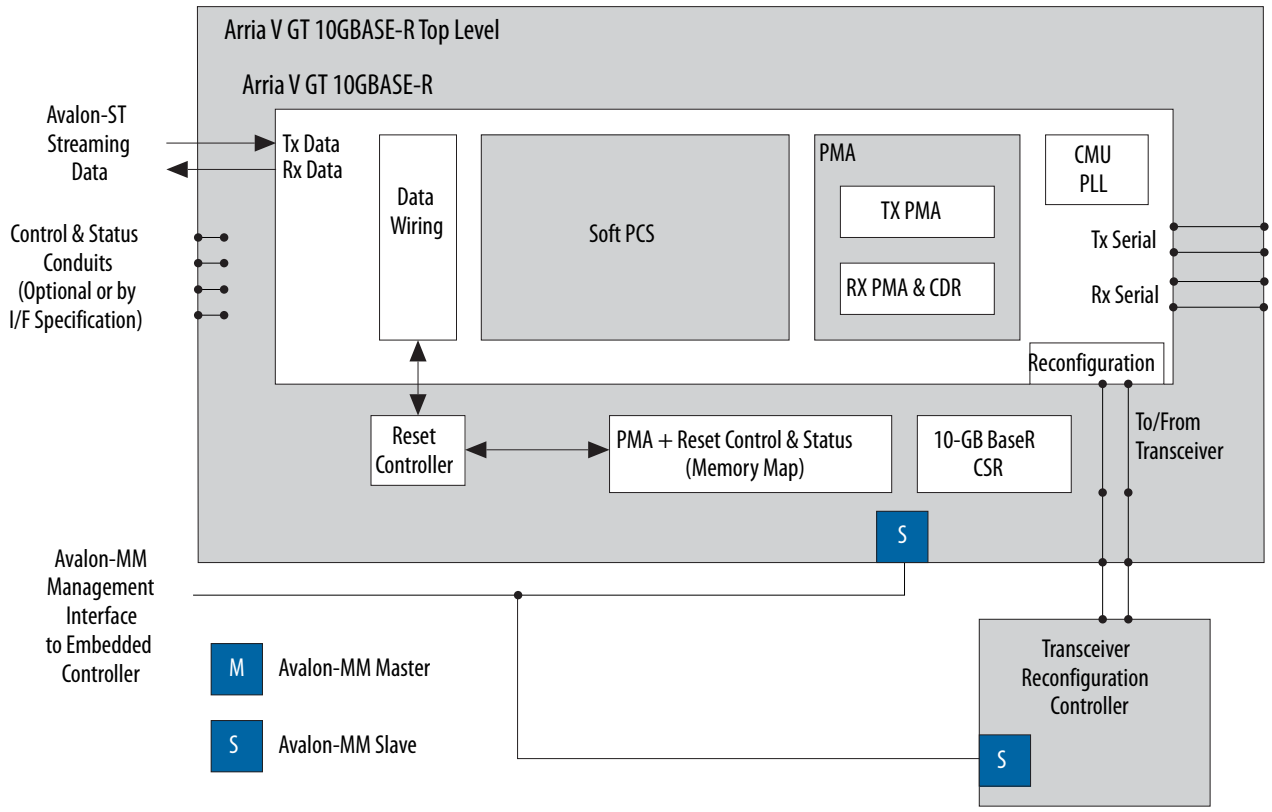


Figure 3-4: 10GBASE-R PHY IP Core In Arria V GZ Devices

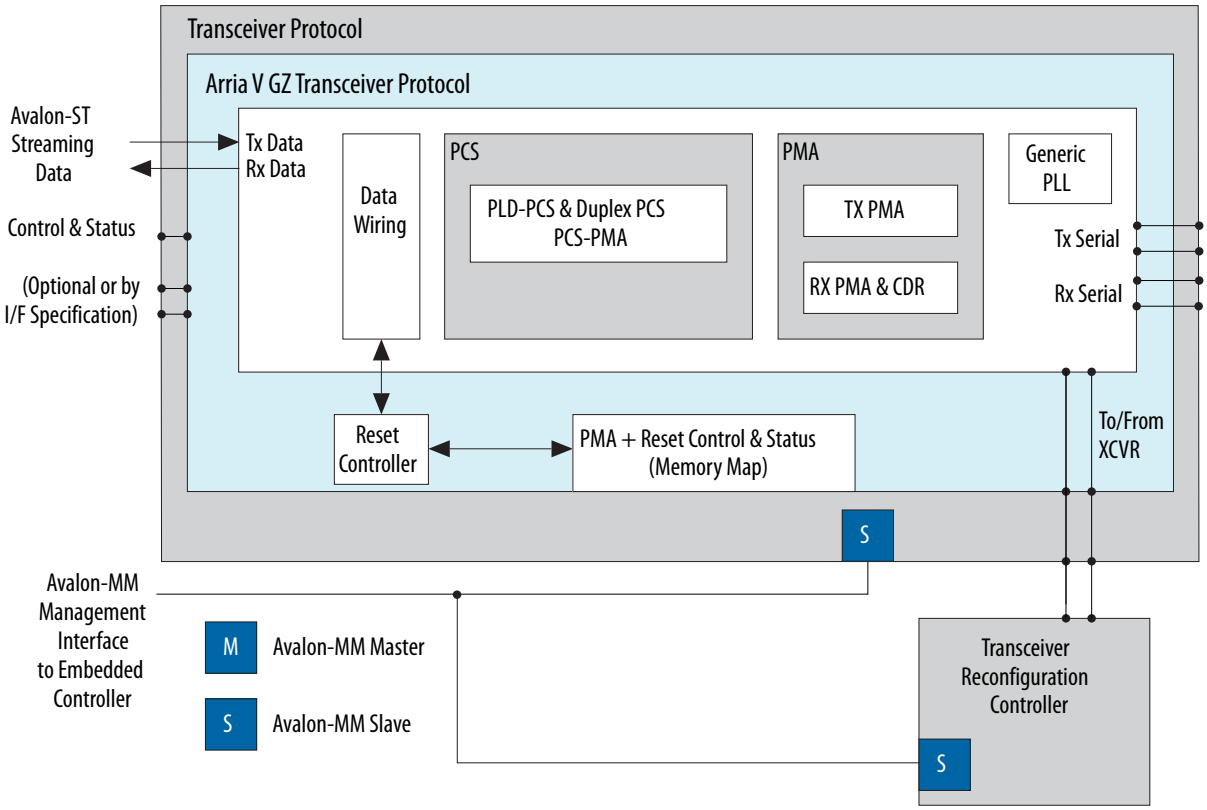
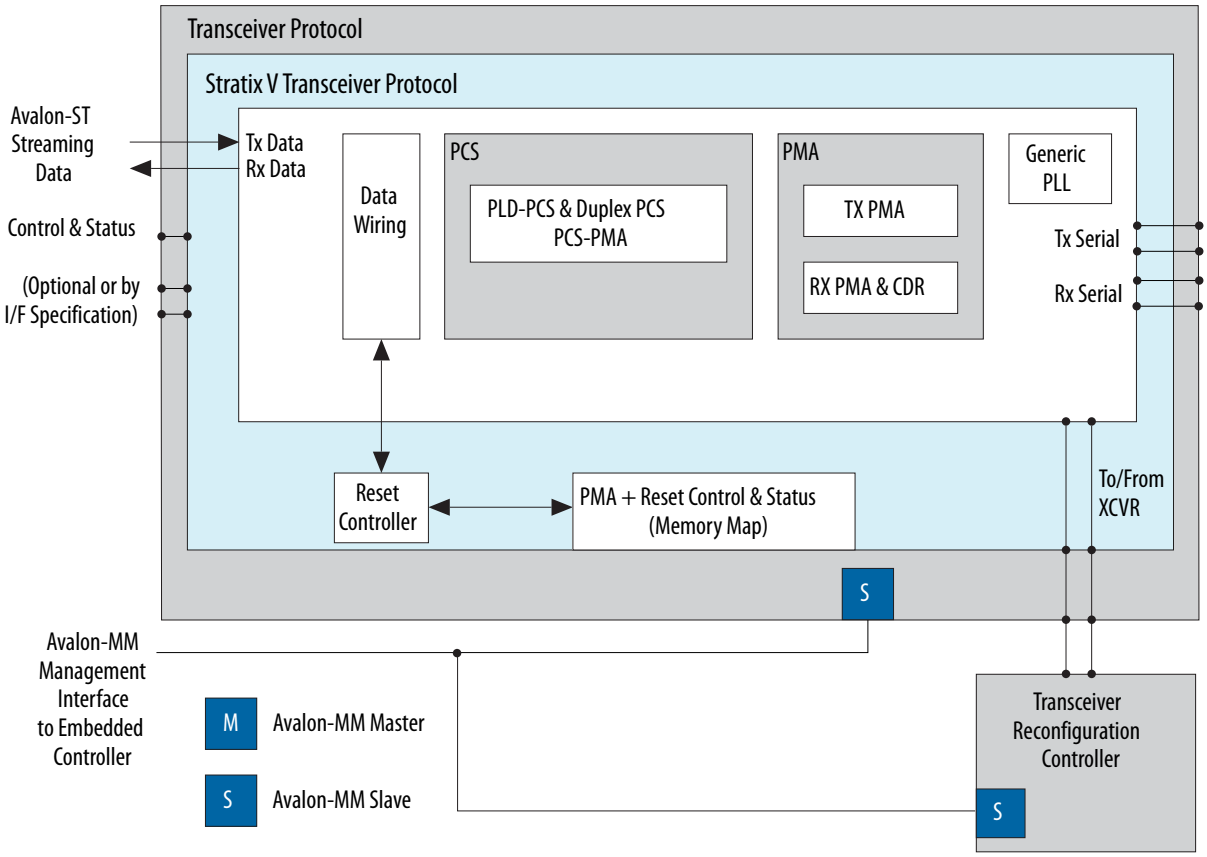


Figure 3-5: 10GBASE-R PHY IP Core In Stratix V Devices



The following table lists the latency through the PCS and PMA for Arria V GT devices with a 66-bit PMA. The FPGA fabric to PCS interface is 64 bits wide. The frequency of the parallel clock is 156.25 MHz which is line rate (10.3125 Gpbs)/interface width (64).

Table 3-1: Latency for TX and RX PCS and PMA Arria V Devices

	PCS (Parallel Clock Cycles	PMA (UI)
TX	28	131
RX	33	99

The following table lists the latency through the PCS and PMA for Stratix V devices with a 40-bit PMA. The FPGA fabric to PCS interface is 64 bits wide. The frequency of the parallel clock is 156.25 MHz which is line rate (10.3125 Gbps)/interface width (64).

Table 3-2: Latency for TX and RX PCS and PMA Stratix V Devices

	PCS (Parallel Clock Cycles)				PMA (UI)
	32-bit PMA Width		40-bit PMA Width		
	Minimum	Maximum	Minimum	Maximum	
TX	7	12	8	12	124
RX	14	33	15	34	43

Related Information

- **IEEE 802.3 Clause 49**
- **10-Gbps Ethernet MAC MegaCore Function User Guide**
- **Transceiver Configurations in Stratix V Devices**

10GBASE-R PHY Release Information

Release information for the IP core.

Table 3-3: 10GBASE-R Release Information

Item	Description
Version	13.1
Release Date	November 2013
Ordering Codes[3]	IP-10GBASERPCS (primary) IPR-10GBASERPCS (renewal)
Product ID	00D2
Vendor ID	6AF7

10GBASE-R PHY Device Family Support

Device support for the IP core.

IP cores provide either final or preliminary support for target Altera device families. These terms have the following definitions:

- *Final support*—Verified with final timing models for this device.
- *Preliminary support*—Verified with preliminary timing models for this device.

[3] No ordering codes or license files are required for Stratix V devices.

Table 3-4: Device Family Support

Device Family	Support
Arria V GT devices–Soft PCS and Hard PMA	Final
Arria V ST devices-Soft PCS and Hard PMA	Final
Arria V GZ	Final
Stratix IV GT devices–Soft PCS and Hard PMA	Final
Stratix V devices–Hard PCS and PMA	Final
Other device families	No support

Note: For speed grade information, refer to "Transceiver Performance Specifications" in the *DC and Switching Characteristics* chapter in the *Stratix IV Handbook* for Stratix IV devices or *Stratix V Device Datasheet*.

Related Information

- **DC and Switching Characteristics**
- **Stratix V Device Datasheet.**

10GBASE-R PHY Performance and Resource Utilization for Stratix IV Devices

Because the 10GBASE-R PHY is implemented in hard logic it uses less than 1% of the available ALMs, memory, primary and secondary logic registers. The following table lists the typical expected device resource utilization for duplex channels using the current version of the Intel Quartus Prime software targeting a Stratix IV GT device. The numbers of combinational ALUTs, logic registers, and memory bits are rounded to the nearest 100.

Table 3-5: 10GBASE-R PHY Performance and Resource Utilization—Stratix IV GT Device

Channels	Combinational ALUTs	Logic Registers (Bits)	Memory Bits
1	5200	4100	4700
4	15600	1300	18800
10	38100	32100	47500

10GBASE-R PHY Performance and Resource Utilization for Arria V GT Devices

The following table lists the resource utilization when targeting an Arria V (5AGTFD7K3F4015) device. Resource utilization numbers reflect changes to the resource utilization reporting starting in the Quartus II software v12.1 release for 28 nm device families and upcoming device families. The numbers of ALMs and logic registers are rounded up to the nearest 100.

Note: For information about Intel Quartus Prime resource utilization reporting, refer to Fitter Resources Reports in the Intel Quartus Prime Help.

Table 3-6: 10GBASE-R PHY Performance and Resource Utilization—Arria V GT Device

Channels	ALMs	Primary Logic Registers	Secondary Logic Registers	Memory 10K
1	2800	3000	300	7

Related Information

Fitter Resources Reports

10GBASE-R PHY Performance and Resource Utilization for Arria V GZ and Stratix V Devices

Because the 10GBASE-R PHY is implemented in hard logic in Arria V GZ and Stratix V devices, it uses less than 1% of the available ALMs, memory, primary and secondary logic registers.

The following table lists the total latency for an Ethernet packet with a 9600 byte payload and an inter-packet gap of 12 characters. The latency includes the number of cycles to transmit the payload from the TX XGMII interface, through the TX PCS and PMA, looping back through the RX PMA and PCS to the RX XGMII interface. (*Stratix V Clock Generation and Distribution* illustrates this datapath.)

Table 3-7: Latency

PPM Difference	Cycles
0 PPM	35
-200 PPM	35
+200 PPM	42

Note: If latency is critical, Altera recommends designing your own soft 10GBASE-R PCS and connecting to the *Low Latency PHY IP Core*.

Parameterizing the 10GBASE-R PHY

The 10GBASE-R PHY IP Core is available for the **Arria V**, **Arria V GZ**, **Stratix IV**, or **Stratix V** device families. Complete the following steps to configure the 10GBASE-R PHY IP Core:

1. Under **Tools** > **IP Catalog**, select the device family of your choice.
2. Under **Tools** > **IP Catalog** > **Interface Protocols** > **Ethernet** > select **10GBASE-R PHY**.
3. Use the tabs on the MegaWizard Plug-In Manager to select the options required for the protocol.
4. Refer to the following topics to learn more about the parameters:
   a. General Option Parameters on page 3-9
   b. Analog Parameters for Stratix IV Devices on page 3-12
5. Click **Finish** to generate your parameterized 10GBASE-R PHY IP Core.

General Option Parameters

This section describes general parameters.

This section describes the 10GBASE-R PHY parameters, which you can set using the MegaWizard Plug-In Manager.

Table 3-8: General Options

Name	Value	Description
General Options		
Device family	**Arria V** **Arria V GZ** **Stratix IV GT** **Stratix V**	Specifies the target device.
Number of channels	`1-32`	The total number of 10GBASE-R PHY channels.
Mode of operation	**Duplex** **TX Only** **RX Only**	Arria V and Stratix V devices allow duplex, TX, or RX mode. Stratix IV GT devices only support duplex mode.
PLL type	**CMU, ATX**	For Arria V GZ, Stratix IV, and Stratix V devices: You can select either the **CMU** or **ATX** PLL. The **CMU** PLL has a larger frequency range than the **ATX** PLL. The ATX PLL is designed to improve jitter performance and achieves lower channel-to-channel skew. Another advantage of the **ATX** PLL is that it does not use a transceiver channel, while the **CMU** PLL does. Altera recommends the ATX PLL for data rates <= 8 Gbps.
Reference Clock Frequency	**322.265625 MHz** **644.53125 MHz**	Arria V and Stratix V devices support both frequencies. Stratix IV GT devices only support **644.53125** MHz.

Name	Value	Description
PCS / PMA interface width	**32** **40**	For Stratix V and Arria V GZ devices only: Specifies the data interface width between the 10G PCS and the transceiver PMA. Smaller width corresponds to lower PCS latency but higher frequency. • For 40 bit width, rx_recovered_clock is 257.8125 MHz and the gearbox ratio is 66:40. • For 40 bit width, rx_recovered_clock is 322.265626 MHz and the gearbox ratio is 66:32. 32 bit PCS / PMA interface with does not support data rates up to 10.3125 Gbps in C4/I4 Arria V GZ device variants. Refer to *Arria V GZ Device Datasheet* for details on data rates supported by different device variants.
Additional Options		
Enable additional control and status pins	**On/Off**	If you turn this option **On**, the following 2 signals are brought out to the top level of the IP core to facilitate debugging: rx_hi_ber and rx_block_lock.
Enable rx_recovered_clk pin	**On/Off**	When you turn this option **On**, the RX recovered clock signal is an output signal.
Enable pll_locked status port	**On/Off**	For Arria V and Stratix V devices: When you turn this option On, a PLL locked status signal is included as a top-level signal of the core.
Use external PMA control and reconfig	**On/Off**	For Stratix IV devices: If you turn this option on, the PMA controller and reconfiguration block are external, rather than included in the 10GBASE-R PHY IP Core, allowing you to use the same PMA controller and reconfiguration IP cores for other protocols in the same transceiver quad. When you turn this option **On**, the `cal_blk_powerdown` (0x021) and `pma_tx_pll_is_locked` (0x022) registers are available.
Enable rx_coreclkin port	**On/Off**	When selected, `rx_coreclkin` is sourced from the 156.25 MHz `xgmii_rx_clk` signal avoiding the use of a FPLL to generate this clock. This clock drives the read side of RX FIFO.

Name	Value	Description
Enable embedded reset control	**On/Off**	When **On**, the automatic reset controller initiates the reset sequence for the transceiver. When **Off** you can design your own reset logic using `tx_analogreset` , `rx_analogreset`, `tx_digitalreset`, `rx_digital-reset`, and `pll_powerdown` which are top-level ports of the Custom Transceiver PHY. You may also use the Transceiver PHY Reset Controller to reset the transceivers. For more information, refer to the *Transceiver Reconfiguration Controller IP Core* . By default, the CDR circuitry is in automatic lock mode whether you use the embedded reset controller or design your own reset logic. You can switch the CDR to manual mode by writing the `pma_rx_setlockto-data` or `pma_rx_set_locktoref` registers to 1. If either the `pma_rx_set_locktodata` and `pma_rx_set_locktoref` is set, the CDR automatic lock mode is disabled.
Starting channel number	**0-96**	For Stratix IV devices, specifies the starting channel number. Must be 0 or a multiple of 4. You only need to set this parameter if you are using external PMA and reconfiguration modules. In Stratix V devices, by default, the logical channel 0 is assigned to either physical transceiver channel 1 or channel 4 of a transceiver bank. However, if you have already created a PCB with a different lane assignment for logical channel 0, you can use the work around shown in the example below. Assignment of the starting channel number is required for serial transceiver dynamic reconfigura-tion.
Enable IEEE 1588 latency adjustment ports	**On/Off**	When you turn this option On, the core includes logic to implement the IEEE 1588 Precision Time Protocol.

Example 3-1: Changing the Default Logical Channel 0 Channel Assignments in Stratix V Devices for ×6 or ×N Bonding

This example shows how to change the default logical channel 0 assignment in Stratix V devices by redefining the pma_bonding_master parameter using the Intel Quartus Prime Assignment Editor. In this example, the pma_bonding_master was originally assigned to physical channel 1. (The original assignment could also have been to physical channel 4.) The to parameter reassigns the

pma_bonding_master to the 10GBASE-R instance name. You must substitute the instance name from your design for the instance name shown in quotation marks.

```
set_parameter -name pma_bonding_master "\"1\"" -to "<PHY IP instance name>"
```

Related Information

- **Transceiver PHY Reset Controller IP Core** on page 18-1
- **1588 Delay Requirements** on page 3-29
- **Arria V GZ Device Datasheet**

Analog Parameters for Stratix IV Devices

For Stratix IV devices, you specify analog options on the **Analog Options** tab.

Table 3-9: PMA Analog Options for Stratix IV Devices

Name	Value	Description
Transmitter termination resistance	**OCT_85_OHMS, OCT_100_OHMS, OCT_120_OHMS, OCT_150_OHMS**	Indicates the value of the termination resistor for the transmitter.
Transmitter V_{OD} control setting	**0–7**	Sets V_{OD} for the various TX buffers.
Pre-emphasis pre-tap setting	**0–7**	Sets the amount of pre-emphasis on the TX buffer.
Invert the pre-emphasis pre-tap polarity setting	**On, Off**	Determines whether or not the pre-emphasis control signal for the pre-tap is inverted. If you turn this option on, the pre-emphasis control signal is inverted.
Pre-emphasis first post-tap setting	**0-15**	Sets the amount of pre-emphasis for the 1st post-tap.
Pre-emphasis second post-tap setting	**0–7**	Sets the amount of pre-emphasis for the 2nd post-tap.
Invert the pre-emphasis second post-tap polarity	**On, Off**	Determines whether or not the pre-emphasis control signal for the second post-tap is inverted. If you turn this option on, the pre-emphasis control signa is inverted.

Name	Value	Description
Receiver common mode voltage	**Tri-State** **0.82V** **1.1v**	Specifies the RX common mode voltage.
Receiver termination resistance	**OCT_85_OHMS** **OCT_100_OHMs** **OCT_120_OHMS** **OCT_150_OHMS**	Indicates the value of the termination resistor for the receiver.
Receiver DC	**0-4**	Sets the equalization DC gain using one of the following settings: • 0: 0 dB • 1: 3 dB • 2: 6 dB • 3: 9 dB • 4: 12 dB
Receiver static equalizer setting:	**0-15**	This option sets the equalizer control settings. The equalizer uses a pass band filter. Specifying a low value passes low frequencies. Specifying a high value passes high frequencies.

Analog Parameters for Arria V, Arria V GZ, and Stratix V Devices

Click on the appropriate links to review the analog parameters for these devices.

Related Information

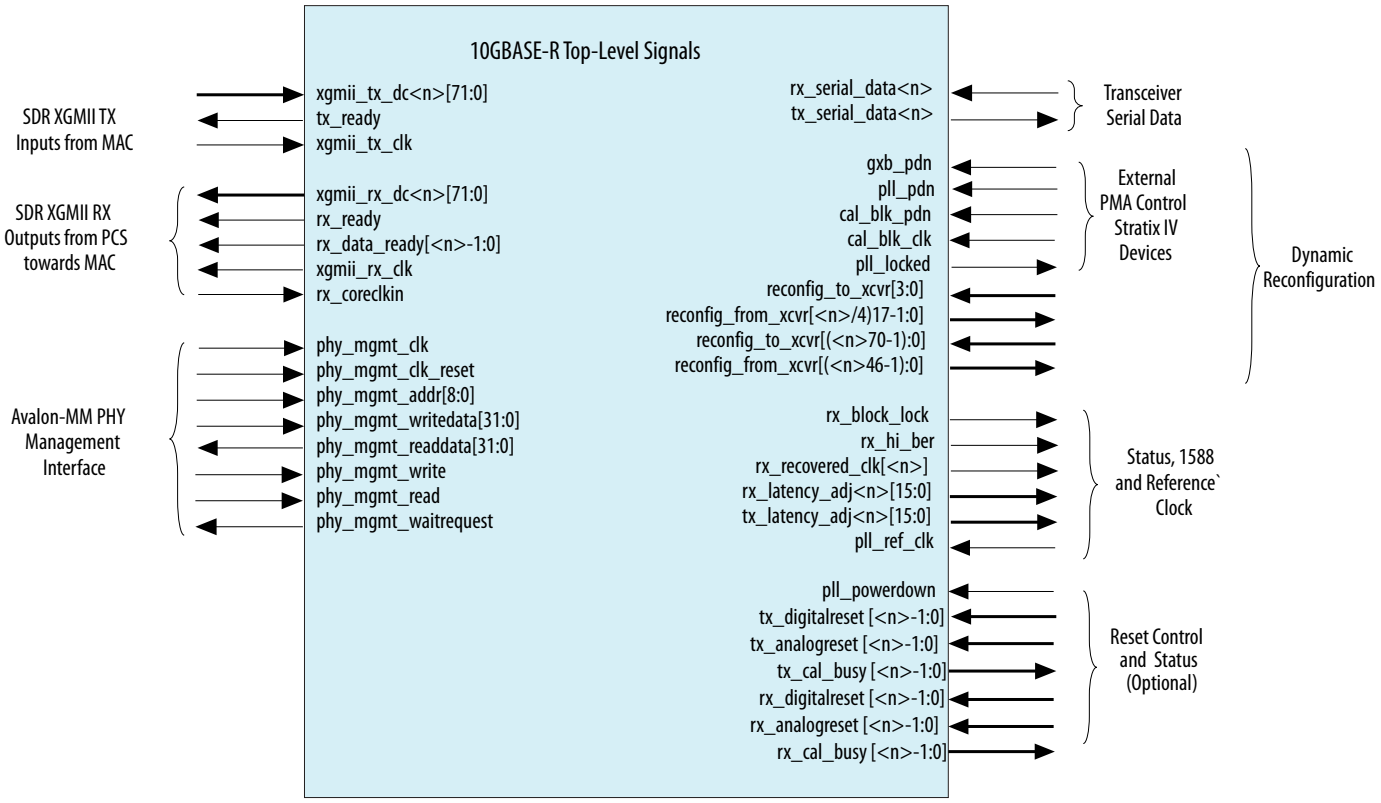
- **Analog Settings for Arria V Devices** on page 20-2
- **Analog Settings for Arria V GZ Devices** on page 20-11
- **Analog Settings for Stratix V Devices** on page 20-35

10GBASE-R PHY Interfaces

This section describes the 10GBASE-R PHY interfaces.

The following figure illustrates the top-level signals of the 10BASE-R PHY; <*n*> is the channel number.

Figure 3-6: 10GBASE-R PHY Top-Level Signals



Note: The **block diagram** shown in the GUI labels the external pins with the interface type and places the interface name inside the box. The interface type and name are used in the Hardware Component Description File (**_hw.tcl**). If you turn on **Show signals**, the **block diagram** displays all top-level signal names.

For more information about **_hw.tcl** files refer to refer to the *Component Interface Tcl Reference* chapter in volume 1 of the *Intel Quartus Prime Handbook*.

Related Information

Component Interface Tcl Reference

10GBASE-R PHY Data Interfaces

This section describes the 10GBASE-R PHY data interfaces.

The TX signals are driven from the MAC to the PCS. The RX signals are driven from the PCS to the MAC.

Table 3-10: SDR XGMII TX Inputs

Signal Name	Direction	Description
XGMII TX Interface		

Signal Name	Direction	Description
xgmii_tx_dc_[<n>71:0]	Input	Contains 8 lanes of data and control for XGMII. Each lane consists of 8 bits of data and 1 bit of control. • Lane 0-[7:0]/[8] • Lane 1-[16:9]/[17] • Lane 2-[25:18]/[26] • Lane 3-[34:27]/[35] • Lane 4-[43:36]/[44] • Lane 5-[52:45]/[53] • Lane 6-[61:54]/[62] • Lane 7-[70:63]/[71] Refer to **Table 3-11** for the mapping of the xgmii_tx_dc data and control to the xgmii_sdr_data and xgmii_sdr_ctrl signals.
tx_ready	Output	Asserted when the TX channel is ready to transmit data. Because the readyLatency on this Avalon-ST interface is 0, the MAC may drive tx_ready as soon as it comes out of reset.
xgmii_tx_clk	Input	The XGMII TX clock which runs at 156.25 MHz. Connect xgmii_tx_clk to xgmii_rx_clk to guarantee this clock is within 150 ppm of the transceiver reference clock.
XGMII RX Interface		
xgmii_rx_dc_<n>[71:0]	Output	Contains 8 lanes of data and control for XGMII. Each lane consists of 8 bits of data and 1 bit of control. • Lane 0-[7:0]/[8] • Lane 1-[16:9]/[17] • Lane 2-[25:18]/[26] • Lane 3-[34:27]/[35] • Lane 4-[43:36]/[44] • Lane 5-[52:45]/[53] • Lane 6-[61:54]/[62] • Lane 7-[70:63]/[71] Refer to **Table 3-12** for the mapping of the xgmii_rx_dc data and control to the xgmii_sdr_data and xgmii_sdr_ctrl signals.
rx_ready	Output	Asserted when the RX reset is complete.

Signal Name	Direction	Description
rx_data_ready [<n>-1:0]	Output	When asserted, indicates that the PCS is sending data to the MAC. Because the readyLatency on this Avalon-ST interface is 0, the MAC must be ready to receive data whenever this signal is asserted. After rx_ready is asserted indicating the exit from the reset state, the MAC should store xgmii_rx_dc_<n>[71:0] in each cycle where rx_data_ready<n> is asserted.
xgmii_rx_clk	Output	This clock is generated by the same reference clock that is used to generate the transceiver clock. Its frequency is 156.25 MHz. Use this clock for the MAC interface to minimize the size of the FIFO between the MAC and SDR XGMII RX interface.
rx_coreclkin	Input	When you turn on Create rx_coreclkin port, this signal is available as a 156.25 MHz clock input port to drive the RX datapath interface (RX read FIFO).
Serial Interface		
rx_serial_data_<n>	Input	Differential high speed serial input data using the PCML I/O standard. The clock is recovered from the serial data stream.
tx_serial_data_<n>	Output	Differential high speed serial input data using the PCML I/O standard. The clock is embedded from the serial data stream.

Table 3-11: Mapping from XGMII TX Bus to XGMII SDR Bus

Signal Name	XGMII Signal Name	Description
xgmii_tx_dc_[7:0]	xgmii_sdr_data[7:0]	Lane 0 data
xgmii_tx_dc_[8]	xgmii_sdr_ctrl[0]	Lane 0 control
xgmii_tx_dc_[16:9]	xgmii_sdr_data[15:8]	Lane 1 data
xgmii_tx_dc_[17]	xgmii_sdr_ctrl[1]	Lane 1 control
xgmii_tx_dc_[25:18]	xgmii_sdr_data[23:16]	Lane 2 data
xgmii_tx_dc_[26]	xgmii_sdr_ctrl[2]	Lane 2 control
xgmii_tx_dc_[34:27]	xgmii_sdr_data[31:24]	Lane 3 data
xgmii_tx_dc_[35]	xgmii_sdr_ctrl[3]	Lane 3 control
xgmii_tx_dc_[43:36]	xgmii_sdr_data[39:32]	Lane 4 data

Signal Name	XGMII Signal Name	Description
xgmii_tx_dc_[44]	xgmii_sdr_ctrl[4]	Lane 4 control
xgmii_tx_dc_[52:45]	xgmii_sdr_data[47:40]	Lane 5 data
xgmii_tx_dc_[53]	xgmii_sdr_ctrl[5]	Lane 5 control
xgmii_tx_dc_[61:54]	xgmii_sdr_data[55:48]	Lane 6 data
xgmii_tx_dc_[62]	xgmii_sdr_ctrl[6]	Lane 6 control
xgmii_tx_dc_[70:63]	xgmii_sdr_data[63:56]	Lane 7 data
xgmii_tx_dc_[71]	xgmii_sdr_ctrl[7]	Lane 7 control

Table 3-12: Mapping from XGMII RX Bus to the XGMII SDR Bus

Signal Name	XGMII Signal Name	Description
xgmii_rx_dc_[7:0]	xgmii_sdr_data[7:0]	Lane 0 data
xgmii_rx_dc_[8]	xgmii_sdr_ctrl[0]	Lane 0 control
xgmii_rx_dc_[16:9]	xgmii_sdr_data[15:8]	Lane 1 data
xgmii_rx_dc_[17]	xgmii_sdr_ctrl[1]	Lane 1 control
xgmii_rx_dc_[25:18]	xgmii_sdr_data[23:16]	Lane 2 data
xgmii_rx_dc_[26]	xgmii_sdr_ctrl[2]	Lane 2 control
xgmii_rx_dc_[34:27]	xgmii_sdr_data[31:24]	Lane 3 data
xgmii_rx_dc_[35]	xgmii_sdr_ctrl[3]	Lane 3 control
xgmii_rx_dc_[43:36]	xgmii_sdr_data[39:32]	Lane 4 data
xgmii_rx_dc_[44]	xgmii_sdr_ctrl[4]	Lane 4 control
xgmii_rx_dc_[52:45]	xgmii_sdr_data[47:40]	Lane 5 data
xgmii_rx_dc_[53]	xgmii_sdr_ctrl[5]	Lane 5 control
xgmii_rx_dc_[61:54]	xgmii_sdr_data[55:48]	Lane 6 data
xgmii_rx_dc_[62]	xgmii_sdr_ctrl[6]	Lane 6 control
xgmii_rx_dc_[70:63]	xgmii_sdr_data[63:56]	Lane 7 data
xgmii_rx_dc_[71]	xgmii_sdr_ctrl[7]	Lane 7 control

10GBASE-R PHY Status, 1588, and PLL Reference Clock Interfaces

This section describes the 10GBASE-R PHY status, 1588, and PLL reference clock interfaces.

Table 3-13: 10GBASE-R Status, 1588, and PLL Reference Clock Outputs

Signal Name	Direction	Description
rx_block_lock	Output	Asserted to indicate that the block synchronizer has established synchronization.
rx_hi_ber	Output	Asserted by the BER monitor block to indicate a Sync Header high bit error rate greater than 10^{-4}.
rx_recovered_clk[<n>:0]	Output	This is the RX clock, which is recovered from the received data stream.
pll_locked	Output	When asserted, indicates that the TX PLL is locked.
IEEE 1588 Precision Time Protocol		
rx_latency_adj_10g [15:0]	Output	When you enable 1588, this signal outputs the real time latency in XGMII clock cycles (156.25 MHz) for the RX PCS and PMA datapath for 10G mode. Bits 0 to 9 represent the fractional number of clock cycles. Bits 10 to 15 represent the number of clock cycles.
tx_latency_adj_10g [15:0]	Output	When you enable 1588, this signal outputs real time latency in XGMII clock cycles (156.25 MHz) for the TX PCS and PMA datapath for 10G mode. Bits 0 to 9 represent the fractional number of clock cycles. Bits 10 to 15 represent the number of clock cycles.
PLL Reference Clock		
pll_ref_clk	Input	For Stratix IV GT devices, the TX PLL reference clock must be 644.53125 MHz. For Arria V and Stratix V devices, the TX PLL reference clock can be either 644.53125 MHz or 322.265625 MHz.

Optional Reset Control and Status Interface

This topic describes the signals in the optional reset control and status interface. These signals are available if you do not enable the embedded reset controller.

Table 3-14: Avalon-ST RX Interface

Signal Name	Direction	Description
pll_powerdown	Input	When asserted, resets the TX PLL.

Signal Name	Direction	Description
tx_digitalreset[<n>-1:0]	Input	When asserted, reset all blocks in the TX PCS. If your design includes bonded TX PCS channels, refer to *Timing Constraints for Reset Signals when Using Bonded PCS Channels for a SDC* constraint you must include in your design.
tx_analogreset[<n>-1:0]	Input	When asserted, resets all blocks in the TX PMA. **Note:** For Arria V devices, while compiling a multi-channel transceiver design, you will see a compile warning (12020) in Intel Quartus Prime software related to the signal width of tx_analogreset. You can safely ignore this warning. Also, per-channel TX analog reset is not supported in Intel Quartus Prime software. Channel 0 TX analog resets all the transceiver channels.
tx_cal_busy[<n>-1:0]	Output	When asserted, indicates that the initial TX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP. You must hold the channel in reset until calibration completes.
rx_digitalreset[<n>-1:0]	Input	When asserted, resets the RX PCS.
rx_analogreset[<n>-1:0]	Input	When asserted, resets the RX CDR.
rx_cal_busy[<n>-1:0]	Output	When asserted, indicates that the initial RX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP.

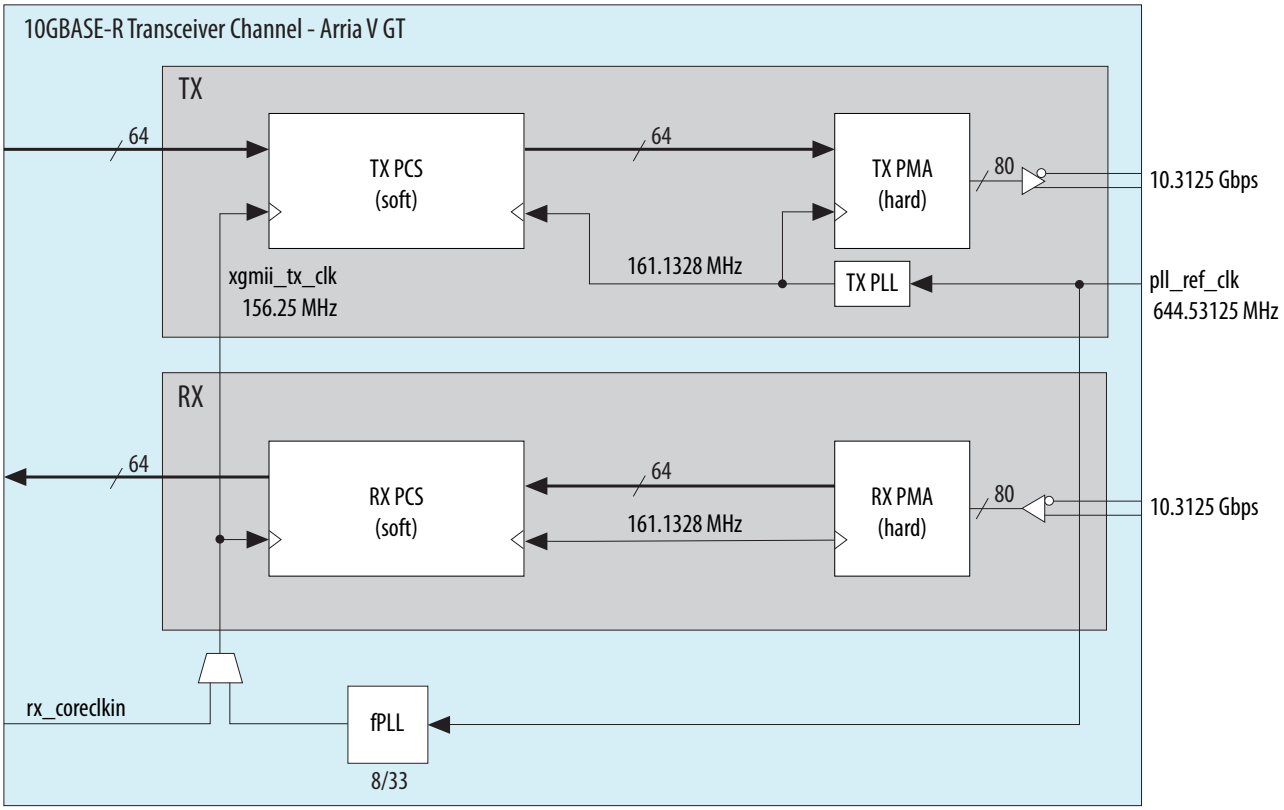
Related Information

- **Timing Constraints for Bonded PCS and PMA Channels** on page 18-11
- **Transceiver Reset Control in Stratix V Devices**
- **Transceiver Reset Control in Arria V Devices**
- **Transceiver Reset Control in Cyclone V Devices**

10GBASE-R PHY Clocks for Arria V GT Devices

The following figure illustrates Arria V GT clock generation and distribution.

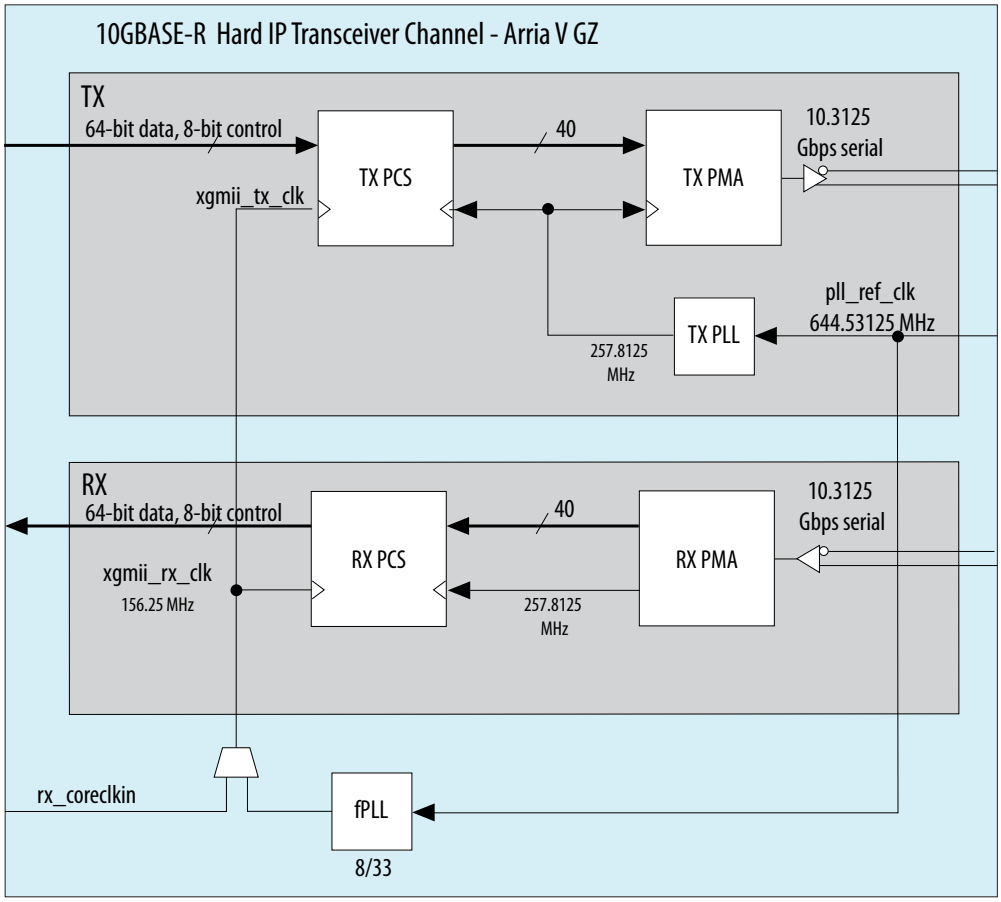
Figure 3-7: Arria V GT Clock Generation and Distribution



10GBASE-R PHY Clocks for Arria V GZ Devices

The following figure illustrates clock generation and distribution for Arria V GZ devices.

Figure 3-8: Arria V GZ Clock Generation and Distribution



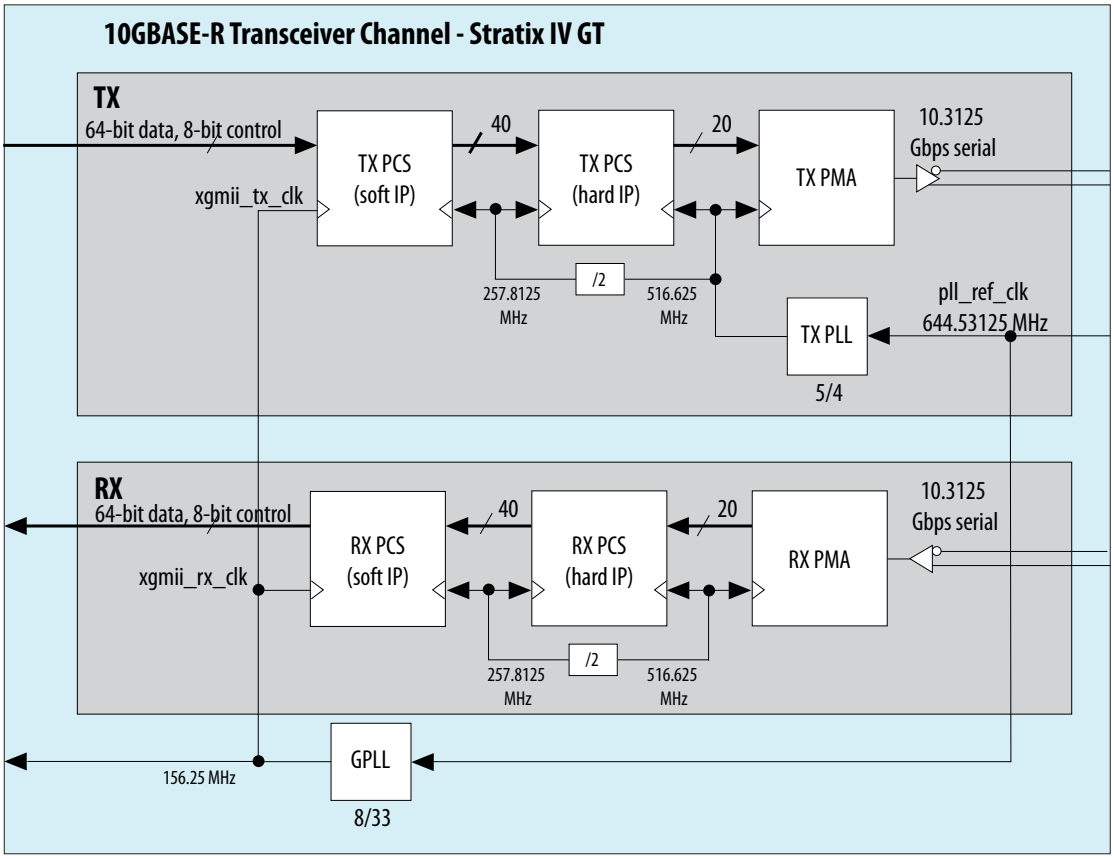
10GBASE-R PHY Clocks for Stratix IV Devices

The `phy_mgmt_clk_reset` signal is the global reset that resets the entire PHY. A positive edge on this signal triggers a reset.

Refer to the *Reset Control and Power Down* chapter in volume 2 of the *Stratix IV Device Handbook* for additional information about reset sequences in Stratix IV devices.

The PCS runs at 257.8125 MHz using the `pma_rx_clock` provided by the PMA. You must provide the PMA an input reference clock running at 644.53725 MHz to generate the 257.8125 MHz clock.

Figure 3-9: Stratix IV Clock Generation and Distribution



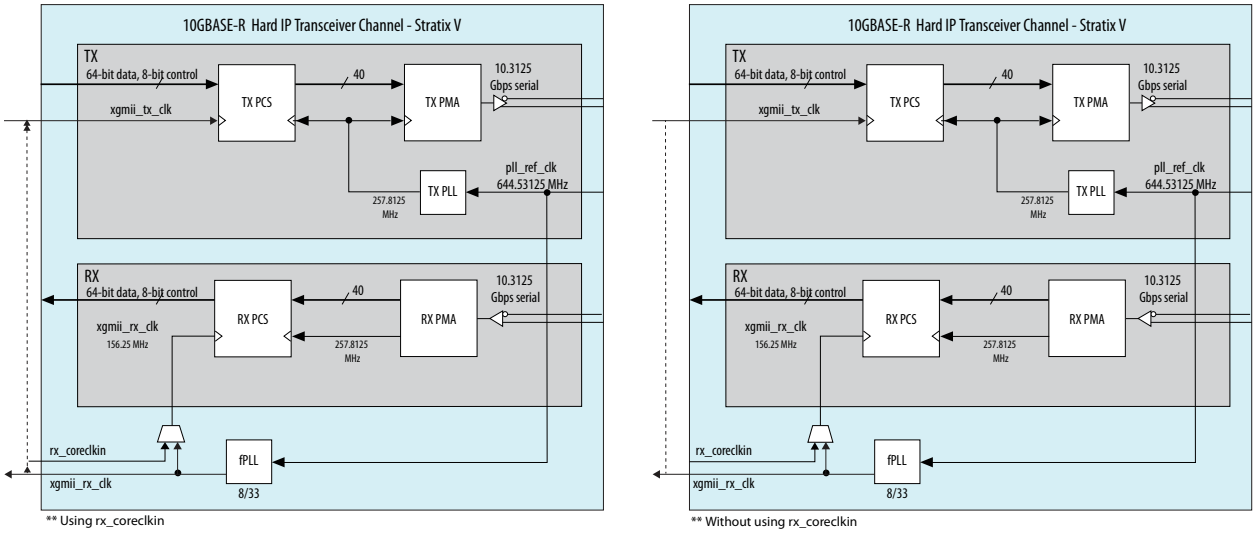
Related Information

Reset Control and Power Down

10GBASE-R PHY Clocks for Stratix V Devices

The following figure illustrates clock generation and distribution in Stratix V devices.

Figure 3-10: Stratix V Clock Generation and Distribution



To ensure proper functioning of the PCS, the maximum PPM difference between the `pll_ref_clk` and `xgmii_tx_clk` clock inputs is 0 PPM. The FIFO in the RX PCS can compensate ±100 PPM between the RX PMA clock and `xgmii_rx_clk`. You should use `xgmii_rx_clk` to drive `xgmii_tx_clk`. The CDR logic recovers 257.8125 MHz clock from the incoming data.

10GBASE-R PHY Register Interface and Register Descriptions

The Avalon-MM PHY management interface provides access to the 10GBASER-R PHY PCS and PMA registers. You can use an embedded controller acting as an Avalon-MM master to send read and write commands to this Avalon-MM slave interface.

Table 3-15: Avalon-MM PHY Management Interface

Signal Name	Direction	Description
`phy_mgmt_clk`	Input	The clock signal that controls the Avalon-MM PHY management, interface. For Stratix IV devices, the frequency range is 37.5-50 MHz. There is no frequency restriction for Stratix V devices; however, if you plan to use the same clock for the PHY management interface and transceiver reconfiguration, you must restrict the frequency range of `phy_mgmt_clk` to 100-150 MHz to meet the specification for the transceiver reconfiguration clock.
`phy_mgmt_clk_reset`	Input	Global reset signal that resets the entire 10GBASE-R PHY. This signal is active high and level sensitive. This signal is not synchronized internally.

Signal Name	Direction	Description
phy_mgmt_addr[8:0]	Input	9-bit Avalon-MM address.
phy_mgmt_writedata[31:0]	Input	Input data.
phy_mgmt_readdata[31:0]	Output	Output data.
phy_mgmt_write	Input	Write signal. Asserted high.
phy_mgmt_read	Input	Read signal. Asserted high.
phy_mgmt_waitrequest	Output	When asserted, indicates that the Avalon-MM slave interface is unable to respond to a read or write request. When asserted, control signals to the Avalon-MM slave interface must remain constant.

Refer to the "*Typical Slave Read and Write Transfers*" and "*Master Transfers*" sections in the "*Avalon Memory-Mapped Interfaces*" chapter of the *Avalon Interface Specifications* for timing diagrams.

The following table specifies the registers that you can access over the Avalon-MM PHY management interface using word addresses and a 32-bit embedded processor. A single address space provides access to all registers.

Note: Writing to reserved or undefined register addresses may have undefined side effects.

Table 3-16: 10GBASE-R Register Descriptions

Word Addr	Bit	R/W	Name	Description
PMA Common Control and Status				
0x021	[31:0]	RW	cal_blk_powerdown	Writing a 1 to channel <*n*> powers down the calibration block for channel <*n*>. This register is only available if you select **Use external PMA control and reconfig** on the **Additional Options** tab of the GUI.
0x022	[31:0]	RO	pma_tx_pll_is_locked	Bit[P] indicates that the TX clock multiplier unit CMU PLL [P] is locked to the input reference clock. This register is only available if you select **Use external PMA control and reconfig** on the **Additional Options** tab of the GUI.
Reset Control Registers-Automatic Reset Controller				
0x041	[31:0]	RW	reset_ch_bitmask	Reset controller channel bitmask for digital resets. The default value is all 1 s. Channel <*n*> can be reset when bit<*n*> = 1. Channel <*n*> cannot be reset when bit<*n*>=0.

Word Addr	Bit	R/W	Name	Description
0x042	[1:0]	WO	reset_control (write)	Writing a 1 to bit 0 initiates a TX digital reset using the reset controller module. The reset affects channels enabled in the reset_ch_bitmask. Writing a 1 to bit 1 initiates a RX digital reset of channels enabled in the reset_ch_bitmask. Both bits 0 and 1 self-clear.
		RO	reset_status (read)	Reading bit 0 returns the status of the reset controller TX ready bit. Reading bit 1 returns the status of the reset controller RX ready bit.
0x044	[31:0]	RW	reset_fine_control	You can use the reset_fine_control register to create your own reset sequence. The reset control module performs a standard reset sequence at power on and whenever the phy_mgmt_clk_reset is asserted. Bits [31:4,0] are reserved.
	[31:4,0]	RW	Reserved	It is safe to write 0s to reserved bits.
	[1]	RW	reset_tx_digital	Writing a 1 causes the internal TX digital reset signal to be asserted, resetting all channels enabled in reset_ch_bitmask. You must write a 0 to clear the reset condition.
	[2]	RW	reset_rx_analog	Writing a 1 causes the internal RX digital reset signal to be asserted, resetting the RX analog logic of all channels enabled in reset_ch_bitmask. You must write a 0 to clear the reset condition.
	[3]	RW	reset_rx_digital	Writing a 1 causes the RX digital reset signal to be asserted, resetting the RX digital channels enabled in reset_ch_bitmask. You must write a 0 to clear the reset condition.
PMA Channel Control and Status				
0x061	[31:0]	RW	phy_serial_loopback	Writing a 1 to channel <*n*> puts channel <*n*> in serial loopback mode. For information about pre- or post-CDR serial loopback modes, refer to Loopback Modes.

Word Addr	Bit	R/W	Name	Description
0x064	[31:0]	RW	pma_rx_set_locktodata	When set, programs the RX CDR PLL to lock to the incoming data. Bit <*n*> corresponds to channel <*n*>.
0x065	[31:0]	RW	pma_rx_set_locktoref	When set, programs the RX CDR PLL to lock to the reference clock. Bit <*n*> corresponds to channel <*n*>.
0x066	[31:0]	RO	pma_rx_is_lockedtodata	When asserted, indicates that the RX CDR PLL is locked to the RX data, and that the RX CDR has changed from LTR to LTD mode. Bit <*n*> corresponds to channel <*n*>.
0x067	[31:0]	RO	pma_rx_is_lockedtoref	When asserted, indicates that the RX CDR PLL is locked to the reference clock. Bit <*n*> corresponds to channel <*n*>.
10GBASE-R PCS				
0x080	[31:0]	WO	INDIRECT_ADDR	Provides for indirect addressing of all PCS control and status registers. Use this register to specify the logical channel number of the PCS channel you want to access.
0x081	[2]	RW	RCLR_ERRBLK_CNT	When set to 1, clears the error block count register. **To block**: Block synchronizer
	[3]	RW	RCLR_BER_COUNT	When set to 1, clears the bit error rate (BER) register. **To block**: BER monitor

Word Addr	Bit	R/W	Name	Description
0x082	[0]	R	PCS_STATUS	For Stratix IV devices: When asserted indicates that the PCS link is up.
	[1]	R	HI_BER	When asserted by the BER monitor block, indicates that the PCS is recording a high BER. **From block**: BER monitor
	[2]	R	BLOCK_LOCK	When asserted by the block synchronizer, indicates that the PCS is locked to received blocks. **From Block**: Block synchronizer
	[3]	R	TX_FIFO_FULL	When asserted, indicates the TX FIFO is full. **From block**: TX FIFO
	[4]	R	RX_FIFO_FULL	When asserted, indicates the RX FIFO is full. **From block**: RX FIFO
	[5]	R	RX_SYNC_HEAD_ERROR	For Stratix V devices, when asserted, indicates an RX synchronization error. This signal is Stratix V devices only.
	[6]	R	RX_SCRAMBLER_ERROR	For Stratix V devices: When asserted, indicates an RX scrambler error.
	[7]	R	RX_DATA_READY	When asserted indicates that the RX interface is ready to send out received data. **From block**: 10 Gbps Receiver PCS
0x083	[5:0]	R	BER_COUNT[5:0]	For Stratix IV devices only, records the bit error rate (BER). **From block**: BER monitor
	[13:6]	R	ERROR_BLOCK_COUNT[7:0]	For Stratix IV devices only, records the number of blocks that contain errors. **From Block**: Block synchronizer
	[14]	R	LATCHED_HI_BER	Latched version of HI_BER . **From block**: BER monitor
	[15]	R	LATCHED_BLOCK_LOCK	Latched version of BLOCK_LOCK. **From Block**: Block synchronizer

Related Information

- **Loopback Modes** on page 17-59
- **Avalon Interface Specifications**

10GBASE-R PHY Dynamic Reconfiguration for Stratix IV Devices

The 10GBASE-R PHY includes additional top-level signals when configured with an external modules for PMA control and dynamic reconfiguration.

You enable this configuration by turning on **Use external PMA control and reconfig** available for Stratix IV GT devices.

Table 3-17: External PMA and Reconfiguration Signals

Signal Name	Direction	Description
gxb_pdn	Input	When asserted, powers down the entire GT block. Active high. For Stratix IV de
pll_pdn	Input	When asserted, powers down the TX PLL. Active high.
cal_blk_pdn	Input	When asserted, powers down the calibration block. Active high.
cal_blk_clk	Input	Calibration clock. For Stratix IV devices only. It must be in the range 37.5-50 MHz. You can use the same clock for the phy_mgmt_clk and the cal_blk_clk.
pll_locked	Output	When asserted, indicates that the TX PLL is locked.
reconfig_to_xcvr[3:0]	Input	Reconfiguration signals from the Transceiver Reconfiguration Controller to the PHY device. This signal is only available in Stratix IV devices.
reconfig_from_xcvr [(<n>/4) 17-1:0]	Output	Reconfiguration RAM. The PHY device drives this RAM data to the transceiver reconfiguration IP. This signal is only available in Stratix IV devices.

10GBASE-R PHY Dynamic Reconfiguration for Arria V and Stratix V Devices

For Arria V and Stratix V devices, each channel and each TX PLL have separate dynamic reconfiguration interfaces. The MegaWizard Plug-In Manager provides informational messages on the connectivity of these interfaces. The example below shows the messages for a single duplex channel.

Although you must initially create a separate reconfiguration interface for each channel and TX PLL in your design, when the Intel Quartus Prime software compiles your design, it reduces the number of reconfiguration interfaces by merging reconfiguration interfaces. The synthesized design typically includes a reconfiguration interface for at least three channels because three channels share an Avalon-MM slave interface which connects to the Transceiver Reconfiguration Controller IP Core. Conversely, you cannot connect the three channels that share an Avalon-MM interface to different Transceiver Reconfiguration Controllers. Doing so causes a Fitter error. For more information, refer to **Transceiver Reconfiguration Controller to PHY IP Connectivity** on page 17-57. Allowing the Intel Quartus Prime software to merge reconfiguration interfaces gives the Fitter more flexibility in placing transceiver channels.

Example 3-2: Informational Messages for the Transceiver Reconfiguration Interface

```
Reconfiguration interface offset 0 is connected to the transceiver channel.

PHY IP will require 2 reconfiguration interfaces for connection to the external
reconfiguration controller.

Reconfiguration interface offset 0 is connected to the transceiver channel.

Reconfiguration interface offset 1 is connected to the transmit PLL.
```

The following table describes the signals in the reconfiguration interface; this interface uses the Avalon-MM PHY Management interface clock.

Table 3-18: Reconfiguration Interface

Signal Name	Direction	Description
`reconfig_to_xcvr [(<n>70-1):0]`	Input	Reconfiguration signals from the Transceiver Reconfiguration Controller. <*n*> grows linearly with the number of reconfiguration interfaces. This signal is only available in Stratix V devices.
`reconfig_from_xcvr [(<n>46-1):0]`	Output	Reconfiguration signals to the Transceiver Reconfiguration Controller. <*n*> grows linearly with the number of reconfiguration interfaces. This signal is only available in Stratix V devices.

1588 Delay Requirements

The 1588 protocol requires symmetric delays or known asymmetric delays for all external connections.

In calculating the delays for all external connections, you must consider the delay contributions of the following elements:

- The PCB traces
- The backplane traces
- The delay through connectors
- The delay through cables

Accurate calculation of the channel-to-channel delay is important in ensuring the overall system accuracy.

10GBASE-R PHY TimeQuest Timing Constraints

The timing constraints for Stratix IV GT designs are in **alt_10gbaser_phy.sdc**. If your design does not meet timing with these constraints, use LogicLock™ for the `alt_10gbaser_pcs block`. You can also apply LogicLock to the `alt_10gbaser_pcs` and slightly expand the lock region to meet timing.

The following example provides the Synopsys Design Constraints file (**.sdc**) timing constraints for the 10GBASE-R IP Core when implemented in a Stratix IV device. To pass timing analysis, you must decouple the clocks in different time domains. Be sure to verify the each clock domain is correctly buffered in the

top level of your design. You can find the **.sdc** file in your top-level working directory. This is the same directory that includes your top-level **.v** or **.vhd** file.

Example 3-3: Synopsys Design Constraints for Clocks

```
#**************************************************************
# Timing Information
#**************************************************************
set_time_format -unit ns -decimal_places 3
#**************************************************************
# Create Clocks
#**************************************************************
create_clock -name {xgmii_tx_clk} -period 6.400 -waveform { 0.000 3.200 }
[get_ports {xgmii_tx_clk}]
create_clock -name {phy_mgmt_clk}   -period 20.00 -waveform { 0.000 10.000 }
[get_ports {phy_mgmt_clk}]
create_clock -name {pll_ref_clk} -period 1.552 -waveform { 0.000 0.776  }
[get_ports {ref_clk}]
#derive_pll_clocks
derive_pll_clocks -create_base_clocks
#derive_clocks -period "1.0"
# Create Generated Clocks
#**************************************************************
create_generated_clock -name pll_mac_clk -source [get_pins -compati-
bility_mode {*altpll_component|auto_generated|pll1|clk[0]}]
create_generated_clock -name pma_tx_clk -source [get_pins -compati-
bility_mode {*siv_alt_pma|pma_direct|auto_generated|transmit_pcs0|clkout}]
**************************************************************
## Set Clock Latency
#**************************************************************
#**************************************************************
# Set Clock Uncertainty
#**************************************************************
#**************************************************************
derive_clock_uncertainty
set_clock_uncertainty -from [get_clocks {*siv_alt_pma|pma_ch*.pma_direct|
receive_pcs*|clkout}] -to pll_ref_clk -setup 0.1
set_clock_uncertainty -from [get_clocks {*siv_alt_pma|pma_direct|
auto_generated|transmit_pcs0|clkout}] -to pll_ref_clk -setup 0.08
set_clock_uncertainty -from [get_clocks {*siv_alt_pma|pma_ch*.pma_direct|
receive_pcs*|clkout}] -to pll_ref_clk -hold 0.1
set_clock_uncertainty -from [get_clocks {*siv_alt_pma|pma_direct|
auto_generated|transmit_pcs0|clkout}] -to pll_ref_clk -hold 0.08
#**************************************************************
# Set Input Delay
#**************************************************************
#**************************************************************
# Set Output Delay
#**************************************************************# Set Clock
Groups
#**************************************************************
set_clock_groups -exclusive -group phy_mgmt_clk -group xgmii_tx_clk -group
[get_clocks {*siv_alt_pma|pma_ch*.pma_direct|transmit_pcs*|clkout}] -group
[get_clocks {*siv_alt_pma|pma_ch*.pma_direct|receive_pcs*|clkout}] -group
[get_clocks {*pll_siv_xgmii_clk|altpll_component|auto_generated|pll1|
clk[0]}]
##**************************************************************
# Set False Path
#**************************************************************
set_false_path -from {*siv_10gbaser_xcvr*clk_reset_ctrl|rx_pma_rstn} -to
[get_clocks {{*siv_alt_pma|pma_ch*.pma_direct|transmit_pcs*|clkout}
{*siv_alt_pma|pma_ch*.pma_direct|receive_pcs*|clkout} {*pll_siv_xgmii_clk|
altpll_component|auto_generated|pll1|clk[0]} phy_mgmt_clk xgmii_tx_clk}]
set_false_path -from {*siv_10gbaser_xcvr*clk_reset_ctrl|rx_usr_rstn} -to
[get_clocks {{*siv_alt_pma|pma_ch*.pma_direct|transmit_pcs*|clkout}
```

```
{*siv_alt_pma|pma_ch*.pma_direct|transmit_pcs*|clkout}  {*pll_siv_xgmii_clk|
altpll_component|auto_generated|pll1|clk[0]} phy_mgmt_clk xgmii_tx_clk}]
set_false_path -from {*siv_10gbaser_xcvr*clk_reset_ctrl|tx_pma_rstn} -to
[get_clocks {{*siv_alt_pma|pma_ch*.pma_direct|receive_pcs*|clkout}
{*siv_alt_pma|pma_ch*.pma_direct|transmit_pcs*|clkout}  {*pll_siv_xgmii_clk|
altpll_component|auto_generated|pll1|clk[0]} phy_mgmt_clk xgmii_tx_clk}]
set_false_path -from {*siv_10gbaser_xcvr*clk_reset_ctrl|tx_usr_rstn} -to
[get_clocks {{*siv_alt_pma|pma_ch*.pma_direct|receive_pcs*|clkout}
{*siv_alt_pma|pma_ch*.pma_direct|transmit_pcs*|clkout}  {*pll_siv_xgmii_clk|
altpll_component|auto_generated|pll1|clk[0]} phy_mgmt_clk xgmii_tx_clk}]
set_false_path -from {*siv_10gbaser_xcvr*rx_analog_rst_lego|rinit} -to
[get_clocks {{*siv_alt_pma|pma_ch*.pma_direct|receive_pcs*|clkout}
{*siv_alt_pma|pma_ch*.pma_direct|transmit_pcs*|clkout}  {*pll_siv_xgmii_clk|
altpll_component|auto_generated|pll1|clk[0]} phy_mgmt_clk xgmii_tx_clk}]
set_false_path -from {*siv_10gbaser_xcvr*rx_digital_rst_lego|rinit} -to
[get_clocks {{*siv_alt_pma|pma_ch*.pma_direct|receive_pcs*|clkout}
{*siv_alt_pma|pma_ch*.pma_direct|transmit_pcs*|clkout}  {*pll_siv_xgmii_clk|
altpll_component|auto_generated|pll1|clk[0]} phy_mgmt_clk xgmii_tx_clk}]
#**************************************************************
# Set Multicycle Paths
#**************************************************************
**************************************************************
# Set Maximum Delay
#**************************************************************
#**************************************************************
# Set Minimum Delay
#**************************************************************
#**************************************************************
# Set Input Transition
#**************************************************************
```

Note: This **.sdc** file is only applicable to the 10GBASE-R PHY IP Core when compiled in isolation. You can use it as a reference to help in creating your own **.sdc** file.

Note: For Arria V and Stratix V devices, timing constraints are built into the HDL code.

Note: The SDC timing constraints and approaches to identify false paths listed for Stratix V Native PHY IP apply to all other transceiver PHYs listed in this user guide. Refer to *SDC Timing Constraints of Stratix V Native PHY* for details.

Related Information

- **SDC Timing Constraints of Stratix V Native PHY** on page 13-72
This section describes SDC examples and approaches to identify false timing paths.
- **About LogicLock Regions**

10GBASE-R PHY Simulation Files and Example Testbench

Refer to Running a Simulation Testbench for a description of the directories and files that the Intel Quartus Prime software creates automatically when you generate your 10GBASE-R PHY IP Core.

Related Information

Running a Simulation Testbench on page 1-6

Backplane Ethernet 10GBASE-KR PHY IP Core

4

2020.06.02

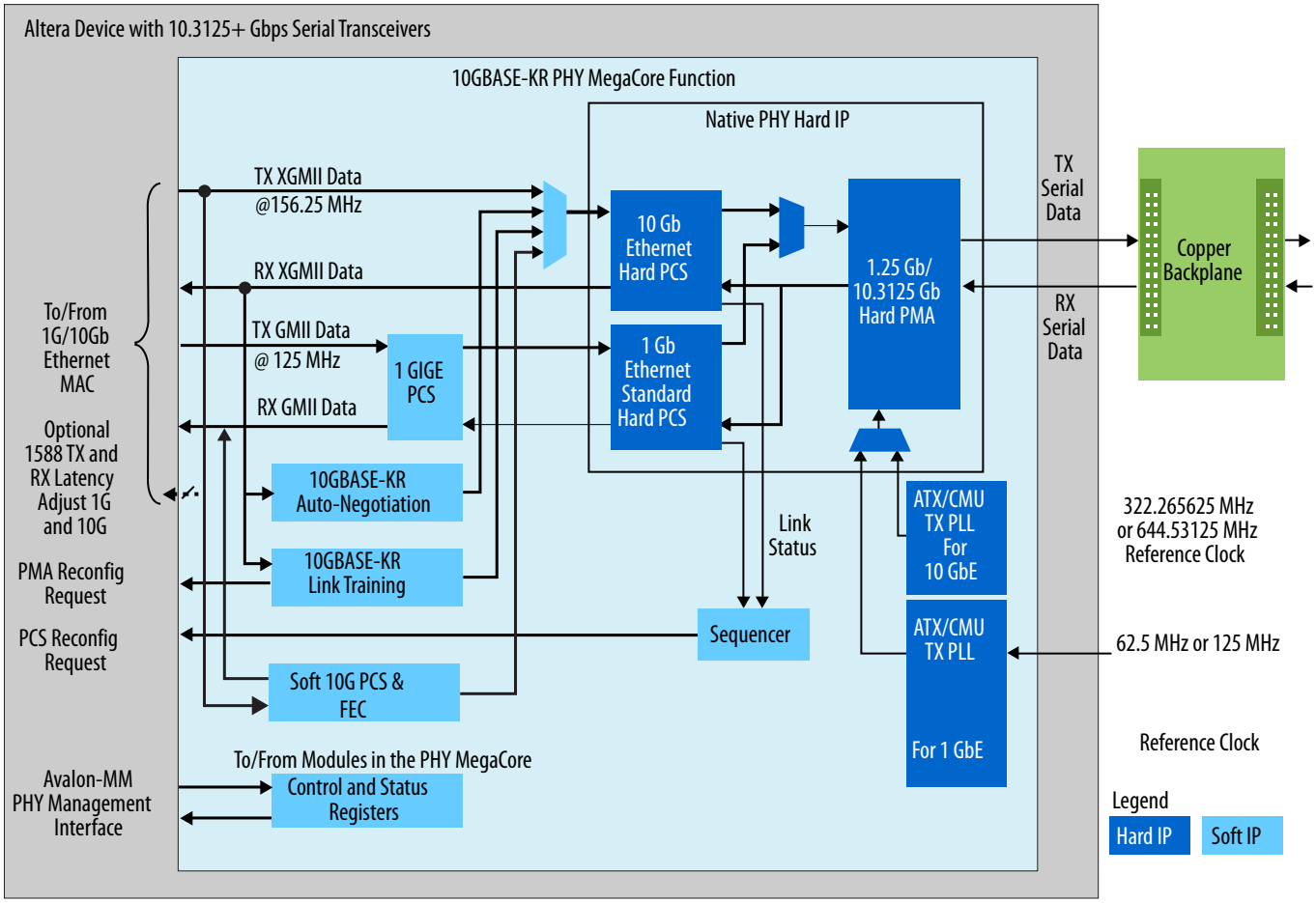
UG-01080

Subscribe Send Feedback

The Backplane Ethernet 10GBASE-KR PHY MegaCore® function is available for Stratix® V and Arria V GZ devices.

This transceiver PHY allows you to instantiate both the hard Standard PCS and the higher performance hard 10G PCS and hard PMA for a single Backplane Ethernet channel. It implements the functionality described in the *IEEE Std 802.3ap-2007 Standard*. Because each instance of the 10GBASE-KR PHY IP Core supports a single channel, you can create multi-channel designs by instantiating more than one instance of the core. The following figure shows the 10GBASE-KR transceiver PHY and additional blocks that are required to implement this core in your design.

Figure 4-1: 10GBASE-KR PHY MegaCore Function and Supporting Blocks



The Backplane Ethernet 10GBASE-KR PHY IP Core includes the following new modules to enable operation over a backplane:

- Link Training (LT)— The LT mechanism allows the 10GBASE-KR PHY to automatically configure the link-partner TX PMDs for the lowest Bit Error Rate (BER). LT is defined in Clause 72 of IEEE Std 802.3ap-2007.
- Auto negotiation (AN)—The Altera 10GBASE-KR PHY IP Core can auto-negotiate between 1000BASE-KX (1GbE) and 10GBASE-KR (10GbE) PHY types. The AN function is mandatory for Backplane Ethernet. It is defined in Clause 73 of the IEEE Std 802.3ap-2007.
- Forward Error Correction—Forward Error Correction (FEC) function is an optional feature defined in *Clause 74* of *IEEE 802.3ap-2007*. It provides an error detection and correction mechanism allowing noisy channels to achieve the Ethernet-mandated Bit Error Rate (BER) of 10^{-12} .

Related Information

IEEE Std 802.3ap-2007 Standard

10GBASE-KR PHY Release Information

Table 4-1: 10GBASE-KR PHY Release Information

Item	Description
Version	13.1
Release Date	November 2013
Ordering Codes	IP-10GBASEKR PHY (primary)
Product ID	0106
Vendor ID	6AF7

Device Family Support

IP cores provide either final or preliminary support for target Altera device families. These terms have the following definitions:

- *Final support*—Verified with final timing models for this device.
- *Preliminary support*—Verified with preliminary timing models for this device.

Table 4-2: Device Family Support

Device Family	Support	Supported Speed Grades
Arria V GZ devices–Hard PCS and PMA	Final	I3L, C3, I4, C4
Stratix V devices–Hard PCS and PMA	Final	All speed grades except I4 and C4
Other device families	No support	

Altera verifies that the current version of the Intel Quartus Prime software compiles the previous version of each IP core. Any exceptions to this verification are reported in the **MegaCore IP Library Release Notes and Errata**. Altera does not verify compilation with IP core versions older than the previous release.

Note: For speed grade information, refer to *DC and Switching Characteristics for Stratix V Devices* in the *Stratix V Device Datasheet*.

Related Information

Stratix V Device Datasheet

10GBASE-KR PHY Performance and Resource Utilization

This topic provides performance and resource utilization for the IP core in Arria V GZ and Stratix V devices.

The following table shows the typical expected resource utilization for selected configurations using the current version of the Intel Quartus Prime software targeting a Stratix V GT (5SGTMC7K2F40C2) device. The numbers of ALMs and logic registers are rounded up to the nearest 100. Resource utilization numbers reflect changes to the resource utilization reporting starting in the Quartus II software v14.1 release for 28 nm device families and upcoming device families.

Table 4-3: 10GBASE-KR PHY Performance and Resource Utilization

Module Options	ALMs	Logic Registers	Memory
10GBASE-KR PHY only, no AN or LT	400	700	0
10GBASE-KR PHY with AN and Sequencer	1000	1700	0
10GBASE-KR PHY with LT and Sequencer,	2100	2300	0
10GBASE-KR PHY with AN, LT, and Sequencer	2700	3300	0
10GBASE-KR MIF, Port A depth 256, width 16, ROM (For reconfiguration from low latency or 1GbE mode)	0	0	1 (M20K)
Low Latency MIF, Port A depth 256, width 16, ROM (Required for auto-negotiation and link training.)	0	0	1 (M20K)
10GBASE-KR PHY with FEC	3700	5100	40 (M20K)

Parameterizing the 10GBASE-KR PHY

The10GBASE-KR PHY IP Core is available for the **Arria V GZ** and **Stratix V** device families. The IP variant allows you specify either the **Backplane-KR** or **1Gb/10Gb Ethernet** variant. When you select the **Backplane-KR** variant, the **Link Training (LT)** and **Auto Negotiation (AN)** tabs appear. The **1Gb/10Gb Ethernet** variant (1G/10GbE) does not implement LT and AN parameters.

Complete the following steps to configure the 10GBASE-KR PHY IP Core:

1. Under **Tools** > **IP Catalog**, select the device family of your choice.
2. Under **Tools** > **IP Catalog** > **Interface Protocols** > **Ethernet**, select **10GBASE-KR PHY**.
3. Use the tabs on the MegaWizard Plug-In Manager to select the options required for the protocol.
4. Specify 10GBASE-KR parameters. Refer to the topics listed as Related Links to understand 10GBASE-KR parameters.
5. Click **Finish** to generate your parameterized 10GBASE-KR PHY IP Core.

Related Information

- **10GBASE-KR Link Training Parameters** on page 4-5

10GBASE-KR Link Training Parameters

The 10GBASE-KR variant provides parameters to customize the Link Training parameters.

Table 4-4: Link Training Settings

Name	Value	Description
Enable Link Training	**On/Off**	When you turn this option **On**, the core includes the link training module which configures the remote link-partner TX PMD for the lowest Bit Error Rate (BER). LT is defined in Clause 72 of *IEEE Std 802.3ap-2007*.
Enable daisy chain mode	**On/Off**	When you turn this option **On**, the core includes support for non-standard link configurations where the TX and RX interfaces connect to different link partners. This mode overrides the TX adaptation algorithm.
Enable microprocessor interface	**On/Off**	When you turn this option **On**, the core includes a microprocessor interface which enables the microprocessor mode for link training.
Maximum bit error count	**15, 31,63, 127, 255**	Specifies the maximum number of errors before the *Link Training Error* bit (0xD2, bit 4) is set indicating an unacceptable bit error rate. You can use this parameter to tune PMA settings. For example, if you see no difference in error rates between two different sets of PMA settings, you can increase the width of the bit error counter to determine if a larger counter enables you to distinguish between PMA settings.
Number of frames to send before sending actual data	**127, 255**	Specifies the number of additional training frames the local link partner delivers to ensure that the link partner can correctly detect the local receiver state.
PMA Parameters		
VMAXRULE	`0-63`	Specifies the maximum V_{OD}. The default value is 60 which represents 1200 mV.

Name	Value	Description
VMINRULE	0-63	Specifies the minimum V_{OD}. The default value is 9 which represents 165 mV.
VODMINRULE	0-63	Specifies the minimum V_{OD} for the first tap. The default value is 24 which represents 440mV.
VPOSTRULE	0-31	Specifies the maximum value that the internal algorithm for pre-emphasis will ever test in determining the optimum post-tap setting. The default value is 31.
VPRERULE	0-15	Specifies the maximum value that the internal algorithm for pre-emphasis will ever test in determining the optimum pre-tap setting. The default value is 15.
PREMAINVAL	0-63	Specifies the Preset V_{OD} Value. Set by the Preset command as defined in Clause 72.6.10.2.3.1 of the link training protocol. This is the value from which the algorithm starts. The default value is 60.
PREPOSTVAL	0-31	Specifies the preset Pre-tap Value. The default value is 0.
PREPREVAL	0-15	Specifies the preset Post-tap value. The default value is 0.
INITMAINVAL	0-63	Specifies the Initial VOD Value. Set by the Initialize command in Clause 72.6.10.2.3.2 of the link training protocol. The default value is 52.
INITPOSTVAL	0-31	Specifies the initial first Post-tap value. The default value is 30.
INITPREVAL	0-15	Specifies the Initial Pre-tap Value. The default value is 5.

10GBASE-KR Auto-Negotiation and Link Training Parameters

Table 4-5: Auto Negotiation and Link Training Settings

Name	Range	Description
Enable Auto-Negotiation	**On** **Off**	Enables or disables the Auto-Negotiation feature.

Name	Range	Description
Pause ability-C0	**On** **Off**	Depends upon MAC. Local device pause capability C2:0 = D12:10 of AN word. C0 is the same as PAUSE.
Pause ability-C1	**On** **Off**	Depends upon MAC. Local device pause capability C2:0 = D12:10 of AN word. C1 is the same as ASM_DIR.
Enable Link Training	**On** **Off**	Enables or disables the Link Training feature.
Maximum bit error count	**15, 31, 63, 127, 255, 511, 1023**	Specifies the number of bit errors for the error counter expected during each step of the link training. If the number of errors exceeds this number for each step, the core returns an error. The number of errors depends upon the amount of time for each step and the quality of the physical link media. The default value is 511.
Number of frames to send before sending actual data	**127, 255**	This timer is started when the local receiver is trained and detects that the remote receiver is ready to receive data. The local physical medium dependent (PMD) layer delivers `wait_timer` additional training frames to ensure that the link partner correctly detects the local receiver state. The default value is 127.

10GBASE-R Parameters

The 10GBASE-R parameters specify basic features of the 10GBASE-R PCS. The FEC options allow you to specify the FEC ability.

Table 4-6: 10GBASE-R Parameters

Parameter Name	Options	Description
Enable IEEE 1588 Precision Time Protocol	**On/Off**	When you turn this option **On**, the core includes logic to implement the IEEE 1588 Precision Time Protocol.
Reference clock frequency	**644.53125MHz** **322.265625MHz**	Specifies the input reference clock frequency. The default is **322.265625MHz**.

Parameter Name	Options	Description
PLL Type	**ATX** **CMU**	Specifies the PLL type. You can specify either a **CMU** or **ATX** PLL. The ATX PLL has better jitter performance at higher data rates than the CMU PLL. Another advantage of the **ATX** PLL is that it does not use a transceiver channel, while the **CMU** PLL does.
Enable additional control and status pins	**On/Off**	When you turn this option **On**, the core includes the `rx_block_lock` and `rx_hi_ber` ports.
Enable rx_recovered_clk pin	**On/Off**	When you turn this option **On**, the core includes the `rx_recovered_clk` port.
Enable pll_locked status port	**On/Off**	When you turn this option **On**, the core includes the `pll_locked` port.

Table 4-7: FEC Options

Parameter Name	Options	Description
Include FEC sublayer	**On/Off**	When you turn this option **On**, the core includes logic to implement FEC and a soft 10GBASE-R PCS.
Set `FEC_ability` bit on power up and reset	**On/Off**	When you turn this option **On**, the core sets the FEC ability bit on power up and reset.
Set `FEC_Enable` bit on power up and reset	**On/Off**	When you turn this option **On**, the core sets the FEC enable bit on power up and reset.
Set `FEC_Error_Indication_ability` bit on power up and reset	**On/Off**	When you turn this option **On**, the core indicates errors to the PCS.
Good parity counter threshold to achieve FEC block lock	Default value: 4	Specifies the number of good parity blocks the RX FEC module must receive before indicating block lock as per *Clause 74.10.2.1 of IEEE 802.3ap-2007*.
Invalid parity counter threshold to lose FEC block lock	Default value: 8	Specifies the number of bad parity blocks the RX FEC module must receive before indicating loss of block lock as per *Clause 74.10.2.1 of IEEE 802.3ap-2007*.
Use M20K for FEC Buffer (if available)	**On/Off**	When you turn this option **On**, the Intel Quartus Prime software saves resources by replacing the FEC buffer with M20K memory.

Related Information

Analog Parameters Set Using QSF Assignments on page 20-1

1GbE Parameters

The 1GbE parameters allow you to specify options for the 1GbE mode.

Table 4-8: 1Gb Ethernet Parameters

Parameter Name	Options	Description
Enable 1Gb Ethernet protocol	On/Off	When you turn this option **On**, the core includes the GMII interface and related logic.
Expose MII interface	On/Off	When you turn this option **On**, the core exposes the MII interface and related logic.
Enable IEEE 1588 Precision Time Protocol	On/Off	When you turn this option **On**, the core includes a module in the PCS to implement the IEEE 1588 Precision Time Protocol.
PHY ID (32 bit)	`32-bit value`	An optional 32-bit value that serves as a unique identifier for a particular type of PCS. The identifier includes the following components: • Bits 3-24 of the Organizationally Unique Identifier (OUI) assigned by the IEEE • 6-bit model number • 4-bit revision number If unused, do not change the default value which is 0x00000000.
PHY Core version (16 bits)	`16-bit value`	This is an optional 16-bit value identifies the PHY core version.
Reference clock frequency	125.00 MHz 62.50 MHz	Specifies the clock frequency for the 1GBASE-KR PHY IP Core. The default is 125 MHz.

Related Information

1588 Delay Requirements on page 3-29

Speed Detection Parameters

Selecting the speed detection option gives the PHY the ability to detect to link partners that support 1G/10GbE but have disabled Auto-Negotiation. During Auto-Negotiation, if AN cannot detect Differential Manchester Encoding (DME) pages from a link partner, the Sequencer reconfigures to 1GbE and 10GbE modes (Speed/Parallel detection) until it detects a valid 1G or 10GbE pattern.

Table 4-9: Speed Detection

Parameter Name	Options	Description
Enable automatic speed detection	**On** **Off**	When you turn this option **On**, the core includes the Sequencer block that sends reconfiguration requests to detect 1G or 10GbE when the Auto Negotiation block is not able to detect AN data.
Avalon-MM clock frequency	`100-162 MHz`	Specifies the clock frequency for `phy_mgmt_clk`.

Parameter Name	Options	Description
Link fail inhibit time for 10Gb Ethernet	`504 ms`	Specifies the time before `link_status` is set to FAIL or OK. A link fails if the `link_fail_inhibit_time` has expired before `link_status` is set to OK. The legal range is 500-510 ms. For more information, refer to *"Clause 73 Auto Negotiation for Backplane Ethernet"* in *IEEE Std 802.3ap-2007*.
Link fail inhibit time for 1Gb Ethernet	`40-50 ms`	Specifies the time before `link_status` is set to FAIL or OK . A link fails if the `link_fail_inhibit_time` has expired before `link_status` is set to OK. The legal range is 40-50 ms.
Enable PCS-Mode port	**On** **Off**	Enables or disables the PCS-Mode port.

PHY Analog Parameters

You can specify analog parameters using the Intel Quartus Prime Assignment Editor, the Pin Planner, or the Intel Quartus Prime Settings File (**.qsf**).

Related Information

- **Analog Settings for Arria V GZ Devices** on page 20-11
- **Analog PCB Settings for Stratix V Devices** on page 20-35

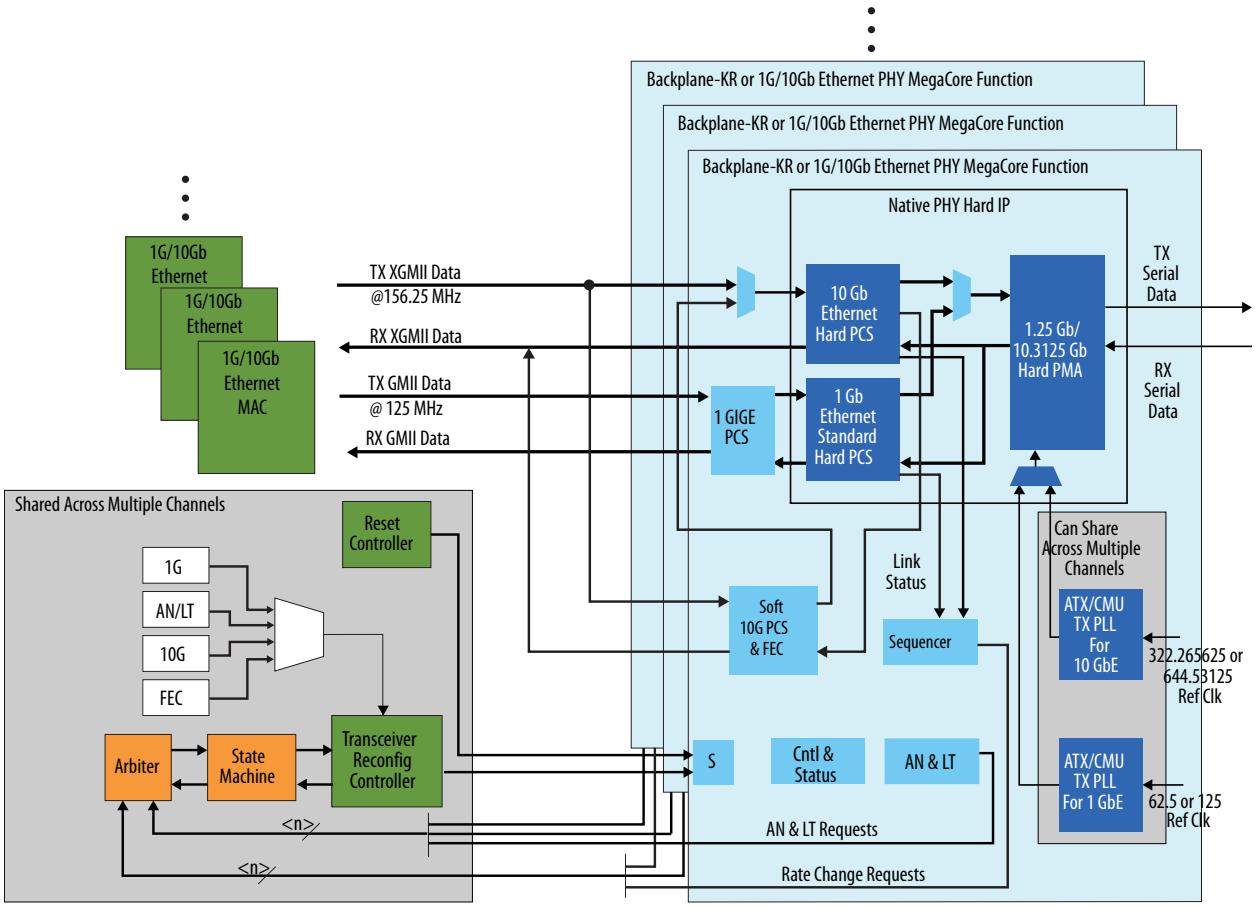
10GBASE-KR PHY IP Core Functional Description

This topic provides high-level block diagram of the 10GBASE-KR hardware.

The following figure shows the 10GBASE-KR PHY IP Core and the supporting modules required for integration into your system. In this figure, the colors have the following meanings:

- Green-Altera- Cores available Intel Quartus Prime IP Library, including the 1G/10Gb Ethernet MAC, the Reset Controller, and Transceiver Reconfiguration Controller.
- Orange-Arbitration Logic Requirements. Logic you must design, including the Arbiter and State Machine. Refer to **10GBASE-KR PHY Arbitration Logic Requirements** on page 4-16 and **10GBASE-KR PHY State Machine Logic Requirements** on page 4-16 for a description of this logic.
- White - 1G,10G and AN/LT settings files that you must generate. Refer to **Creating a 10GBASE-KR Design** on page 4-58 for more information.
- Blue-The 10GBASE-KR PHY IP core available in the Intel Quartus Prime IP Library.

Figure 4-2: Detailed 10GBASE-KR PHY IP Core Block Diagram



As this figure illustrates, the 10GBASE-KR PHY is built on the Native PHY and includes the following additional blocks implemented in soft logic to implement Ethernet functionality defined in *Clause 72* of *IEEE 802.3ap-2007*.

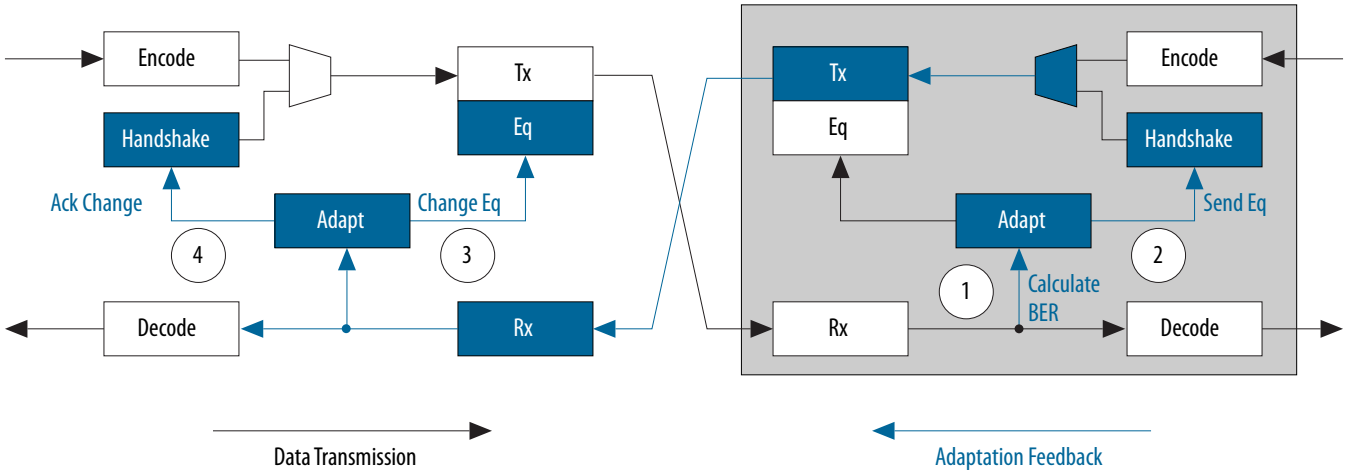
Link Training (LT), Clause 72

This module performs link training as defined in Clause 72. The module facilitates two features:

- Daisy-chain mode for non-standard link configurations where the TX and RX interfaces connect to different link partners instead of in a spoke and hub or switch topology.
- An embedded processor mode to override the state-machine-based training algorithm. This mode allows an embedded processor to establish link data rates instead of establishing the link using the state-machine-based training algorithm.

The following figure illustrates the link training process, where the link partners exchange equalization data.

Figure 4-3: TX Equalization for Link Partners



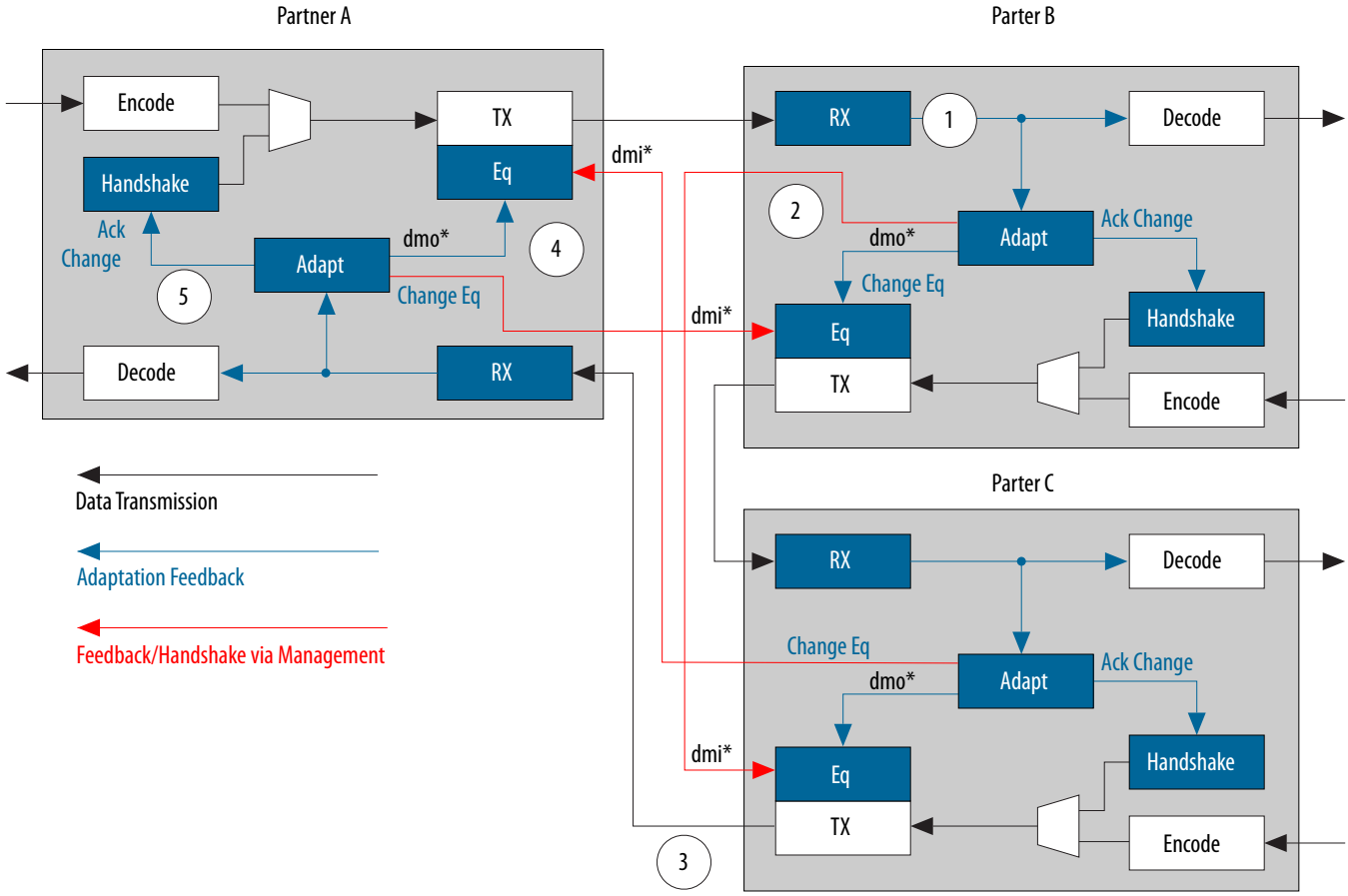
TX equalization includes the following steps which are identified in this figure.

1. The receiving link partner calculates the BER.
2. The receiving link partner transmits an update to the transmitting link partner TX equalization parameters to optimize the TX equalization settings
3. The transmitting partner updates its TX equalization settings.
4. The transmitting partner acknowledges the change.

This process is performed first for the V_{OD}, then the pre-emphasis, the first post-tap, and then pre-emphasis pre-tap.

The optional backplane daisy-chain topology can replace the spoke or hub switch topology. The following illustration highlights the steps required for TX Equalization for Daisy Chain Mode.

Figure 4-4: TX Equalization in Daisy-Chain Mode



Data transmission proceeds clockwise from link partner A, to B, to C. TX equalization includes the following steps which are identified in the figure :

1. The receiving partner B calculates the BER for data received from transmitting partner A.
2. The receiving partner B sends updates for TX link partner C.
3. The receiving link partner C transmits an update to the transmitting link partner A.
4. Transmit partner A updates its equalization settings.
5. Transmit partner A acknowledges the change.

This procedure is repeated for the other two link partners.

Sequencer

The Sequencer (Rate change) block controls the start-up (reset, power-on) sequence of the PHY IP. It automatically selects which PCS (1G, 10GbE, or Low Latency) is required and sends requests to reconfigure the PCS. The Sequencer also performs the parallel detection function that reconfigures between the 1G and 10GbE PCS until the link is established or times out.

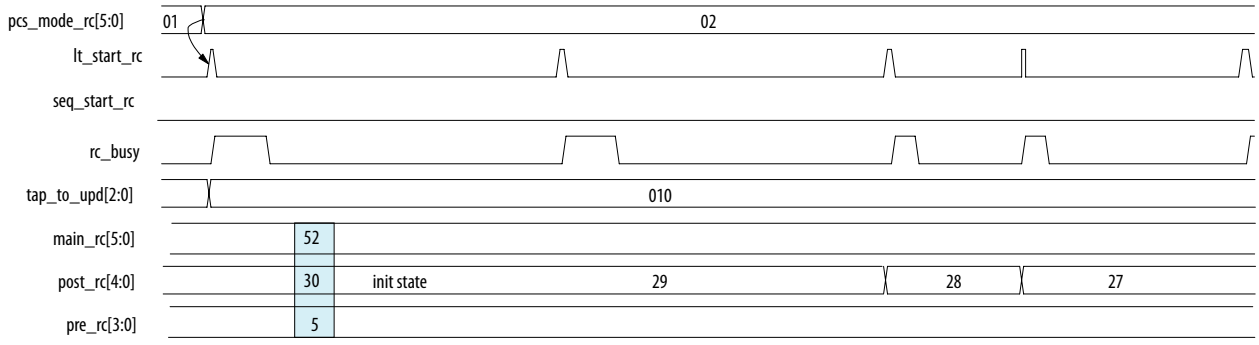
Auto Negotiation (AN), Clause 73

The Auto Negotiation module in the 10GBASE-KR PHY IP implements Clause 73 of the Ethernet standard. This module currently supports auto negotiation between 1GbE and 10GBASE-R data rates.

Auto negotiation with XAUI is not supported. Auto negotiation is run upon power up or if the auto negotiation module is reset.

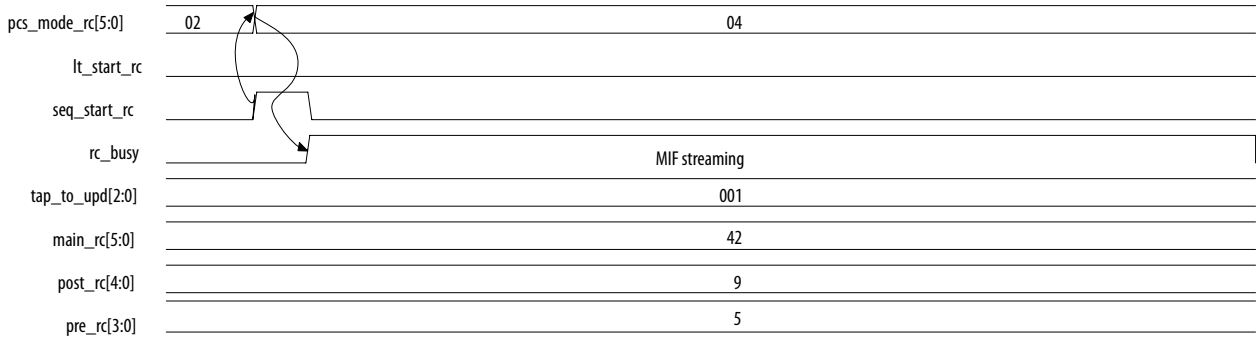
The following figures illustrate the handshaking between the Auto Negotiation, Link Training, Sequencer and Transceiver Reconfiguration Controller blocks. Reconfig controller should use `lt_start_rc` signal in combination with `main_rc`, `post_rc`, `pre_rc`, and `tap_to_upd` to change TX equalization settings.

Figure 4-5: Transition from Auto Negotiation to Link Training Mode



The Transceiver Reconfiguration Controller uses `seq_start_rc` in combination with the `pcs_mode_rc` value to initiate a change to Auto Negotiation mode or from Link Training mode to 10GBASE-KR Data mode. After TX equalization completes, this timing diagram shows the transition from Link Training mode to 10GBASE-KR Data mode and MIF streaming.

Figure 4-6: Transition from Link Training to Data Mode



Related Information

Changing Transceiver Settings Using Streamer-Based Reconfiguration on page 17-44

10GBASE-KR Dynamic Reconfiguration from 1G to 10GbE

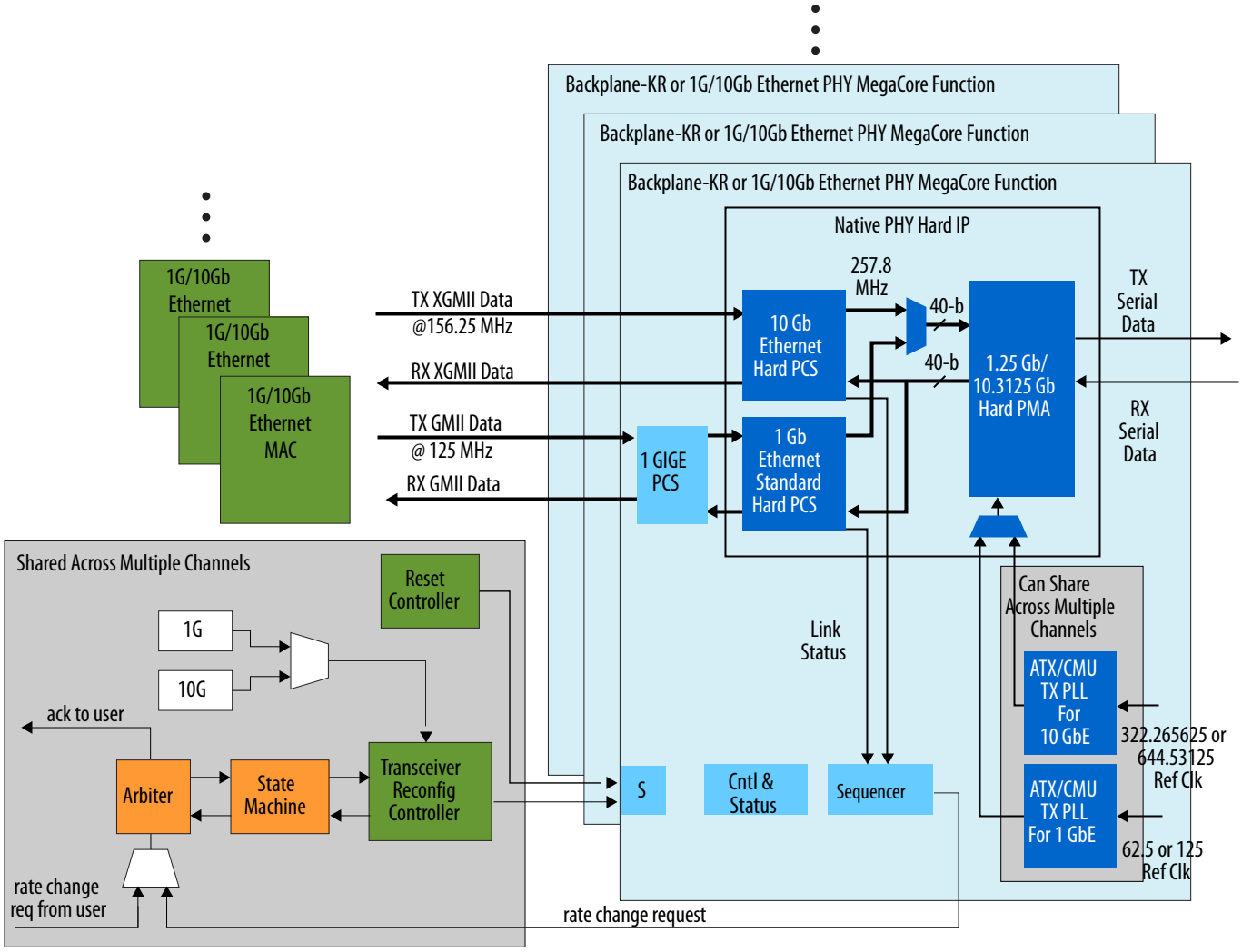
This topic illustrates the necessary logic to reconfigure between the 1G and 10G data rates.

The following figure illustrates the necessary modules to create a design that can dynamically change between 1G and 10GbE operation on a channel-by-channel basis.

In this figure, the colors have the following meanings:

- Green-Altera- Cores available Intel Quartus Prime IP Library, including the 1G/10Gb Ethernet MAC, the Reset Controller, and Transceiver Reconfiguration Controller.
- Arbitration Logic Requirements Orange-Logic you must design, including the Arbiter and State Machine. Refer to **10GBASE-KR PHY Arbitration Logic Requirements** on page 4-16 and **10GBASE-KR PHY State Machine Logic Requirements** on page 4-16 for a description of this logic.
- White-1G and 10G settings files that you must generate. Refer to **Creating a 10GBASE-KR Design** on page 4-58 for more information.
- Blue-The 10GBASE-KR PHY IP core available in the Intel Quartus Prime IP Library.

Figure 4-7: Block Diagram for Reconfiguration Example



Related Information

Creating a 10GBASE-KR Design on page 4-58

10GBASE-KR PHY Arbitration Logic Requirements

This topic describes the arbitration functionality that you must implement.

The arbiter should implement the following logic. You can modify this logic based on your system requirements:

1. Accept requests from either the Sequencer or Link Training block. Prioritize requests to meet system requirements. Requests should consist of the following two buses:
   - Channel number—specifies the requested channel
   - Mode—specifies 1G or 10G data modes or AN or LT modes for the corresponding channel
2. Select a channel for reconfiguration and send an ack/busy signal to the requestor. The requestor should deassert its request signal when the ack/busy is received.
3. Pass the selected channel and rate information or PMA reconfiguration information for LT to the state machine for processing.
4. Wait for a done signal from the state machine indicating that the reconfiguration process is complete and it is ready to service another request.

Related Information

10GBASE-KR Dynamic Reconfiguration from 1G to 10GbE on page 4-14

10GBASE-KR PHY State Machine Logic Requirements

The state machine should implement the following logic. You can modify this logic based on your system requirements:

1. Wait for `reconfig_busy` from the Transceiver Reconfiguration Controller to be deasserted and the `tx_ready` and `rx_ready` signals from the Transceiver PHY Reset Controller to be asserted. These conditions indicate that the system is ready to service a reconfiguration request.
2. Set the appropriate channel for reconfiguration.
3. Initiate the MIF streaming process. The state machine should also select the appropriate MIF (stored in the ROMs) to stream based on the requested mode.
4. Wait for the `reconfig_busy` signal from the Transceiver Reconfiguration Controller to assert and then deassert indicating the reconfiguration process is complete.
5. Toggle the digital resets for the reconfigured channel and wait for the link to be ready.
6. Deassert the `ack/busy` signal for the selected channel. Deassertion of `ack/busy` indicates to the arbiter that the reconfiguration process is complete and the system is ready to service another request.

Related Information

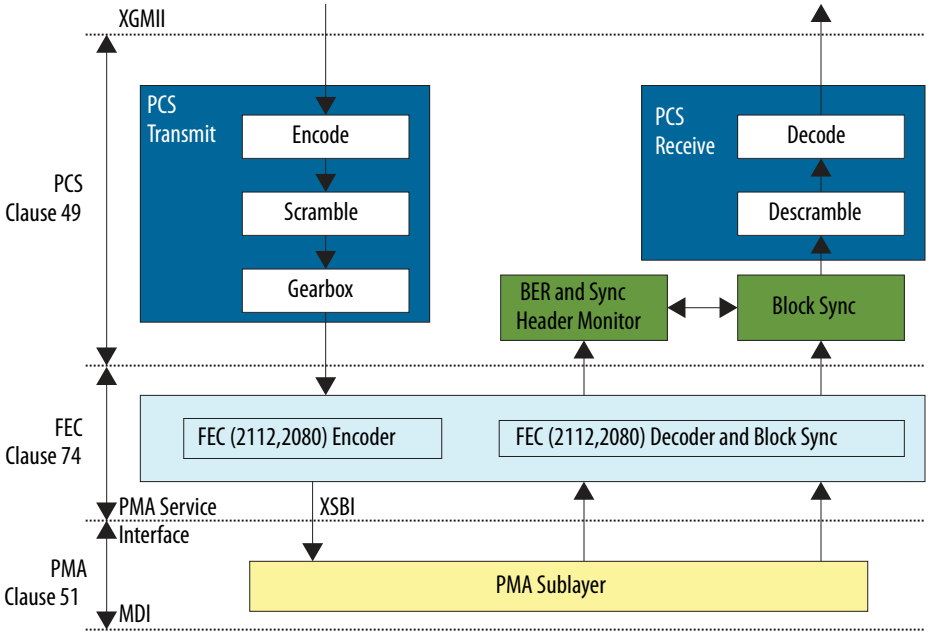
- **Transceiver PHY Reset Controller IP Core** on page 18-1
- **Transceiver Reconfiguration Controller IP Core Overview** on page 17-1

Forward Error Correction (Clause 74)

The optional Forward Error Correction (FEC) function is defined in *Clause 74* of *IEEE 802.3ap-2007*. It provides an error detection and correction mechanism allowing noisy channels to achieve the Ethernet-mandated Bit Error Rate (BER) of 10^{-12} .

The following figure illustrates the interface between the FEC, PCS and PMA modules as defined in *IEEE802.3ap-2007*.

Figure 4-8: FEC Functional Block Diagram



The FEC capability is encoded in the `FEC Ability` and `FEC Requested` bits of the base `Link Codeword`. It is transmitted within a Differential Manchester Encoded page during Auto Negotiation. The link enables the FEC function if the link partners meet the following conditions:

- Both partners advertise the FEC Ability
- At least one partner requests FEC

Note: If neither device requests FEC, FEC is not enabled even if both devices have the FEC Ability.

The TX FEC encoder (2112, 2080) creates 2112-bit FEC blocks or codewords from 32, 64B/66B encoded and scrambled 10GBASE-R words. It compresses the 32, 66-bit words into 32, 65-bit words and generates 32-bit parity using the following polynomial:

$$g(x) = x^{32} + x^{23} + x^{21} + x^{11} + x^{2} + 1$$

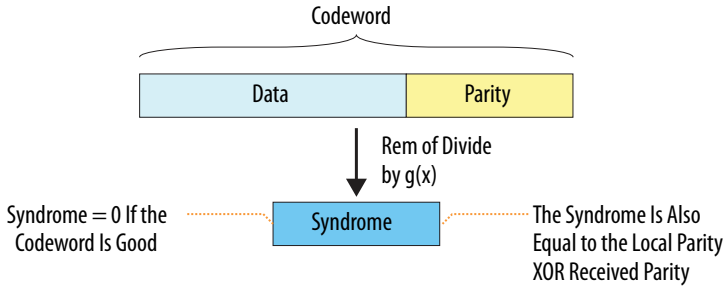
Parity is appended to the encoded data. The receiving device can use parity to detect and correct burst errors of up to 11 bits. The FEC encoder preserves the standard 10GBASE-KR line rate of 10.3125 Gbps by compressing the 32 sync bits from 64B/66B words. The TX FEC module is clocked at 161.1 MHz.

Figure 4-9: FEC Codeword Format

T_0	64 Bit Payload Word 0	T_1	64 Bit Payload Word 1	T_2	64 Bit Payload Word 2	T_3	64 Bit Payload Word 3
T_4	64 Bit Payload Word 4	T_5	64 Bit Payload Word 5	T_6	64 Bit Payload Word 6	T_7	64 Bit Payload Word 7
T_8	64 Bit Payload Word 8	T_9	64 Bit Payload Word 9	T_{10}	64 Bit Payload Word 10	T_{11}	64 Bit Payload Word 11
T_{12}	64 Bit Payload Word 12	T_{13}	64 Bit Payload Word 13	T_{14}	64 Bit Payload Word 14	T_{15}	64 Bit Payload Word 15
T_{16}	64 Bit Payload Word 16	T_{17}	64 Bit Payload Word 17	T_{18}	64 Bit Payload Word 18	T_{19}	64 Bit Payload Word 19
T_{20}	64 Bit Payload Word 20	T_{21}	64 Bit Payload Word 21	T_{22}	64 Bit Payload Word 22	T_{23}	64 Bit Payload Word 23
T_{24}	64 Bit Payload Word 24	T_{25}	64 Bit Payload Word 25	T_{26}	64 Bit Payload Word 26	T_{27}	64 Bit Payload Word 27
T_{28}	64 Bit Payload Word 28	T_{29}	64 Bit Payload Word 29	T_{30}	64 Bit Payload Word 30	T_{31}	64 Bit Payload Word 31
32 Parity Bits		Total Block Length = (32 x 65) + 32 = 2,112 Bits					

Error detection and correction consists of calculating the *syndrome* of the received codeword. The *syndrome* is the remainder from the polynomial division of the received codeword by g(x). If the syndrome is zero, the codeword is correct. If the syndrome is non-zero, you can use it to determine the most likely error.

Figure 4-10: Codewords, Parity and Syndromes



TX FEC Module Scrambler

In addition to the TX FEC encoder, the TX FEC module includes the following functions:

- **FEC Scrambler:** The FEC scrambler scrambles the encoded output. The polynomial used to scramble the encoded output ensures DC balance to facilitate block synchronization at the receiver. It is shown below.

  $$X = x^{58} + X^{39} + 1$$

- **FEC Gearbox:** The FEC gearbox adapts the FEC data width to the smaller bus width of the interface to the PCS. It supports a special 65:64 gearbox ratio.

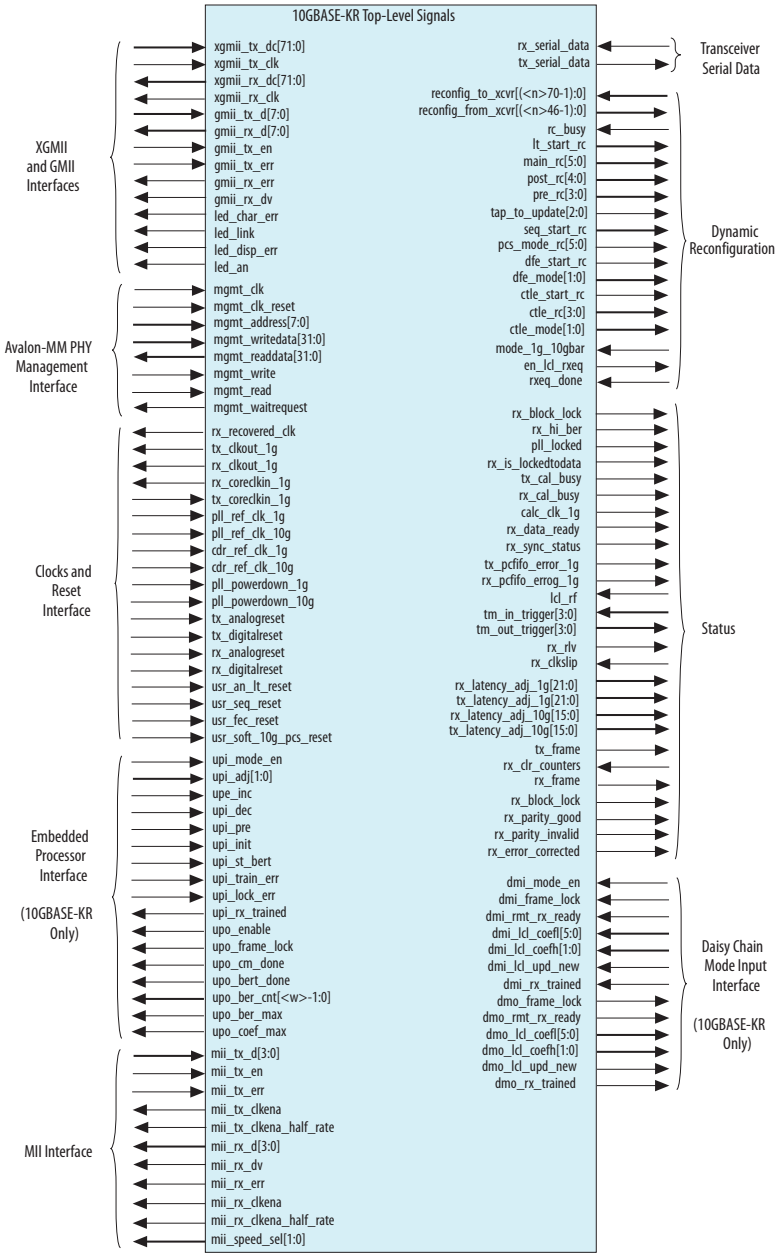
RX FEC Module

The RX FEC module is clocked at 161.1 MHz. It includes the following functions:

- **FEC Block Synchronizer:** The FEC block synchronizer achieves FEC block delineation by locking to correctly received FEC blocks. An algorithm with hysteresis maintains block and word delineation.
- **FEC Descrambler:** The FEC descrambler descrambles the received data to regenerate unscrambled data utilizing the original FEC scrambler polynomial.
- **FEC Decoder:**The FEC decoder performs the (2112, 2080) decoding by analyzing the received FEC block for errors. It can correct burst errors of 11 bits per FEC block. The FEC receive gearbox adapts the data width to the larger bus width of the PCS channel. It supports a 64:65 ratio.
- **FEC Transcode Decoder:** The FEC transcode decoder performs 65-bit to 64B/66B reconstruction by regenerating the 64B/66B sync header.

10BASE-KR PHY Interfaces

Figure 4-11: 10GBASE-KR Top-Level Signals



The block diagram shown in the GUI labels the external pins with the interface type and places the interface name inside the box. The interface type and name are used in the _hw.tcl file. If you turn on **Show signals**, the **block diagram** displays all top-level signal names. For more information about _hw.tcl files, refer to refer to the *Component Interface Tcl Reference* chapter in volume 1 of the *Intel Quartus Prime Handbook*

Related Information

Component Interface Tcl Reference

10GBASE-KR PHY Clock and Reset Interfaces

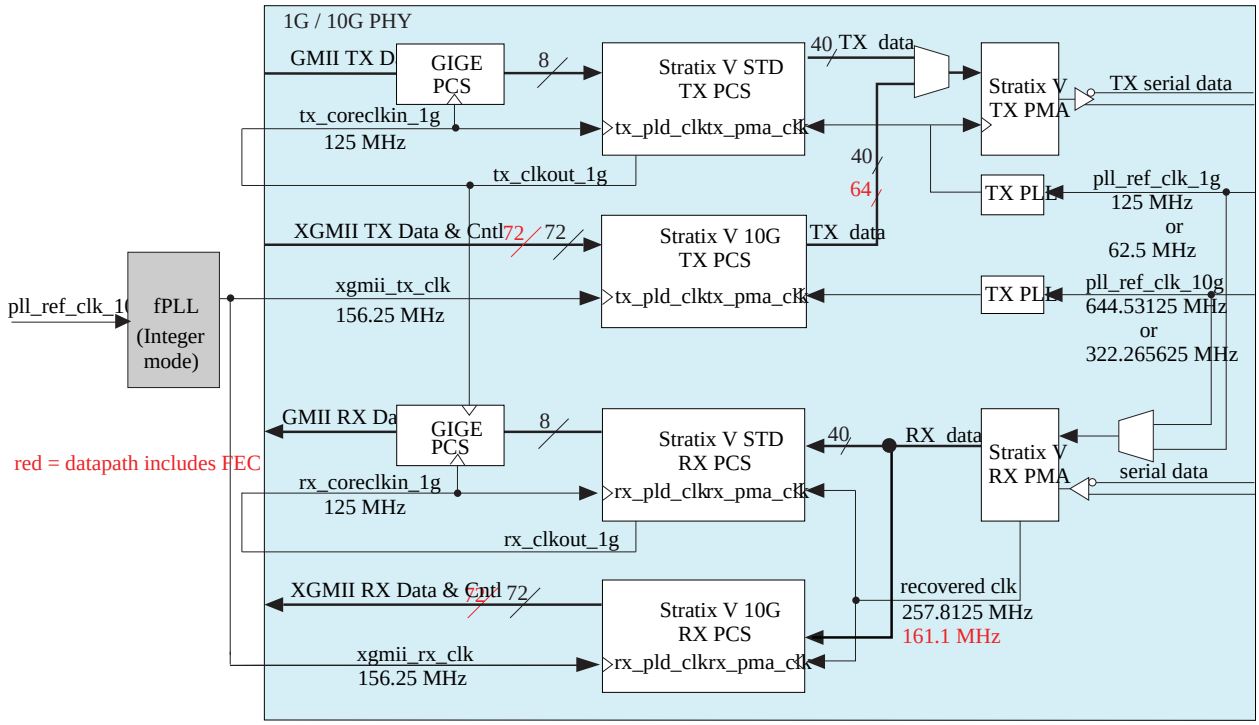
This topic provides a block diagram of the 10GBASE-KR clock and reset connectivity and describes the clock and reset signals.

Use the Transceiver PHY Reset Controller IP Core to automatically control the transceiver reset sequence. This reset controller also has manual overrides for the TX and RX analog and digital circuits to allow you to reset individual channels upon reconfiguration.

If you instantiate multiple channels within a transceiver bank they share TX PLLs. If a reset is applied to this PLL, it will affect all channels. Altera recommends leaving the TX PLL free-running after the start-up reset sequence is completed. After a channel is reconfigured you can simply reset the digital portions of that specific channel instead of going through the entire reset sequence. If you are not using the sequencer and the data link is lost, you must assert the `rx_digitalreset` when the link recovers. For more information about reset, refer to the "Transceiver PHY Reset Controller IP Core" chapter in the *Altera Transceiver PHY IP Core User Guide*.

The following figure provides an overview of the clocking for this core.

Figure 4-12: Clocks for Standard and 10G PCS and TX PLLs



To ensure proper functioning of the PCS, the maximum PPM difference between the `pll_ref_clk_10g` and the `xgmii_tx_clk` clock inputs is 0 PPM.

The following table describes the clock and reset signals. The frequencies of the XGMII clocks increases to 257.8125 MHz when you enable 1588.

Table 4-10: Clock and Reset Signals

Signal Name	Direction	Description
rx_recovered_clk	Output	The RX clock which is recovered from the received data. You can use this clock as a reference to lock an external clock source. Its frequency is 125 or 257.8125 MHz.
tx_clkout_1g	Output	GMII TX clock for the 1G TX parallel data source interface. The frequency is 125 MHz.
rx_clkout_1g	Output	GMII RX clock for the 1G RX parallel data source interface. The frequency is 125 MHz.
rx_coreclkin_1g	Input	Clock to drive the read side of the RX phase compensation FIFO in the Standard PCS. The frequency is 125 MHz.
tx_coreclkin_1g	Input	Clock to drive the write side of the TX phase compensation FIFO in the Standard PCS. The frequency is 125 MHz.
pll_ref_clk_1g	Input	Reference clock for the PMA block for the 1G mode. Its frequency is 125 or 62.5 MHz.
pll_ref_clk_10g	Input	Reference clock for the PMA block in 10G mode. Its frequency is 644.53125 or 322.265625 MHz.
pll_powerdown_1g	Input	Resets the 1Gb TX PLLs.
pll_powerdown_10g	Input	Resets the 10Gb TX PLLs.
tx_analogreset	Input	Resets the analog TX portion of the transceiver PHY.
tx_digitalreset	Input	Resets the digital TX portion of the transceiver PHY.
rx_analogreset	Input	Resets the analog RX portion of the transceiver PHY.
rx_digitalreset	Input	Resets the digital RX portion of the transceiver PHY.
usr_an_lt_reset	Input	Resets only the AN and LT logic. This signal is only available for the 10GBASE-KR variants.
usr_seq_reset	Input	Resets the sequencer. Initiates a PCS reconfiguration, and may restart AN, LT or both if these modes are enabled.
usr_fec_reset	Input	When asserted, resets the 10GBASE-KR FEC module.
usr_soft_10g_pcs_reset	Input	When asserted, resets the 10G PCS associated with the FEC module.

Related Information

- **Transceiver PHY Reset Controller IP Core** on page 18-1
- **Transceiver Reconfiguration Controller IP Core Overview** on page 17-1

10GBASE-KR PHY Data Interfaces

The following table describes the signals in the XGMII and GMII interfaces. The MAC drives the TX XGMII and GMII signals to the 10GBASE-KR PHY. The 10GBASE-KR PHY drives the RX XGMII or GMII signals to the MAC.

Table 4-11: XGMII and GMII Signals

Signal Name	Direction	Description
10GBASE-KR XGMII Data Interface		
xgmii_tx_dc[71:0]	Input	XGMII data and control for 8 lanes. Each lane consists of 8 bits of data and 1 bit of control.
xgmii_tx_clk	Input	Clock for single data rate (SDR) XGMII TX interface to the MAC. It should connect to xgmii_rx_clk . The frequency is 156.25 MHz irrespective of 1588 being enabled or disabled. Driven from the MAC. This clock is derived from the transceiver reference clock (pll_ref_clk_10g).
xgmii_rx_dc[71:0]	Output	RX XGMII data and control for 8 lanes. Each lane consists of 8 bits of data and 1 bit of control.
xgmii_rx_clk	Input	Clock for SDR XGMII RX interface to the MAC. The frequency is 156.25 MHz irrespective of 1588 being enabled or disabled. Driven from the MAC. This clock is derived from the transceiver reference clock (pll_ref_clk_10g).
10GBASE-KR GMII Data Interface		
gmii_tx_d[7:0]	Input	TX data for 1G mode. Synchronized to tx_clkout_1g clock. The TX PCS 8B/10B module encodes this data which is sent to link partner.
gmii_rx_d[7:0]	Output	RX data for 1G mode. Synchronized to rx_clkout_1g clock. The RX PCS 8B/10B decoders decodes this data and sends it to the MAC.
gmii_tx_en	Input	When asserted, indicates the start of a new frame. It should remain asserted until the last byte of data on the frame is present on gmii_tx_d .
gmii_tx_err	Input	When asserted, indicates an error. May be asserted at any time during a frame transfer to indicate an error in that frame.
gmii_rx_err	Output	When asserted, indicates an error. May be asserted at any time during a frame transfer to indicate an error in that frame.
gmii_rx_dv	Output	When asserted, indicates the start of a new frame. It remains asserted until the last byte of data on the frame is present on gmii_rx_d .

10GBASE-KR GMII Data Interface		
led_char_err	Output	10-bit character error. Asserted for one rx_clkout_1g cycle when an erroneous 10-bit character is detected
led_link	Output	When asserted, indicates successful link synchronization.
led_disp_err	Output	Disparity error signal indicating a 10-bit running disparity error. Asserted for one rx_clkout_1g cycle when a disparity error is detected. A running disparity error indicates that more than the previous and perhaps the current received group had an error.
led_an	Output	Clause 37 Auto-Negotiation status. The PCS function asserts this signal when Auto-Negotiation completes.

10GBASE-KR PHY XGMII Mapping to Standard SDR XGMII Data

The 72-bit TX XGMII data bus format is different than the standard SDR XGMII interface. The following table lists the mapping of this non-standard format to the standard SDR XGMII interface.

Table 4-12: TX XGMII Mapping to Standard SDR XGMII Interface

Signal Name	SDR XGMII Signal Name	Description
xgmii_tx_dc[7:0]	xgmii_sdr_data[7:0]	Lane 0 data
xgmii_tx_dc[8]	xgmii_sdr_ctrl[0]	Lane 0 control
xgmii_tx_dc[16:9]	xgmii_sdr_data[15:8]	Lane 1 data
xgmii_tx_dc[17]	xgmii_sdr_ctrl[1]	Lane 1 control
xgmii_tx_dc[25:18]	xgmii_sdr_data[23:16]	Lane 2 data
xgmii_tx_dc[26]	xgmii_sdr_ctrl[2]	Lane 2 control
xgmii_tx_dc[34:27]	xgmii_sdr_data[31:24]	Lane 3 data
xgmii_tx_dc[35]	xgmii_sdr_ctrl[3]	Lane 3 control
xgmii_tx_dc[43:36]	xgmii_sdr_data[39:32]	Lane 4 data
xgmii_tx_dc[44]	xgmii_sdr_ctrl[4]	Lane 4 control
xgmii_tx_dc[52:45]	xgmii_sdr_data[47:40]	Lane 5 data
xgmii_tx_dc[53]	xgmii_sdr_ctrl[5]	Lane 5 control
xgmii_tx_dc[61:54]	xgmii_sdr_data[55:48]	Lane 6 data
xgmii_tx_dc[62]	xgmii_sdr_ctrl[6]	Lane 6 control
xgmii_tx_dc[70:63]	xgmii_sdr_data[63:56]	Lane 7 data

Signal Name	SDR XGMII Signal Name	Description
xgmii_tx_dc[71]	xgmii_sdr_ctrl[7]	Lane 7 control

The 72-bit RX XGMII data bus format is different from the standard SDR XGMII interface. The following table lists the mapping of this non-standard format to the standard SDR XGMII interface:

Table 4-13: RX XGMII Mapping to Standard SDR XGMII Interface

Signal Name	XGMII Signal Name	Description
xgmii_rx_dc[7:0]	xgmii_sdr_data[7:0]	Lane 0 data
xgmii_rx_dc[8]	xgmii_sdr_ctrl[0]	Lane 0 control
xgmii_rx_dc[16:9]	xgmii_sdr_data[15:8]	Lane 1 data
xgmii_rx_dc[17]	xgmii_sdr_ctrl[1]	Lane 1 control
xgmii_rx_dc[25:18]	xgmii_sdr_data[23:16]	Lane 2 data
xgmii_rx_dc[26]	xgmii_sdr_ctrl[2]	Lane 2 control
xgmii_rx_dc[34:27]	xgmii_sdr_data[31:24]	Lane 3 data
xgmii_rx_dc[35]	xgmii_sdr_ctrl[3]	Lane 3 control
xgmii_rx_dc[43:36]	xgmii_sdr_data[39:32]	Lane 4 data
xgmii_rx_dc[44]	xgmii_sdr_ctrl[4]	Lane 4 control
xgmii_rx_dc[52:45]	xgmii_sdr_data[47:40]	Lane 5 data
xgmii_rx_dc[53]	xgmii_sdr_ctrl[5]	Lane 5 control
xgmii_rx_dc[61:54]	xgmii_sdr_data[55:48]	Lane 6 data
xgmii_rx_dc[62]	xgmii_sdr_ctrl[6]	Lane 6 control
xgmii_rx_dc[70:63]	xgmii_sdr_data[63:56]	Lane 7 data
xgmii_rx_dc[71]	xgmii_sdr_ctrl[7]	Lane 7 control

10GBASE-KR PHY Serial Data Interface

This topic describes the serial data interface.

Signal Name	Direction	Description
rx_serial_data	Input	RX serial input data
tx_serial_data	Output	TX serial output data

MII Interface Signals

The following signals are available when you turn on the **Expose MII interface** parameter.

Table 4-14: MII Interface Signals

Signal Name	Direction	Description
mii_tx_d[3:0]	Input	TX data to be encoded and sent to link partner.
mii_tx_en	Input	MII transmit control signal.
mii_tx_err	Input	MII transmit error signal.
mii_tx_clkena	Output	Clock enabled signal from PHY to MAC. Following are the effective rates: • For 100 Mbps: 25 MHz • For 10 Mbps: 2.5 MHz
mii_tx_clkena_half_rate	Output	Clock enabled signal from PHY to MAC. Following are the effective rates: • For 100 Mbps: 12.5 MHz • For 10 Mbps: 1.25 MHz
mii_rx_d[3:0]	Output	RX data to be encoded and sent to link partner.
mii_rx_dv	Output	MII receive control signal.
mii_rx_err	Output	MII receive error signal.
mii_rx_clkena	Output	Clock enabled signal from PHY to MAC. Following are the effective rates: • For 100 Mbps: 25 MHz • For 10 Mbps: 2.5 MHz
mii_rx_clkena_half_rate	Output	Clock enabled signal from PHY to MAC. Following are the effective rates: • For 100 Mbps: 12.5 MHz • For 10 Mbps: 1.25 MHz
mii_speed_sel[1:0]	Output	This signal indicates the current speed of the PHY. • 2'b00: 10 Gbps • 2'b01: 1 Gbps • 2'b10: 100 Mbps • 2'b11: 10 Mbps

10GBASE-KR PHY Control and Status Interfaces

The 10GBASE-KR XGMII and GMII interface signals drive data to and from PHY.

Table 4-15: Control and Status Signals

Signal Name	Direction	Description
rx_block_lock	Output	Asserted to indicate that the block synchronizer has established synchronization.

Signal Name	Direction	Description
rx_hi_ber	Output	Asserted by the BER monitor block to indicate a Sync Header high bit error rate greater than 10^{-4}.
pll_locked	Output	When asserted, indicates the TX PLL is locked.
rx_is_lockedtodata	Output	When asserted, indicates the RX channel is locked to input data.
tx_cal_busy	Output	When asserted, indicates that the initial TX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP. You must hold the channel in reset until calibration completes.
rx_cal_busy	Output	When asserted, indicates that the initial RX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP.
calc_clk_1g	Input	An independent clock to calculate the latency of the SGMII TX and RX FIFOs. It is only required for when you enable 1588 in 1G mode. The calc_clk_1g should have a frequency that is not equivalent to 8 ns (125MHz). The accuracy of the PCS latency measurement is limited by the greatest common denominator (GCD) of the RX and TX clock periods (8 ns) and calc_clk_1g. The GCD is 1 ns, if no other higher common factor exists. When the GCD is 1, the accuracy of the measurement is 1 ns. If the period relationship has too small a phase, the phase measurement requires more time than is available. Theoretically, 8.001 ns would provide 1 ps of accuracy. But this phase measurement period requires 1000 cycles to converge which is beyond the averaging capability of the design. The GCD of the clock periods should be no less than 1/64 ns (15ps). To achieve high accuracy for all speed modes, the recommended frequency for calc_clk_1g is 80 MHz. In addition, the 80 MHz clock should have same parts per million (ppm) as the 125 MHz pll_ref_clk_1g input. The random error without a rate match FIFO mode is: • +/- 1 ns at 1000 Mbps • +/- 5 ns at 100 Mbps • +/- 25 ns at 10 Mbps
rx_sync_status	Output	When asserted, indicates the Standard PCS word aligner has aligned to in incoming word alignment pattern.
tx_pcfifo_error_1g	Output	When asserted, indicates that the Standard PCS TX phase compensation FIFO is either full or empty.
rx_pcfifo_error_1g	Output	When asserted, indicates that the Standard PCS RX phase compensation FIFO is either full or empty.

Signal Name	Direction	Description
lcl_rf	Input	When asserted, indicates a Remote Fault (RF).The MAC to sends this fault signal to its link partner. Remote Fault (RF) is encoded in bit D13 of the base Link Codeword. Bit 3 of the Auto Negotiation Advanced Remote Fault register (0xC2) records this error.
tm_in_trigger[3:0]	Input	This is an optional signal that can be used for hardware testing by using an oscilloscope or logic analyzer to trigger events. If unused, tie this signal to 1'b0.
tm_out_trigger[3:0]	Output	This is an optional signal that can be used for hardware testing by using an oscilloscope or logic analyzer to trigger events. You can ignore this signal if not used.
rx_rlv	Output	When asserted, indicates a run length violation.
rx_clkslip	Input	When you turn this signal on, the deserializer skips one serial bit or the serial clock is paused for one cycle to achieve word alignment. As a result, the period of the parallel clock can be extended by 1 unit interval (UI). This is an optional control input signal.
rx_latency_adj_1g[21:0]	Output	When you enable 1588, this signal outputs the real time latency in GMII clock cycles (125 MHz) for the RX PCS and PMA datapath for 1G mode. Bits 0 to 9 represent the fractional number of clock cycles. Bits 10 to 21 represent the number of clock cycles.
tx_latency_adj_1g[21:0]	Output	When you enable 1588, this signal outputs real time latency in GMII clock cycles (125 MHz) for the TX PCS and PMA datapath for 1G mode. Bits 0 to 9 represent the fractional number of clock cycles. Bits 10 to 21 represent the number of clock cycles.
rx_latency_adj_10g[15:0]	Output	When you enable 1588, this signal outputs the real time latency in XGMII clock cycles (156.25 MHz) for the RX PCS and PMA datapath for 10G mode. Bits 0 to 9 represent the fractional number of clock cycles. Bits 10 to 15 represent the number of clock cycles.
tx_latency_adj_10g[15:0]	Output	When you enable 1588, this signal outputs real time latency in XGMII clock cycles (156.25 MHz) for the TX PCS and PMA datapath for 10G mode. Bits 0 to 9 represent the fractional number of clock cycles. Bits 10 to 15 represent the number of clock cycles.
rx_data_ready	Output	When asserted, indicates that the MAC can begin sending data to the 10GBASE-KR PHY IP Core.
tx_frame	Output	Asynchronous status flag output of the TX FEC module. When asserted, indicates the beginning of the generated 2112-bit FEC frame.
rx_clr_counters	Input	When asserted, resets the status counters in the RX FEC module. This is an asynchronous input.

Signal Name	Direction	Description
rx_frame	Output	Asynchronous status flag output of the RX FEC module. When asserted, indicates the beginning of a 2112-bit received FEC frame.
rx_block_lock	Output	Asynchronous status flag output of the RX FEC module. When asserted, indicates successful FEC block lock.
rx_parity_good	Output	Asynchronous status flag output of the RX FEC module. When asserted, indicates that the parity calculation is good for the current received FEC frame. Used in conjunction with the rx_frame signal.
rx_parity_invalid	Output	Asynchronous status flag output of the RX FEC module. When asserted, indicates that the parity calculation is not good for the current received FEC frame. Used in conjunction with the rx_frame signal.
rx_error_corrected	Output	Asynchronous status flag output of the RX FEC module. When asserted, indicates that an error was found and corrected in the current received FEC frame. Used in conjunction with the rx_frame signal.

Daisy-Chain Interface Signals

The optional daisy-chain interface signals connect link partners using a daisy-chain topology.

Table 4-16: Daisy Chain Interface Signals

Signal Name	Direction	Description
dmi_mode_en	Input	When asserted, enable Daisy Chain mode.
dmi_frame_lock	Input	When asserted, the daisy chain state machine has locked to the training frames.
dmi_rmt_rx_ready	Input	Corresponds to bit 15 of Status report field. When asserted, the remote receiver.
dmi_lcl_coefl[5:0]	Input	Local update low bits[5:0]. In daisy-chained configurations, the local update coefficients substitute for the coefficients that would be set using Link Training.
dmi_lcl_coefh[1:0]	Input	Local update high bits[13:12]. In daisy-chained configurations, the local update coefficients substitute for the coefficients that would be set using Link Training.
dmi_lcl_upd_new	Input	When asserted, indicates a local update has occurred.

Signal Name	Direction	Description
dmi_rx_trained	Input	When asserted, indicates that the state machine has finished local training.
dmo_frame_lock	Output	When asserted, indicates that the state machine has locked to the training frames.
dmo_rmt_rx_ready	Output	Corresponds to the link partner's remote receiver ready bit.
dmo_lcl_coefl[5:0]	Output	Local update low bits[5:0]. In daisy-chained configurations, the local update coefficients substitute for the coefficients that would be set using Link Training.
dmo_lcl_coefh[1:0]	Output	Local update high bits[13:12]. In daisy-chained configurations, the local update coefficients substitute for the coefficients that would be set using Link Training.
dmo_lcl_upd_new	Output	When asserted, indicates a local update has occurred.
dmo_rx_trained	Output	When asserted, indicates that the state machine has finished local training.

Embedded Processor Interface Signals

The optional embedded processor interface signals allow you to use the embedded processor mode of Link Training. This mode overrides the TX adaptation algorithm and allows an embedded processor to initialize the link.

Table 4-17: Embedded Processor Interface Signals

Signal Name	Direction	Description
upi_mode_en	Input	When asserted, enables embedded processor mode.
upi_adj[1:0]	Input	Selects the active tap. The following encodings are defined: • 2'b01: Main tap • 2'b10: Post-tap • 2'b11: Pre-tap
upi_inc	Input	When asserted, sends the increment command.
upi_dec	Input	When asserted, sends the decrement command.
upi_pre	Input	When asserted, sends the preset command.
upi_init	Input	When asserted, sends the initialize command.
upi_st_bert	Input	When asserted, starts the BER timer.
upi_train_err	Input	When asserted, indicates a training error.

Signal Name	Direction	Description
upi_rx_trained	Input	When asserted, the local RX interface is trained.
upo_enable	Output	When asserted, indicates that the 10GBASE-KR PHY IP Core is ready to receive commands from the embedded processor.
upo_frame_lock	Output	When asserted, indicates the receiver has achieved training frame lock.
upo_cm_done	Output	When asserted, indicates the master state machine handshake is complete.
upo_bert_done	Output	When asserted, indicates the BER timer is at its maximum count.
upo_ber_cnt[<w>-1:0]	Output	Records the BER count.
upo_ber_max	Output	When asserted, the BER counter has rolled over.
upo_coef_max	Output	When asserted, indicates that the remote coefficients are at their maximum or minimum values.

Dynamic Reconfiguration Interface Signals

You can use the dynamic reconfiguration interface signals to dynamically change between 1G,10G data rates and AN or LT mode. These signals also used to update TX coefficients during Link Training..

Table 4-18: Dynamic Reconfiguration Interface Signals

Signal Name	Direction	Description
reconfig_to_xcvr [(<n>70-1):0]	Input	Reconfiguration signals from the Reconfiguration Design Example. <*n*> grows linearly with the number of reconfiguration interfaces.
reconfig_from_xcvr [(<n>46-1):0]	Output	Reconfiguration signals to the Reconfiguration Design Example. <*n*> grows linearly with the number of reconfiguration interfaces.
rc_busy	Input	When asserted, indicates that reconfiguration is in progress.
lt_start_rc	Output	When asserted, starts the TX PMA equalization reconfiguration.
main_rc[5:0]	Output	The main TX equalization tap value which is the same as V_{OD}. The following example mappings to the V_{OD} settings are defined: • 6'b111111: FIR_MAIN_12P6MA • 6'b111110: FIR_MAIN_12P4MA • 6'b000001: FIR_MAIN_P2MA • 6'b000000: FIR_MAIN_DISABLED

Signal Name	Direction	Description
post_rc[4:0]	Output	The post-cursor TX equalization tap value. This signal translates to the first post-tap settings. The following example mappings are defined: • 5'b11111: FIR_1PT_6P2MA • 5'b11110: FIR_1PT_6P0MA • 5'b00001: FIR_1PT_P2MA • 5'b00000: FIR_1PT_DISABLED
pre_rc[3:0]	Output	The pre-cursor TX equalization tap value. This signal translates to pre-tap settings. The following example mappings are defined: • 4'b1111: FIR_PRE_3P0MA • 4'b1110: FIR_PRE_P28MA • 4'b0001: FIR_PRE_P2MA • 4'b0000: FIR_PRE_DISABLED
tap_to_upd[2:0]	Output	Specifies the TX equalization tap to update to optimize signal quality. The following encodings are defined: • 3'b100: main tap • 3'b010: post-tap • 3'b001: pre-tap
seq_start_rc	Output	When asserted, starts PCS reconfiguration.
pcs_mode_rc[5:0]	Output	Specifies the PCS mode for reconfig using 1-hot encoding. The following modes are defined: • 6'b000001: Auto-Negotiation mode • 6'b000010: Link Training mode • 6'b000100: 10GBASE-KR data mode • 6'b001000: GigE data mode • 6'b010000: Reserved • 6'b100000:10G data mode with FEC
dfe_start_rc	Output	When asserted, starts the RX DFE equalization of the PMA.
dfe_mode[1:0]	Output	Specifies the DFE operation mode. Valid at the rising edge of the def_start_rc signal and held until the falling edge of the rc_busy signal. The following encodings are defined: • 2'b00: Disable DFE • 2'b01: DFE triggered mode • 2'b10: Reserved • def_start_rcd'b11: Reserved
ctle_start_rc	Output	When asserted, starts continuous time-linear equalization (CTLE) reconfiguration.

Signal Name	Direction	Description
ctle_mode[1:0]	Output	Specifies CTLE mode. These signals are valid at the rising edge of the ctle_start_rc signal and held until the falling edge of the rc_busy signal. The following encodings are defined: • 2'b00: ctle_rc[3:0] drives the value of CTLE set during link training • 2'b01: Reserved • 2b'10: Reserved • 2'b11: Reserved
ctle_rc[3:0]	Output	RX CTLE value. This signal is valid at the rising edge of the ctle_start_rc signal and held until the falling edge of the rc_busy signal. The valid range of values is 4'b0000-4'b1111.
mode_1g_10gbar	Input	This signal indicates the requested mode for the channel. A 1 indicates 1G mode and a 0 indicates 10G mode. This signal is only used when the sequencer which performs automatic speed detection is disabled.
en_lcl_rxeq	Output	This signal is not used. You can leave this unconnected.
rxeq_done	Input	Link training requires RX equalization to be complete. Tie this signal to 1 to indicate that RX equalization is complete.

Register Interface Signals

The Avalon-MM master interface signals provide access to all registers.

Refer to the *Typical Slave Read and Write Transfers* and *Master Transfers* sections in the *Avalon Memory-Mapped Interfaces* chapter of the *Avalon Interface Specifications* for timing diagrams.

Table 4-19: Avalon-MM Interface Signals

Signal Name	Direction	Description
mgmt_clk	Input	The clock signal that controls the Avalon-MM PHY management, interface. If you plan to use the same clock for the PHY management interface and transceiver reconfiguration, you must restrict the frequency range to 100-125 MHz to meet the specification for the transceiver reconfiguration clock.
mgmt_clk_reset	Input	Resets the PHY management interface. This signal is active high and level sensitive.
mgmt_addr[7:0]	Input	8-bit Avalon-MM address.

Signal Name	Direction	Description
mgmt_writedata[31:0]	Input	Input data.
mgmt_readdata[31:0]	Output	Output data.
mgmt_write	Input	Write signal. Active high.
mgmt_read	Input	Read signal. Active high.
mgmt_waitrequest	Output	When asserted, indicates that the Avalon-MM slave interface is unable to respond to a read or write request. When asserted, control signals to the Avalon-MM slave interface must remain constant.

Related Information

Avalon Interface Specifications

10GBASE-KR PHY Register Definitions

The Avalon-MM master interface signals provide access to the control and status registers.

The following table specifies the control and status registers that you can access over the Avalon-MM PHY management interface. A single address space provides access to all registers.

Note:
- Unless otherwise indicated, the default value of all registers is 0.
- Writing to reserved or undefined register addresses may have undefined side effects.
- To avoid any unspecified bits to be erroneously overwritten, you must perform read-modify-writes to change the register values.

Table 4-20: 10GBASE-KR Register Definitions

Word Addr	Bit	R/W	Name	Description
0xB0	0	RW	Reset SEQ	When set to 1, resets the 10GBASE-KR sequencer, initiates a PCS reconfiguration, and may restart Auto-Negotiation, Link Training or both if AN and LT are enabled (10GBASE-KR mode). SEQ Force Mode[2:0] forces these modes. This reset self clears.
	1	RW	Disable AN Timer	Auto-Negotiation disable timer. If disabled (Disable AN Timer = 1) , AN may get stuck and require software support to remove the ABILITY_DETECT capability if the link partner does not include this feature. In addition, software may have to take the link out of loopback mode if the link is stuck in the ACKNOWLEDGE_DETECT state. To enable this timer set Disable AN Timer = 0.

Word Addr	Bit	R/W	Name	Description
	2	RW	Disable LF Timer	When set to 1, disables the Link Fault timer. When set to 0, the Link Fault timer is enabled.
	6:4	RW	SEQ Force Mode[2:0]	Forces the sequencer to a specific protocol. Must write the Reset SEQ bit to 1 for the Force to take effect. The following encodings are defined: • 3'b000: No force • 3'b001: GigE • 3'b010: Reserved • 3'b011: Reserved • 3'b100: 10GBASE-R • 3'b101: 10GBASE-KR • Others: Reserved
	16	RW	Assert KR FEC Ability	When set to 1, indicates that the FEC ability is supported. This bit defaults to 1 if the **Set FEC_ability bit on power up/ reset** bit is on. For more information, refer to the FEC variable FEC_Enable as defined in *Clause 74.8.2* and 10GBASE-KR PMD control register bit (1.171.0) IEEE 802.3ap-2007.
	17	RW	Enable KR FEC Error Indication	When set to 1, the FEC module indicates errors to the 10G PCS. For more information, refer to the KR FEC variable FEC_enable_Error_to_PCS and 10GBASE-KR PMD control register bit (1.171.1) as defined in *Clause 74.8.3 of IEEE 302.3ap-2007*.
	18	RW	Assert KR FEC Request	When set to 1, indicates that the core is requesting the FEC ability. When this bit changes, you must assert the Reset SEQ bit (0xB0[0]) to renegotiate with the new value.
0xB1	0	R	SEQ Link Ready	When asserted, the sequencer is indicating that the link is ready.
	1	R	SEQ AN timeout	When asserted, the sequencer has had an Auto-Negotiation timeout. This bit is latched and is reset when the sequencer restarts Auto-Negotiation.
	2	R	SEQ LT timeout	When set, indicates that the Sequencer has had a timeout.

Word Addr	Bit	R/W	Name	Description
	13:8	R	SEQ Reconfig Mode[5:0]	Specifies the Sequencer mode for PCS reconfiguration. The following modes are defined: • Bit 8, mode[0]: AN mode • Bit 9, mode[1]: LT Mode • Bit 10, mode[2]: 10G data mode • Bit 11, mode[3]: Gige data mode • Bit 12, mode[4]: Reserved for XAUI • Bit13, mode[5]: 10G FEC mode
	16	R	KR FEC Ability	Indicates whether or not the 10GBASE-KR PHY supports FEC. For more information, refer to the FEC variable FEC_Enable as defined in *Clause 74.8.2* and 10GBASE-KR PMD control register bit (1.171.0) IEEE 802.3ap-2007.
	17	R	Enable KR FEC Error Indication Ability	When set to 1, indicates that the 10GBASE-KR PHY is capable of reporting FEC decoding errors to the PCS. For more information, refer to the KR FEC variable FEC_enable_Error_to_PCS and 10GBASE-KR PMD control register bit (1.171.1) as defined in *Clause 74.8.3 of IEEE 302.3ap-2007*.
0xB2	0	RW	FEC TX trans error	When asserted, indicates that the error insertion feature in the FEC Transcoder is enabled.
	1	RW	FEC TX burst error	When asserted, indicates that the error insertion feature in the FEC Encoder is enabled.
	5:2	RW	FEC TX burst length	Specifies the length of the error burst. Values 1-16 are available.
	10:6		Reserved	
	11	RWSC	FEC TX Error Insert	Writing a 1 inserts 1 error pulse into the TX FEC depending on the Transcoder and Burst error settings. Software clears this register.
	31:15	RWSC	Reserved	
0xB3	31:0	RSC	FEC Corrected Blocks	Counts the number of corrected FEC blocks. Resets to 0 when read. Otherwise, it holds at the maximum count and does not roll over. Refer to *Clause 74.8.4.1 of IEEE 802.3ap-2000* for details.

Word Addr	Bit	R/W	Name	Description
0xB4	31:0	RSC	FEC Uncorrected Blocks	Counts the number of uncorrectable FEC blocks. Resets to 0 when read. Otherwise, it holds at the maximum count and does not roll over. Refer to *Clause 74.8.4.1 of IEEE 802.3ap-2000* for details.
0xC0	0	RW	AN enable	When set to 1, enables Auto-Negotiation function. The default value is 1. For additional information, refer to bit 7.0.12 in Clause 73.8 Management Register Requirements, of *IEEE 802.3ap-2007*.
	1	RW	AN base pages ctrl	When set to 1, the user base pages are enabled. You can send any arbitrary data via the user base page low/high bits. When set to 0, the user base pages are disabled and the state machine generates the base pages to send.
	2	RW	AN next pages ctrl	When set to 1, the user next pages are enabled. You can send any arbitrary data via the user next page low/high bits. When set to 0, the user next pages are disabled. The state machine generates the null message to send as next pages.
	3	R	Local device remote fault	When set to 1, the local device signals Remote Faults in the Auto-Negotiation pages. When set to 0 a fault has not occurred.
	4	RW	Force TX nonce value	When set to 1, forces the TX none value to support some UNH-IOL testing modes. When set to 0, operates normally.
	5	RW	Override AN	When set to 1, the override settings defined by the AN_TECH, AN_FEC and AN_PAUSE registers take effect.

Word Addr	Bit	R/W	Name	Description
0xC1	0	RW	Reset AN	When set to 1, resets all the 10GBASE-KR Auto-Negotiation state machines. This bit is self-clearing.
	4	RW	Restart AN TX SM	When set to 1, restarts the 10GBASE-KR TX state machine. This bit self clears. This bit is active only when the TX state machine is in the AN state. For more information, refer to bit 7.0.9 in Clause 73.8 Management Register Requirements of *IEEE 802.3ap-2007*.
	8	RW	AN Next Page	When asserted, new next page info is ready to send. The data is in the XNP TX registers. When 0, the TX interface sends null pages. This bit self clears. Next Page (NP) is encoded in bit D15 of Link Codeword. For more information, refer to Clause 73.6.9 and bit 7.16.15 of Clause 45.2.7.6 of *IEEE 802.3ap-2007*.
0xC2	1	RO	AN page received	When set to 1, a page has been received. When 0, a page has not been received. The current value clears when the register is read. For more information, refer to bit 7.1.6 in Clause 73.8 of *IEEE 802.3ap-2007*.
	2	RO	AN Complete	When asserted, Auto-Negotiation has completed. When 0, Auto-Negotiation is in progress. For more information, refer to bit 7.1.5 in Clause 73.8 of *IEEE 802.3ap-2007*.
	3	RO	AN ADV Remote Fault	When set to 1, fault information has been sent to the link partner. When 0, a fault has not occurred. The current value clears when the register is read. Remote Fault (RF) is encoded in bit D13 of the base Link Codeword. For more information, refer to Clause 73.6.7 of and bit 7.16.13 of *IEEE 802.3ap-2007*.
	4	RO	AN RX SM Idle	When set to 1, the Auto-Negotiation state machine is in the idle state. Incoming data is not Clause 73 compatible. When 0, the Auto-Negotiation is in progress.

Word Addr	Bit	R/W	Name	Description
	5	RO	AN Ability	When set to 1, the transceiver PHY is able to perform Auto-Negotiation. When set to 0, the transceiver PHY i s not able to perform Auto-Negotiation. If your variant includes Auto-Negotiation, this bit is tied to 1. For more information, refer to bits 7.1.3 and 7.48.0 of Clause 45 of *IEEE 802.3ap-2007*.
	6	RO	AN Status	When set to 1, link is up. When 0, the link is down. The current value clears when the register is read. For more information, refer to bit 7.1.2 of Clause 45 of *IEEE 802.3ap-2007*.
	7	RO	LP AN Ability	When set to 1, the link partner is able to perform Auto-Negotiation. When 0, the link partner is not able to perform Auto-Negotiation. For more information, refer to bit 7.1.0 of Clause 45 of *IEEE 802.3ap-2007*.
	8	RO	Enable FEC	When asserted, indicates that auto-negotiation is complete and that communicate includes FEC. For more information refer to *Clause 7.48.4*.
	9	RO	Seq AN Failure	When set to 1, a sequencer Auto-Negotiation failure has been detected. When set to 0, a Auto-Negotiation failure has not been detected.
	17:12	RO	KR AN Link Ready[5:0]	Provides a one-hot encoding of an_receive_idle = true and link status for the supported link as described in Clause 73.10.1. The following encodings are defined: • 6'b000000: 1000BASE-KX • 6'b000001: Reserved • 6'b000100: 10GBASE-KR • 6'b001000: Reserved • 6'b010000: Reserved • 6'b100000: Reserved
0xC3	15:0	RW	User base page low	The Auto-Negotiation TX state machine uses these bits if the AN base pages ctrl bit is set. The following bits are defined:

Word Addr	Bit	R/W	Name	Description
				• [4:0]: Selector • [9:5]: Echoed nonce which are set by the state machine • [12:10]: Pause bits • [13]: Remote Fault bit • [14]: ACK which is controlled by the SM • [15]: Next page bit Bit 49, the PRBS bit, is generated by the Auto-Negotiation TX state machine.
	21:16	RW	Override AN_TECH[5:0]	Specifies an AN_TECH value to override. The following encodings are defined: • [16]: AN_TECH[0] = 1000Base-KX • [17]: AN_TECH[1] = XAUI • [18]: AN_TECH[2] = 10GBASE-KR • [19]: AN_TECH[3] = 40G • [20]: AN_TECH[4] = CR-4 • [21]: AN_TECH[5] = 100G You must write 0xC0[5] to 1'b1 for these overrides to take effect.
	25:24	RW	Override AN_FEC[1:0]	Specifies an AN_FEC value to override. The following encodings are defined: • [24]: AN_ FEC [0] = Capability • [25]: AN_ FEC [1] = Request You must write 0xC0[5] to 1'b1 for these overrides to take effect.
	30:28	RW	Override AN_PAUSE[2:0]	Specifies an AN_PAUSE value to override. The following encodings are defined: • [28]: AN_PAUSE[0] = Pause Ability • [29]: AN_PAUSE[1] = Asymmetric Direction • [30]: AN_PAUSE[2] = Reserved Need to set 0xC0 bit-5 to take effect.

Word Addr	Bit	R/W	Name	Description
0xC4	31:0	RW	User base page high	The Auto-Negotiation TX state machine uses these bits if the Auto-Negotiation base pages ctrl bit is set. The following bits are defined: • [4:0]: Correspond to bits 20:16 which are TX nonce bits. • [29:5]: Correspond to page bit 45:21 which are the technology ability. Bit 49, the PRBS bit, is generated by the Auto-Negotiation TX state machine.
0xC5	15:0	RW	User Next page low	The Auto-Negotiation TX state machine uses these bits if the Auto-Negotiation next pages ctrl bit is set. The following bits are defined: • [11]: Toggle bit • [12]: ACK2 bit • [13]: Message Page (MP) bit • [14]: ACK controlled by the state machine • [15]: Next page bit For more information, refer to Clause 73.7.7.1 Next Page encodings of *IEEE 802.3ap-2007*. Bit 49, the PRBS bit, is generated by the Auto-Negotiation TX state machine.
0xC6	31:0	RW	User Next page high	The Auto-Negotiation TX state machine uses these bits if the Auto-Negotiation next pages ctrl bit is set. Bits [31:0] correspond to page bits [47:16]. Bit 49, the PRBS bit, is generated by the Auto-Negotiation TX state machine.
0xC7	15:0	RO	LP base page low	The AN RX state machine received these bits from the link partner. The following bits are defined: • [4:0] Selector • [9:5] Echoed Nonce which are set by the state machine • [12:10] Pause bits • [12]: ACK2 bit • [13]: RF bit • [14]: ACK controlled by the state machine • [15]: Next page bit

Word Addr	Bit	R/W	Name	Description
0xC8	31:0	RO	`LP base page high`	The AN RX state machine received these bits from the link partner. The following bits are defined: • [31:30]: Reserved • [29:5]: Correspond to page bits [45:21] which are the technology ability • [4:0]: Correspond to bits [20:16] which are TX Nonce bits
0xC9	15:0	RO	`LP Next page low`	The AN RX state machine receives these bits from the link partner. The following bits are defined: • [15]: Next page bit • [14]: ACK which is controlled by the state machine • [13]: MP bit • [12] ACK2 bit • [11] Toggle bit For more information, refer to Clause 73.7.7.1 Next Page encodings of IEEE 802.3ap-2007.
0xCA	31:0	RO	`LP Next page high`	The AN RX state machine receives these bits from the link partner. Bits [31:0] correspond to page bits [47:16]
0xCB	24:0	RO	`AN LP ADV Tech_A[24:0]`	Received technology ability field bits of Clause 73 Auto-Negotiation. The 10GBASE-KR PHY supports A0 and A2. The following protocols are defined: • A0 1000BASE-KX • A1 10GBASE-KX4 • A2 10GBASE-KR • A3 40GBASE-KR4 • A4 40GBASE-CR4 • A5 100GBASE-CR10 • A24:6 are reserved For more information, refer to Clause 73.6.4 and AN LP base page ability registers (7.19-7.21) of Clause 45 of *IEEE 802.3ap-2007*.
	26:25	RO	`AN LP ADV FEC_F[1:0]`	Received FEC ability bits. FEC [F0:F1] is encoded in bits D46:D47 of the base Link Codeword as described in Clause 73 AN, 73.6.5. Bit[26] corresponding to F1 is the request bit. Bit[25] corresponding to F0 is the FEC ability bit.

Word Addr	Bit	R/W	Name	Description
	27	RO	AN LP ADV Remote Fault	Received Remote Fault (RF) ability bits. RF is encoded in bit D13 of the base link codeword in Clause 73 AN. For more information, refer to Clause 73.6.7 and bits AN LP base page ability register AN LP base page ability registers (7.19-7.21) of Clause 45 of *IEEE 802.3ap-2007*.
	30:28	RO	AN LP ADV Pause Ability_C[2:0]	Received pause ability bits. Pause (C0:C1) is encoded in bits D11:D10 of the base link codeword in Clause 73 AN as follows: • C0 is the same as PAUSE as defined in Annex 28B • C1 is the same as ASM_DIR as defined in Annex 28B • C2 is reserved For more information, refer to bits AN LP base page ability registers (7.19-7.21) of Clause 45 of *IEEE 802.3ap-2007*.
0xD0	0	RW	Link Training enable	When 1, enables the 10GBASE-KR start-up protocol. When 0, disables the 10GBASE-KR start-up protocol. The default value is 1. For more information, refer to Clause 72.6.10.3.1 and 10GBASE-KR PMD control register bit (1.150.1) of *IEEE 802.3ap-2007*.
	1	RW	dis_max_wait_tmr	When set to 1, disables the LT max_wait_ timer . Used for characterization mode when setting much longer BER timer values.
	2	RW	quick_mode	When set to 1, only the init and preset values are used to calculate the best BER.
	3	RW	pass_one	When set to 1, the BER algorithm considers more than the first local minimum when searching for the lowest BER. The default value is 1.
	7:4	RW	main_step_cnt [3:0]	Specifies the number of equalization steps for each main tap update. There are about 20 settings for the internal algorithm to test. The valid range is 1-15. The default value is 4'b0010.
	11:8	RW	prpo_step_cnt [3:0]	Specifies the number of equalization steps for each pre- and post- tap update. From 16-31 steps are possible. The default value is 4'b0001.

Word Addr	Bit	R/W	Name	Description
	14:12	RW	`equal_cnt [2:0]`	Adds hysteresis to the error count to avoid local minimums. The default value is 3'b010. The following encodings are defined: • 3'b000: 0 • 3'b001: 1 • 3'b010: 2 • 3'b100: 4 • 3'b101: 8 • 3'b110: 16
	15	RW	`disable initialize PMA on max_wait_ timeout`	When set to 1, does not initialize the PMA V_{OD}, pretap, posttap values upon entry into the Training_Failure state as defined in *Figure 72-5* of *Clause 72.6.10.4.3* of *IEEE 802.3ap-2007*. This failure occurs when the `max_wait_ timer_done` timeout is reached setting the `Link Training failure` bit (0xD2[3]). Used during UNH-IOL testing. When set to 0, initializes the PMA values upon entry into Training_Failure state.
	16	RW	`Ovride LP Coef enable`	When set to 1, overrides the link partner's equalization coefficients; software changes the update commands sent to the link partner TX equalizer coefficients. When set to 0, uses the Link Training logic to determine the link partner coefficients. Used with 0xD1 bit-4 and 0xD4 bits[7:0].
	17	RW	`Ovride Local RX Coef enable`	When set to 1, overrides the local device equalization coefficients generation protocol. When set, the software changes the local TX equalizer coefficients. When set to 0, uses the update command received from the link partner to determine local device coefficients. Used with 0xD1 bit-8 and 0xD4 bits[23:16]. The default value is 1.
	19:18	RMW	Reserved	You should not modify these bits. To update this register, first read the value of this register then change only the value for bits that are not reserved.

Word Addr	Bit	R/W	Name	Description
	22:20	RW	rx_ctle_mode	RX CTLE mode in the Link Training algorithm. The default value is 3'b000. The following encodings are defined: • 3'b000: CTLE tuning in link training is disabled. Retains user set value of CTLE. • 3'b001: Reserved. • 3'b010: Reserved. • 3'b011: CTLE tuning in link training is enabled. • 3'b100 to 3'b111: reserved.
	23	RW	vod_up	When set to 1, V_{OD} is trained to high values. The default is set to 0 to save power and reduce crosstalk on the link.
	26:24	RW	rx_dfe_mode	RX DFE mode in the link training algorithm. The default value is 3'b000. The following bits are defined: • 3'b000: DFE adaptation in link training is disabled • 3'b001: Reserved • 3'b010: DFE is triggered at the end of link training • 3'b011: DFE is triggered at the end of V_{OD}, Post tap and Pre-tap training • 3'b100 to 3'b111: Reserved
	28	RW	max_mode	When set to 1, link training operates in maximum TX equalization mode. Modifies the link training algorithm to settle on the max pretap and max V_{OD} if the BER counter reaches the maximum for all values. Link training settles on the max_post_step for the posttap value.
	31:29	RW	max_post_step	Number of TX posttap steps from the initialization state when in max_mode.

Word Addr	Bit	R/W	Name	Description
0xD1	0	RW	Restart Link training	When set to 1, resets the 10GBASE-KR start-up protocol. When set to 0, continues normal operation. This bit self clears. For more information, refer to the state variable mr_restart_training as defined in Clause 72.6.10.3.1 and 10GBASE-KR PMD control register bit (1.150.0) *IEEE 802.3ap-2007*.
	4	RW	Updated TX Coef new	When set to 1, there are new link partner coefficients available to send. The LT logic starts sending the new values set in 0xD4 bits[7:0] to the remote device. When set to 0, continues normal operation. This bit self clears. Must enable this override in 0xD0 bit16.
	8	RW	Updated RX coef new	When set to 1, new local device coefficients are available. The LT logic changes the local TX equalizer coefficients as specified in 0xD4 bits[23:16]. When set to 0, continues normal operation. This bit self clears. Must enable the override in 0xD0 bit17.
0xD2	0	RO	Link Trained - Receiver status	When set to 1, the receiver is trained and is ready to receive data. When set to 0, receiver training is in progress. For more information, refer to the state variable rx_trained as defined in Clause 72.6.10.3.1 and bit 10GBASE-KR PMD control register bit 10GBASE_KR PMD status register bit (1.151.0) of *IEEE 802.3ap-2007*.
	1	RO	Link Training Frame lock	When set to 1, the training frame delineation has been detected. When set to 0, the training frame delineation has not been detected. For more information, refer to the state variable frame_lock as defined in Clause 72.6.10.3.1 and 10GBASE_KR PMD status register bit 10GBASE_KR PMD status register bit (1.151.1) of *IEEE 802.3ap-2007*.
	2	RO	Link Training Start-up protocol status	When set to 1, the start-up protocol is in progress. When set to 0, start-up protocol has completed. For more information, refer to the state training as defined in Clause 72.6.10.3.1 and 10GBASE_KR PMD status register bit (1.151.2) of *IEEE 802.3ap-2007*.

Word Addr	Bit	R/W	Name	Description
	3	RO	Link Training failure	When set to 1, a training failure has been detected. When set to 0, a training failure has not been detected For more information, refer to the state variable training_failure as defined in Clause 72.6.10.3.1 and bit 10GBASE_KR PMD status register bit (1.151.3) of IEEE 802.3ap-2007.
	4	RO	Link Training Error	When set to 1, excessive errors occurred during Link Training. When set to 0, the BER is acceptable.
	5	RO	Link Training Frame lock Error	When set to 1, indicates a frame lock was lost during Link Training. If the tap settings specified by the fields of 0xD5 are the same as the initial parameter value, the frame lock error was unrecoverable.
	6	RO	CTLE Frame Lock Loss	When set to 1, indicates that fram lock was lost at some point during CTLE link training.
	7	RO	CTLE Tuning Error	When set to 1, indicates that CTLE did not achieve best results because the BER counter reached the maximum value for each step of CTLE tuning.
0xD3	9:0	RW	ber_time_frames	Specifies the number of training frames to examine for bit errors on the link for each step of the equalization settings. Used only when ber_time_k_frames is 0.The following values are defined: • A value of 2 is about 10^3 bytes • A value of 20 is about 10^4 bytes • A value of 200 is about 10^5 bytes The default value for simulation is 2'b11. The default value for hardware is 0.

Word Addr	Bit	R/W	Name	Description
	19:10	RW	ber_time_k_frames	Specifies the number of thousands of training frames to examine for bit errors on the link for each step of the equalization settings. Set *ber_time_m_frames = 0* for time/bits to match the following values: • A value of 3 is about 10^7 bits = about 1.3 ms • A value of 25 is about 10^8 bits = about 11ms • A value of 250 is about 10^9 bits = about 11 0ms The default value for simulation is 0. The default value for hardware is 0xF.
	29:20	RW	ber_time_m_frames	Specifies the number of millions of training frames to examine for bit errors on the link for each step of the equalization settings. Set *ber_time_k_frames = 4'd1000 = 0x3E8* for time/bits to match the following values: • A value of 3 is about 10^{10} bits = about 1.3 seconds • A value of 25 is about 10^{11} bits = about 11 seconds • A value of 250 is about 10^{12} bits = about 110 seconds The default value is 0x0.

0xD4	5:0	RO or RW	LD coefficient update[5:0]	Reflects the contents of the first 16-bit word of the training frame sent from the local device control channel. Normally, the bits in this register are read-only; however, when you override training by setting the Ovride Coef enable control bit, these bits become writeable. The following fields are defined:
				• [5: 4]: Coefficient (+1) update
				  • 2'b11: Reserved
				  • 2'b01: Increment
				  • 2'b10: Decrement
				  • 2'b00: Hold
				• [3:2]: Coefficient (0) update (same encoding as [5:4])
				• [1:0]: Coefficient (-1) update (same encoding as [5:4])

Word Addr	Bit	R/W	Name	Description
				For more information, refer to bit 10G BASE-KR LD coefficient update register bits (1.154.5:0) in Clause 45.2.1.80.3 of *IEEE 802.3ap-2007*.
	6	RO or RW	`LD Initialize Coefficients`	When set to 1, requests the link partner coefficients be set to configure the TX equalizer to its INITIALIZE state. When set to 0, continues normal operation. For more information, refer to 10G BASE-KR LD coefficient update register bits (1.154.12) in Clause 45.2.1.80.3 and Clause 72.6.10.2.3.2 of *IEEE 802.3ap-2007*.
	7	RO or RW	`LD Preset Coefficients`	When set to 1, requests the link partner coefficients be set to a state where equalization is turned off. When set to 0 the link operates normally. For more information, refer to bit 10G BASE-KR LD coefficient update register bit (1.154.13) in Clause 45.2.1.80.3 and Clause 72.6.10.2.3.2 of *IEEE 802.3ap-2007*.

	13:8	RO	`LD coefficient status[5:0]`	Status report register for the contents of the second, 16-bit word of the training frame most recently sent from the local device control channel. The following fields are defined:
				• [5:4]: Coefficient (post-tap)
				  • 2'b11: Maximum
				  • 2'b01: Minimum
				  • 2'b10: Updated
				  • 2'b00: Not updated
				• [3:2]: Coefficient (0) (same encoding as [5:4])
				• [1:0]: Coefficient (pre-tap) (same encoding as [5:4])
				For more information, refer to bit 10G BASE-KR LD status report register bit (1.155.5:0) in Clause 45.2.1.81 of *IEEE 802.3ap-2007*.

Word Addr	Bit	R/W	Name	Description
	14	RO	`Link Training ready - LD Receiver ready`	When set to 1, the local device receiver has determined that training is complete and is prepared to receive data. When set to 0, the local device receiver is requesting that training continue. Values for the receiver ready bit are defined in Clause 72.6.10.2.4.4. For more information refer to For more information, refer to bit 10G BASE-KR LD status report register bit (1.155.15) in Clause 45.2.1.81 of *IEEE 802.3ap-2007*.

	21:16	RO or RW	`LP coefficient update[5:0]`	Reflects the contents of the first 16-bit word of the training frame most recently received from the control channel.
				Normally the bits in this register are read only; however, when training is disabled by setting low the KR Training enable control bit, these bits become writeable. The following fields are defined:
				• [5: 4]: Coefficient (+1) update
				  • 2'b11: Reserved
				  • 2'b01: Increment
				  • 2'b10: Decrement
				  • 2'b00: Hold
				• [3:2]: Coefficient (0) update (same encoding as [5:4])
				• [1:0]: Coefficient (-1) update (same encoding as [5:4])
				For more information, refer to bit 10G BASE-KR LP coefficient update register bits (1.152.5:0) in Clause 45.2.1.78.3 of *IEEE 802.3ap-2007*.

	22	RO or RW	`LP Initialize Coefficients`	When set to 1, the local device transmit equalizer coefficients are set to the INITIALIZE state. When set to 0, normal operation continues. The function and values of the initialize bit are defined in Clause 72.6.10.2.3.2. For more information, refer to bit 10G BASE-KR LP coefficient update register bits (1.152.12) in Clause 45.2.1.78.3 of *IEEE 802.3ap-2007*.

Word Addr	Bit	R/W	Name	Description
	23	RO or RW	LP Preset Coefficients	When set to 1, The local device TX coefficients are set to a state where equalization is turned off. Preset coefficients are used. When set to 0, the local device operates normally. The function and values of the preset bit is defined in 72.6.10.2.3.1. The function and values of the initialize bit are defined in Clause 72.6.10.2.3.2. For more information, refer to bit 10G BASE-KR LP coefficient update register bits (1.152.13) in Clause 45.2.1.78.3 of IEEE 802.3ap-2007.

	29:24	RO	LP coefficient status[5:0]	Status report register reflects the contents of the second, 16-bit word of the training frame most recently received from the control channel: The following fields are defined:
				• [5:4]: Coefficient (+1)
				  • 2'b11: Maximum
				  • 2'b01: Minimum
				  • 2'b10: Updated
				  • 2'b00: Not updated
				• [3:2]: Coefficient (0) (same encoding as [5:4])
				• n [1:0]: Coefficient (-1) (same encoding as [5:4])
				For more information, refer to bit 10G BASE-KR LP status report register bits (1.153.5:0) in Clause 45.2.1.79 of *IEEE 802.3ap-2007*.

	30	RO	LP Receiver ready	When set to 1, the link partner receiver has determined that training is complete and is prepared to receive data. When set to 0, the link partner receiver is requesting that training continue. Values for the receiver ready bit are defined in Clause 72.6.10.2.4.4. For more information, refer to bit 10G BASE-KR LP status report register bits (1.153.15) in Clause 45.2.1.79 of *IEEE 802.3ap-2007*.

Word Addr	Bit	R/W	Name	Description
0xD5	5:0	R	LT V_{OD} setting	Stores the most recent V_{OD} setting that LT specified using the Transceiver Reconfiguration Controller IP core. It reflects Link Partner commands to fine-tune the V_{OD}.
	12:8	R	LT Post-tap setting	Stores the most recent post-tap setting that LT specified using the Transceiver Reconfiguration Controller IP core. It reflects Link Partner commands to fine-tune the TX pre-emphasis taps.
	19:16	R	LT Pre-tap setting	Stores the most recent pre-tap setting that LT specified using the Transceiver Reconfiguration Controller IP core. It reflects Link Partner commands to fine-tune the TX pre-emphasis taps.
	23:20	R	RXEQ CTLE Setting	Stores the most recent CTLE setting sent to the Transceiver Reconfiguration IP Core during RX Equalization.
	25:24	R	RXEQ CTLE Mode	Stores the most recent CTLE mode that CTLE specified using the Transceiver Reconfiguration IP Core during RX Equalization.
	27:26	R	RXEQ DFE Mode	Stores the most recent DFE setting sent to the Transceiver Reconfiguration IP Core during RX Equalization.
0xD6	5:0	RW	LT VODMAX ovrd	Override value for the **VMAXRULE** parameter. When enabled, this value substitutes for the **VMAXRULE** to allow channel-by-channel override of the device settings. This only effects the local device TX output for the channel specified. This value must be greater than the **INITMAINVAL** parameter for proper operation. Note this will also override the **PREMAINVAL** parameter value.
	6	RW	LT VODMAX ovrd Enable	When set to 1, enables the override value for the **VMAXRULE** parameter stored in the LT VODMAX ovrd register field.

Word Addr	Bit	R/W	Name	Description
	13:8	RW	LT VODMin ovrd	Override value for the **VODMINRULE** parameter. When enabled, this value substitutes for the **VMINRULE** to allow channel-by-channel override of the device settings. This override only effects the local device TX output for this channel. The value to be substituted must be less than the **INITMAINVAL** parameter and greater than the **VMINRULE** parameter for proper operation.
	14	RW	LT VODMin ovrd Enable	When set to 1, enables the override value for the **VODMINRULE** parameter stored in the LT VODMin ovrd register field.
	20:16	RW	LT VPOST ovrd	Override value for the **VPOSTRULE** parameter. When enabled, this value substitutes for the **VPOSTRULE** to allow channel-by-channel override of the device settings. This override only effects the local device TX output for this channel. The value to be substituted must be greater than the **INITPOSTVAL** parameter for proper operation.
	21	RW	LT VPOST ovrd Enable	When set to 1, enables the override value for the **VPOSTRULE** parameter stored in the LT VPOST ovrd register field.
	27:24	RW	LT VPre ovrd	Override value for the **VPRERULE** parameter. When enabled, this value substitutes for the **VPOSTRULE** to allow channel-by-channel override of the device settings. This override only effects the local device TX output for this channel. The value greater than the **INITPREVAL** parameter for proper operation.
	28	RW	LT VPre ovrd Enable	When set to 1, enables the override value for the **VPRERULE** parameter stored in the LT VPre ovrd register field.

PMA Registers

The PMA registers allow you to reset the PMA and provide status information.

Table 4-21: PMA Registers - Reset and Status

The following PMA registers allow you to reset the PMA and provide status information.

Addr	Bit	Access	Name	Description
0x22	0	RO	pma_tx_pll_is_locked	Indicates that the TX PLL is locked to the input reference clock.
0x44	1	RW	reset_tx_digital	Writing a 1 causes the internal TX digital reset signal to be asserted. You must write a 0 to clear the reset condition.
	2	RW	reset_rx_analog	Writing a 1 causes the internal RX analog reset signal to be asserted. You must write a 0 to clear the reset condition.
	3	RW	reset_rx_digital	Writing a 1 causes the internal RX digital reset signal to be asserted. You must write a 0 to clear the reset condition.
0x61	[31:0]	RW	phy_serial_loopback	Writing a 1 puts the channel in serial loopback mode.
0x64	[31:0]	RW	pma_rx_set_locktodata	When set, programs the RX CDR PLL to lock to the incoming data.
0x65	[31:0]	RW	pma_rx_set_locktoref	When set, programs the RX clock data recovery (CDR) PLL to lock to the reference clock.
0x66	[31:0]	RO	pma_rx_is_lockedtodata	When asserted, indicates that the RX CDR PLL is locked to the RX data, and that the RX CDR has changed from LTR to LTD mode.
0x67	[31:0]	RO	pma_rx_is_lockedtoref	When asserted, indicates that the RX CDR PLL is locked to the reference clock.

Table 4-22: PMA Registers - TX and RX Serial Data Interface

The following PMA registers allow you to customize the TX and RX serial data interface

Address	Bit	R/W	Name	Description
0xA8	0	RW	tx_invpolarity	When set to 1, the TX interface inverts the polarity of the TX data. Inverted TX data is output from the 8B/10B encoder.
	1	RW	rx_invpolarity	When set to 1, the RX channels inverts the polarity of the received data. Inverted RX data is input to the 8B/10B decoder.
	2	RW	rx_bitreversal_enable	When set to 1, enables bit reversal on the RX interface. The RX data is input to the word aligner.
	3	RW	rx_bytereversal_enable	When set, enables byte reversal on the RX interface. The RX data is input to the byte deserializer.
	4	RW	force_electrical_idle	When set to 1, forces the TX outputs to electrical idle.

Address	Bit	R/W	Name	Description
0xA9	0	R	rx_syncstatus	When set to 1, indicates that the word aligner is synchronized to incoming data.
	1	R	rx_patterndetect	When set to 1, indicates the 1G word aligner has detected a comma.
	2	R	rx_rlv	When set to 1, indicates a run length violation.
	3	R	rx_rmfifodatainserted	When set to 1, indicates the rate match FIFO inserted code group.
	4	R	rx_rmfifodatadeleted	When set to 1, indicates that rate match FIFO deleted code group.
	5	R	rx_disperr	When set to 1, indicates an RX 8B/10B disparity error.
	6	R	rx_errdetect	When set to 1, indicates an RX 8B/10B error detected.

PCS Registers

Table 4-23: PCS Registers

These registers provide PCS status information.

Addr	Bit	Access	Name	Description
0x800	31:0	RW	Indirect_addr	Because the PHY implements a single channel, this register must remain at the default value of 0 to specify logical channel 0.
0x881	2	RW	RCLR_ERRBLK_CNT	Error Block Counter clear register. When set to 1, clears the RCLR_ERRBLK_CNT register. When set to 0, normal operation continues.
	3	RW	RCLR_BER_COUNT	BER Counter clear register. When set to 1, clears the RCLR_BER_COUNT register. When set to 0, normal operation continues.
0x882	1	RO	HI_BER	High BER status. When set to 1, the PCS is reporting a high BER. When set to 0, the PCS is not reporting a high BER.
	2	RO	BLOCK_LOCK	Block lock status. When set to 1, the PCS is locked to received blocks. When set to 0, the PCS is not locked to received blocks.
	3	RO	TX_FULL	When set to 1, the TX_FIFO is full.
	4	RO	RX_FULL	When set to 1, the RX_FIFO is full.
	5	RO	RX_SYNC_HEAD_ERROR	When set to 1, indicates an RX synchronization error.
	6	RO	RX_SCRAMBLER_ERROR	When set to 1, indicates an RX scrambler error.
	7	RO	Rx_DATA_READY	When set to 1, indicates the PHY is ready to receive data.

Table 4-24: Pattern Generator Registers

1G/10Gbps Ethernet PHY IP core supports the 10G PCS Pattern Generator.

Offset	Bits	R/W	Name	Description
0x12D	[15:0]	R/W	Seed A for PRP	Bits [15:0] of seed A for the pseudo-random pattern.
0x12E	[15:0]			Bits [31:16] of seed A for the pseudo-random pattern.
0x12F	[15:0]			Bits [47:21] of seed A for the pseudo-random pattern.
0x130	[9:0]			Bits [57:48] of seed A for the pseudo-random pattern.
0x131	[15:0]	R/W	Seed B for PRP	Bits [15:0] of seed B for the pseudo-random pattern.
0x132	[15:0]			Bits [31:16] of seed B for the pseudo-random pattern.
0x133	[15:0]			Bits [47:32] of seed B for the pseudo-random pattern.
0x134	[9:0]			Bits [57:48] of seed B for the pseudo-random pattern.

Offset	Bits	R/W	Name	Description
0x135	[15:12]	R/W	Square Wave Pattern	Specifies the number of consecutive 1s and 0s. The following values are available: 1, 4, 5, 6, 8, and 10.
	[10]	R/W	TX PRBS 7 Enable	Enables the PRBS-7 polynomial in the transmitter.
	[8]	R/W	TX PRBS 23 Enable	Enables the PRBS-23 polynomial in the transmitter.
	[6]	R/W	TX PRBS 9 Enable	Enables the PRBS-9 polynomial in the transmitter.
	[4]	R/W	TX PRBS 31 Enable	Enables the PRBS-31 Polynomial in the transmitter.
	[3]	R/W	TX Test Enable	Enables the pattern generator in the transmitter.
	[1]	R/W	TX Test Pattern Select	Selects between the square wave or pseudo-random pattern generator. The following encodings are defined: • 1'b1: Square wave • 1'b0: Pseudo-random pattern or PRBS
	[0]	R/W	Data Pattern Select	Selects the data pattern for the pseudo-random pattern. The following encodings are defined: • 1'b1: Two Local Faults. Two, 32-bit ordered sets are XORd with the pseudo-random pattern. • 1'b0: All 0's
0x137	[2]	R/W	TX PRBS Clock Enable	Enables the transmitter PRBS clock.
	[1]	R/W	TX Square Wave Clock Enable	Enables the square wave clock.

Offset	Bits	R/W	Name	Description
0x15E	[14]	R/W	RX PRBS 7 Enable	Enables the PRBS-7 polynomial in the receiver.
	[13]	R/W	RX PRBS 23 Enable	Enables the PRBS-23 polynomial in the receiver.
	[12]	R/W	RX PRBS 9 Enable	Enables the PRBS-9 polynomial in the receiver.
	[11]	R/W	RX PRBS 31 Enable	Enables the PRBS-31 polynomial in the receiver.
	[10]	R/W	RX Test Enable	Enables the PRBS pattern verifier in the receiver.
0x164	[10]	R/W	RX PRBS Clock Enable	Enables the receiver PRBS Clock.
0x169	[0]	R/W	RX Test Pattern Select	Selects between a square wave or pseudo-random pattern. The following encodings are defined: • 1'b1: Square wave • 1'b0: Pseudo-random pattern or PRBS

Creating a 10GBASE-KR Design

Here are the steps you must take to create a 10GBASE-KR design using this PHY.

1. Generate the 10GBASE-KR PHY with the required parameterization.
2. Generate a Transceiver Reconfiguration Controller with the correct number of reconfiguration interfaces based on the number of channels you are using. This controller is connected to all the transceiver channels. It implements the reconfiguration process.
3. Generate a Transceiver Reset Controller.
4. Create arbitration logic that prioritizes simultaneous reconfiguration requests from multiple channels. This logic should also acknowledge the channel being serviced causing the requestor to deassert its request signal.
5. Create a state machine that controls the reconfiguration process. The state machine should:
   a. Receive the prioritized reconfiguration request from the arbiter
   b. Put the Transceiver Reconfiguration Controller into MIF streaming mode.
   c. Select the correct MIF and stream it into the appropriate channel.
   d. Wait for the reconfiguration process to end and provide status signal to arbiter.
6. Generate one ROM for each required configuration.
7. Create a MIF for each configuration and associate each MIF with a ROM created in Step 6. For example, create a MIF for 1G with 1588 , a MIF for 10G with 1588, and a MIF for AN/LT. AN/LT MIF

is is used to reconfigure the PHY into low latency mode during AN/LT. These MIFs are the three configurations used in the MIF streaming process. The example design contains five required MIFs (1G, 10G, 1G with 1588,10G with 1588 and AN/LT). Altera recommends that you use these MIFs even if you are not using the example design.

8. Generate a fractional PLL to create the 156.25 MHz XGMII clock from the 10G reference clock.
9. Instantiate the PHY in your design based on the required number of channels.
10. To complete the system, connect all the blocks.

Related Information

MIF Generation on page 17-37

Editing a 10GBASE-KR MIF File

This topic shows how to edit a 10GBASE-KR MIF file to change between 1G and 10Gb Ethernet.

The MIF format contains all bit settings for the transceiver PMA and PCS. Because the 10GBASE-KR PHY IP Core only requires PCS reconfiguration for a rate change, the PMA settings should not change. Removing the PMA settings from the MIF file also prevents an unintended overwrite of PMA parameters set through other assignments. A few simple edits to the MIF file removes the PMA settings. Complete the following steps to to remove PMA settings from the MIF file:

1. Replace line 17 with `"13: 0001000000010110; -- PMA - RX changed to removed CTLE".`
2. Replace line 20 with `"16: 0010100000011001; -- PMA - RX continued".`
3. Replace line 4 with `"4: 0001000000000000; -- PMA - TX".`
4. Remove lines 7-10. These lines contain the TX settings (V_{OD}, post-tap, pre-tap).
5. Renumber the lines starting with the old line 11.
6. Change the depth at the top of the file from 168 to 164.

Example 4-1: Edits to a MIF to Remove PMA Settings

```
MIF_A
WIDTH=16;
DEPTH=168;
ADDRESS_RADIX=UNS
DATA_RADIX=BIN;

0:| 0000000000100001; -- Start of MIF opcode - FAMILY - Stratix V
1: 0000000000000010; -- Type of MIF opcode
2: 0000000000000011; -- RefClk switch opcode
3: 0000000000000100; -- CGB PLL switch opcode
4: 0011000000000000; -- PMA - TX
5: 0010100001100000;
6: 0000001111100000;
7: 1000000111010100;
8: 0000000000000000;
9: 0000001111000000;
10: 0100000000110010;
11: 0010100000001100; -- PMA - RX (PLL section
12: 0110000000000000;
13: 0000000000000000;
14: 0000110101100000;
15: 0000000000000010;
16: 0100000000000000;
17: 0100000000010110;-- PMA - RX
18: 0000111010010110;
19: 0100000000010001;
20: 0000000000000000;
21: 0000000000000000;
22: 0010110000000000;
23: 0000000110000001;
24: 0100000100001111;
25: 0001000101110011;
26: 0000100000110000; -- PMA -- COM
```

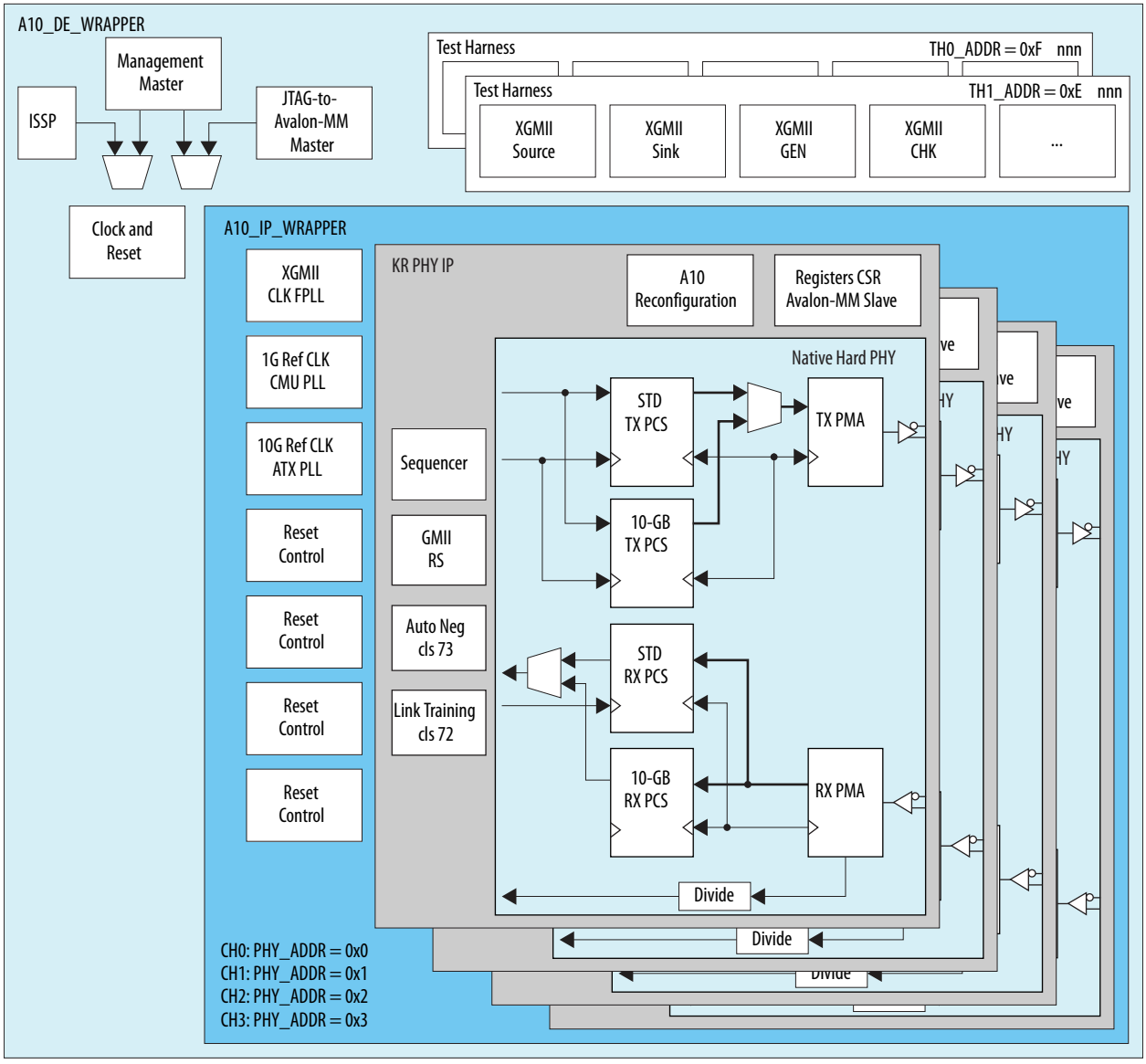
```
MIF_B;
WIDTH=16;
DEPTH=164;
ADDRESS_RADIX=UNS
DATA_RADIX=BIN;

0: 0000000000100001; -- Start of MIF opcode - FAMILY- Stratix V
1: 0000000000000010; -- Type of MIF opcode
2: 0000000000000011; -- RefClk switch opcode
3: 0000000000000100; -- CGB PLL switch opcode
4: 0001000000000000; -- PMA - TX
5: 0010100001100000;
6: 0000001111100000;

7: 0010100000001100; -- PMA - RX (PLL section)
8: 0110000000000000;
9: 0000000000000000;
10: 0000110101100000;
11: 0000000000000010;
12: 0100000000000000;
13: 0001000000010110; -- PMA - RX removed CTLE
14: 0000111010010110;
15: 0100000000010001;
16: 0010100000011001; -- PMA - RX continued
17: 0000000000000000;
18: 0010110000000000;
19: 0000000110000001;
20: 0100000100001111;
21: 0001000101110011;
22: 0000100000110000; -- PMA -- COM
```

Design Example

Figure 4-13: PHY-Only Design Example with Two Backplane Ethernet and Two Line-Side (1G/10G) Ethernet Channels



Related Information

- **Arria 10 Transceiver PHY Design Examples**
- **10-Gbps Ethernet MAC IP Function User Guide.**
  For more information about latency in the MAC as part of the Precision Time Protocol implementation.

SDC Timing Constraints

The SDC timing constraints and approaches to identify false paths listed for Stratix V Native PHY IP apply to all other transceiver PHYs listed in this user guide. Refer to *SDC Timing Constraints of Stratix V Native PHY* for details.

Related Information

SDC Timing Constraints of Stratix V Native PHY on page 13-72
This section describes SDC examples and approaches to identify false timing paths.

Acronyms

This table defines some commonly used Ethernet acronyms.

Table 4-25: Ethernet Acronyms

Acronym	Definition
AN	Auto-Negotiation in Ethernet as described in Clause 73 of IEEE 802.3ap-2007.
BER	Bit Error Rate.
DME	Differential Manchester Encoding.
FEC	Forward error correction.
GMII	Gigabit Media Independent Interface.
KR	Short hand notation for Backplane Ethernet with 64b/66b encoding.
LD	Local Device.
LT	Link training in backplane Ethernet Clause 72 for 10GBASE-KR and 40GBASE-KR4.
LP	Link partner, to which the LD is connected.
MAC	Media Access Control.
MII	Media independent interface.
OSI	Open System Interconnection.
PCS	Physical Coding Sublayer.
PHY	Physical Layer in OSI 7-layer architecture, also in Intel device scope is: PCS + PMA.
PMA	Physical Medium Attachment.
PMD	Physical Medium Dependent.
SGMII	Serial Gigabit Media Independent Interface.
WAN	Wide Area Network.
XAUI	10 Gigabit Attachment Unit Interface.

1G/10Gbps Ethernet PHY IP Core 5

2020.06.02

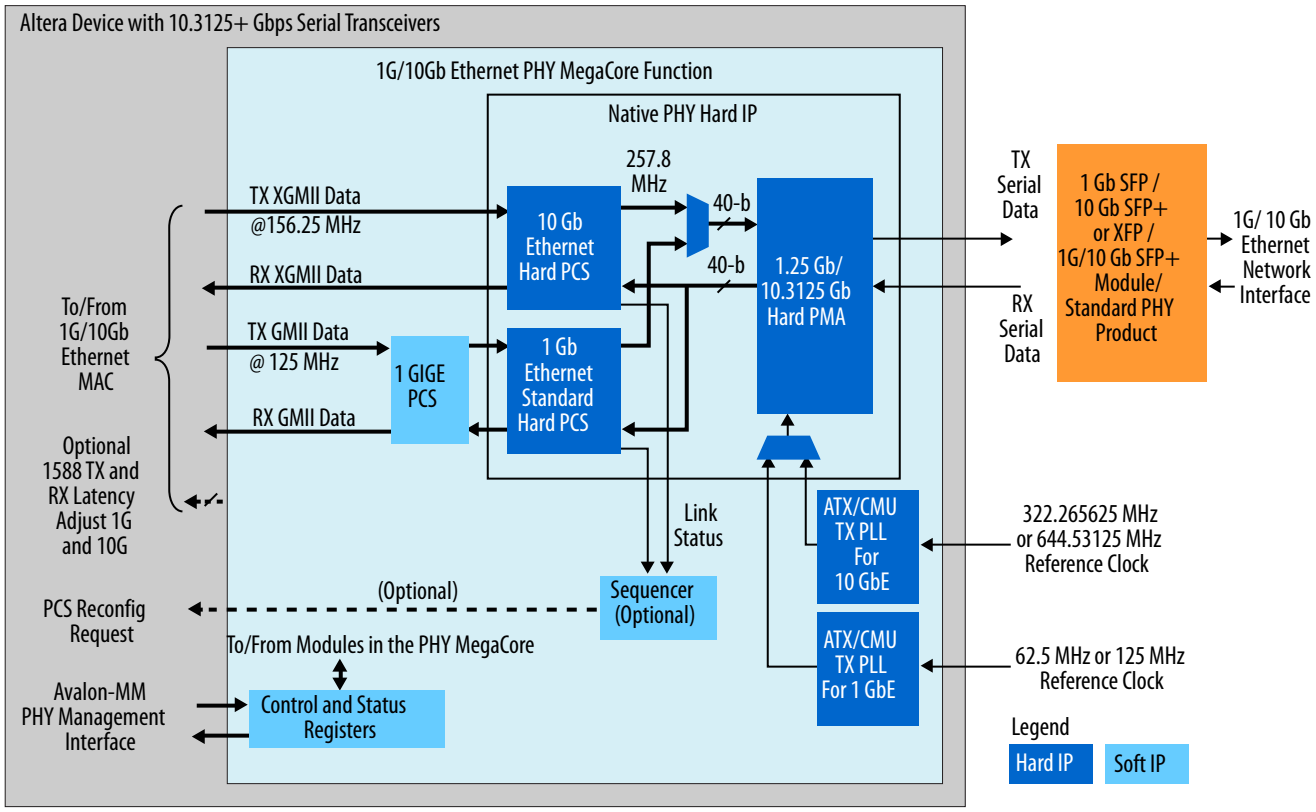
UG-01080 Subscribe Send Feedback

The 1G/10 Gbps Ethernet PHY MegaCore® (1G/10GbE) function allows you to instantiate both the Standard PCS and the higher performance 10G PCS and a PMA. The Standard PCS implements the 1 GbE protocol as defined in Clause 36 of the *IEEE 802.3 2005 Standard* and also supports auto-negotiation as defined in Clause 37 of the *IEEE 802.3 2005 Standard* standard. The 10G PCS implements the 10 Gb Ethernet protocol as defined in *IEEE 802.3 2005 Standard*.

You can switch dynamically between the 1G and 10G PCS using the Altera Transceiver Reconfiguration Controller IP Core to reprogram the core. This Ethernet core targets 1G/10GbE applications including network interfaces to 1G/10GbE dual speed SFP+ pluggable modules, 1G/10GbE 10GBASE-T copper external PHY devices to drive CAT-6/7 shielded twisted pair cables, and chip-to-chip interfaces.

The following figure shows the top-level modules of the 1G/10GbE PHY IP Core. As this figure indicates, the 1G/10 Gbps Ethernet PHY connects to a separately instantiated MAC. The 10G PCS receives and transmits XGMII data. The Standard PCS receives and transmits GMII data. An Avalon Memory-Mapped (Avalon-MM) slave interface provides access to PCS registers. the PMA receives and transmits serial data.

Figure 5-1: Level Modules of the 1G/10GbE PHY MegaCore Function



An Avalon® Memory-Mapped (Avalon-MM) slave interface provides access to the 1G/10GbE PHY IP Core registers. These registers control many of the functions of the other blocks. Many of these bits are defined in *Clause 45 of IEEE Std 802.3ap-2007*.

Related Information

- **IEEE Std 802.3ap-2005 Standard**
- **IEEE Std 802.3ap-2007 Standard**

1G/10GbE PHY Release Information

This topic provides information about this release of the 1G/10GbE PHY IP Core.

Table 5-1: 1G/10GbE Release Information

Item	Description
Version	13.1
Release Date	November 2013
Ordering Codes	IP-1G10GBASER PHY (primary)

Item	Description
Product ID	0106
Vendor ID	6AF7

Device Family Support

IP cores provide either final or preliminary support for target Altera device families. These terms have the following definitions:

- *Final support*—Verified with final timing models for this device.
- *Preliminary support*—Verified with preliminary timing models for this device.

Table 5-2: Device Family Support

Device Family	Support	Supported Speed Grades
Arria V GZ devices–Hard PCS and PMA	Final	I3L, C3, I4, C4
Stratix V devices–Hard PCS and PMA	Final	All speed grades except I4 and C4
Other device families	No support	

Altera verifies that the current version of the Intel Quartus Prime software compiles the previous version of each IP core. Any exceptions to this verification are reported in the **MegaCore IP Library Release Notes and Errata**. Altera does not verify compilation with IP core versions older than the previous release.

Note: For speed grade information, refer to *DC and Switching Characteristics for Stratix V Devices* in the *Stratix V Device Datasheet*.

Related Information

Stratix V Device Datasheet

1G/10GbE PHY Performance and Resource Utilization

This topic provides performance and resource utilization for the IP core in Arria V GZ and Stratix V devices.

The following table shows the typical expected resource utilization for selected configurations using the current version of the Intel Quartus Prime software targeting a Stratix V GT (5SGTMC7K2F40C2) device. The numbers of ALMs and logic registers are rounded up to the nearest 100. Resource utilization numbers reflect changes to the resource utilization reporting starting in the Quartus II software v12.1 release 28 nm device families and upcoming device families.

Table 5-3: 1G/10 GbE PHY Performance and Resource Utilization

PHY Module Options	ALMs	M20K Memory	Logic Registers
1GbE/10GbE - 1GbE only	300	0	600
1GbE/10GbE - 1GbE only with Sequencer	400	0	700
1GbE/10GbE - 1GbE/10GbE with 1588	1000	4	2000
1GbE/10GbE - 1GbE/10GbE with 1588 and Sequencer	1100	4	2000

Parameterizing the 1G/10GbE PHY

The 1G/10GbE PHY IP Core is available for the **Arria V GZ** and **Stratix V** device families. The IP variant allows you specify either the **Backplane-KR** or **1Gb/10Gb Ethernet** variant. When you select the **Backplane-KR** variant, the **Link Training (LT)** and **Auto Negotiation (AN)** tabs appear. The **1Gb/10Gb Ethernet** variant (1G/10GbE) does not implement LT and AN parameters.

Complete the following steps to configure the 1G/10GbE PHY IP Core in the MegaWizard Plug-In Manager:

1. Under **Tools** > **IP Catalog**, select the device family of your choice.
2. Under **Tools** > **IP Catalog** > **Interfaces** > **Ethernet** select **1G10GbE and 10GBASE-KR PHY**.
3. Use the tabs on the MegaWizard Plug-In Manager to select the options required for the protocol.
4. Refer to the topics listed as Related Links to understand and specify 1G/10GbE parameters:
5. Click **Finish** to generate your parameterized 1G/10GbE PHY IP Core.

Related Information

- **Speed Detection Parameters** on page 4-9
- **PHY Analog Parameters** on page 4-10
- **1G/10GbE PHY Interfaces** on page 5-7

1GbE Parameters

The 1GbE parameters allow you to specify options for the 1GbE mode.

Table 5-4: 1Gb Ethernet Parameters

Parameter Name	Options	Description
Enable 1Gb Ethernet protocol	On/Off	When you turn this option **On**, the core includes the GMII interface and related logic.
Expose MII interface	On/Off	When you turn this option **On**, the core exposes the MII interface and related logic.

Parameter Name	Options	Description
Enable IEEE 1588 Precision Time Protocol	On/Off	When you turn this option **On**, the core includes a module in the PCS to implement the IEEE 1588 Precision Time Protocol.
PHY ID (32 bit)	`32-bit value`	An optional 32-bit value that serves as a unique identifier for a particular type of PCS. The identifier includes the following components: • Bits 3-24 of the Organizationally Unique Identifier (OUI) assigned by the IEEE • 6-bit model number • 4-bit revision number If unused, do not change the default value which is 0x00000000.
PHY Core version (16 bits)	`16-bit value`	This is an optional 16-bit value identifies the PHY core version.
Reference clock frequency	125.00 MHz 62.50 MHz	Specifies the clock frequency for the 1GBASE-KR PHY IP Core. The default is 125 MHz.

Related Information

1588 Delay Requirements on page 3-29

Speed Detection Parameters

Selecting the speed detection option gives the PHY the ability to detect to link partners that support 1G/10GbE but have disabled Auto-Negotiation. During Auto-Negotiation, if AN cannot detect Differential Manchester Encoding (DME) pages from a link partner, the Sequencer reconfigures to 1GbE and 10GbE modes (Speed/Parallel detection) until it detects a valid 1G or 10GbE pattern.

Table 5-5: Speed Detection

Parameter Name	Options	Description
Enable automatic speed detection	**On** **Off**	When you turn this option **On**, the core includes the Sequencer block that sends reconfiguration requests to detect 1G or 10GbE when the Auto Negotiation block is not able to detect AN data.
Avalon-MM clock frequency	`100-162 MHz`	Specifies the clock frequency for `phy_mgmt_clk`.

Parameter Name	Options	Description
Link fail inhibit time for 10Gb Ethernet	`504 ms`	Specifies the time before `link_status` is set to FAIL or OK. A link fails if the `link_fail_inhibit_time` has expired before `link_status` is set to OK. The legal range is 500-510 ms. For more information, refer to *"Clause 73 Auto Negotiation for Backplane Ethernet"* in *IEEE Std 802.3ap-2007*.
Link fail inhibit time for 1Gb Ethernet	`40-50 ms`	Specifies the time before `link_status` is set to FAIL or OK . A link fails if the `link_fail_inhibit_time` has expired before `link_status` is set to OK. The legal range is 40-50 ms.
Enable PCS-Mode port	**On** **Off**	Enables or disables the PCS-Mode port.

PHY Analog Parameters

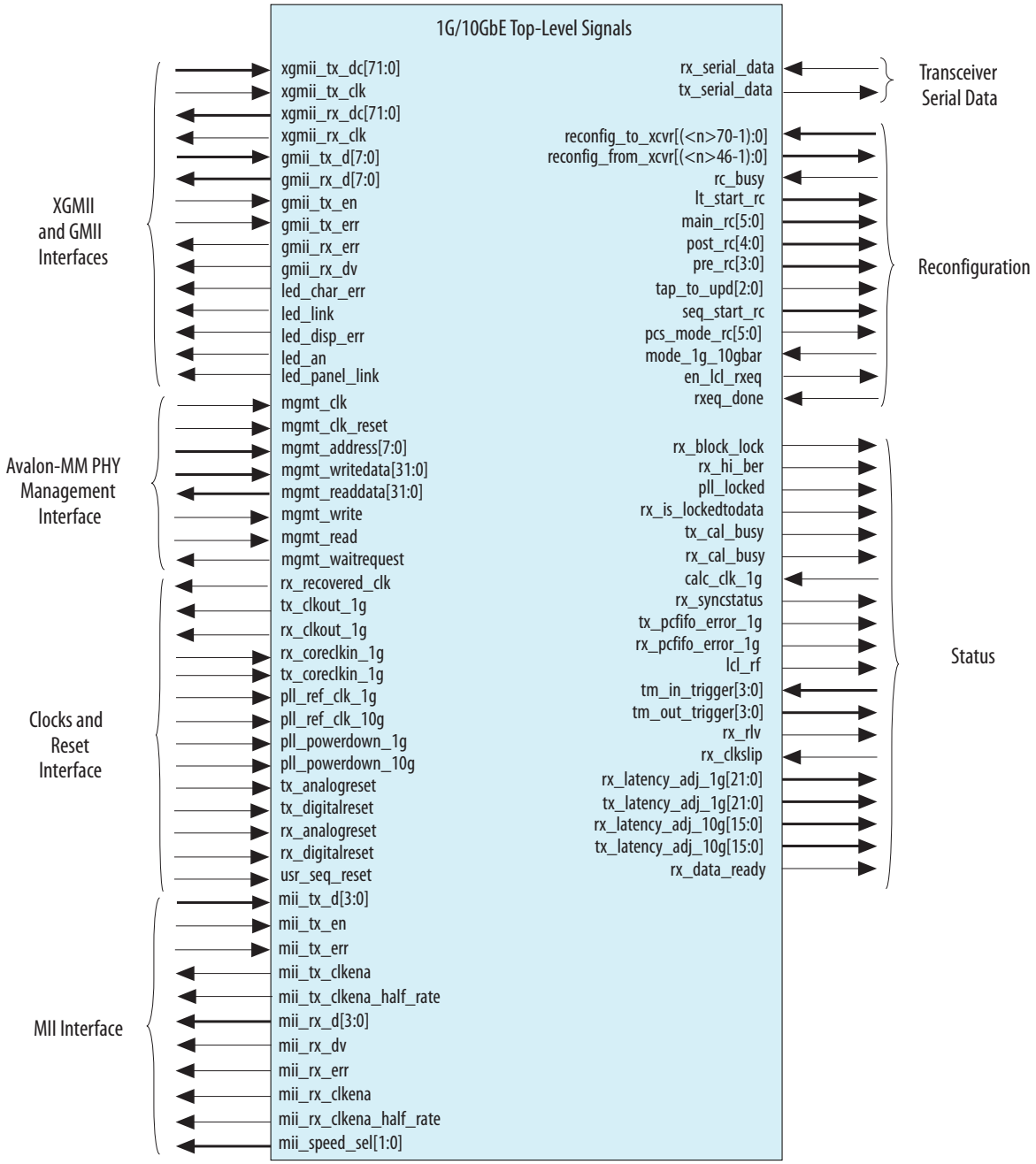
You can specify analog parameters using the Intel Quartus Prime Assignment Editor, the Pin Planner, or the Intel Quartus Prime Settings File (**.qsf**).

Related Information

- **Analog Settings for Arria V GZ Devices** on page 20-11
- **Analog PCB Settings for Stratix V Devices** on page 20-35

1G/10GbE PHY Interfaces

Figure 5-2: 1G/10GbE PHY Top-Level Signals



The block diagram shown in the GUI labels the external pins with the interface type and places the interface name inside the box. The interface type and name are used in the _hw.tcl file. If you turn on **Show signals**, the **block diagram** displays all top-level signal names. For more information about _hw.tcl

files, refer to refer to the *Component Interface Tcl Reference* chapter in volume 1 of the *Intel Quartus Prime Handbook*

Note: Some of the signals shown in are this figure are unused and will be removed in a future release. The descriptions of these identifies them as not functional.

Related Information

Component Interface Tcl Reference

1G/10GbE PHY Clock and Reset Interfaces

This topic illustrates the 1G/10GbE PHY clock and reset connectivity and describes the clock and reset signals.

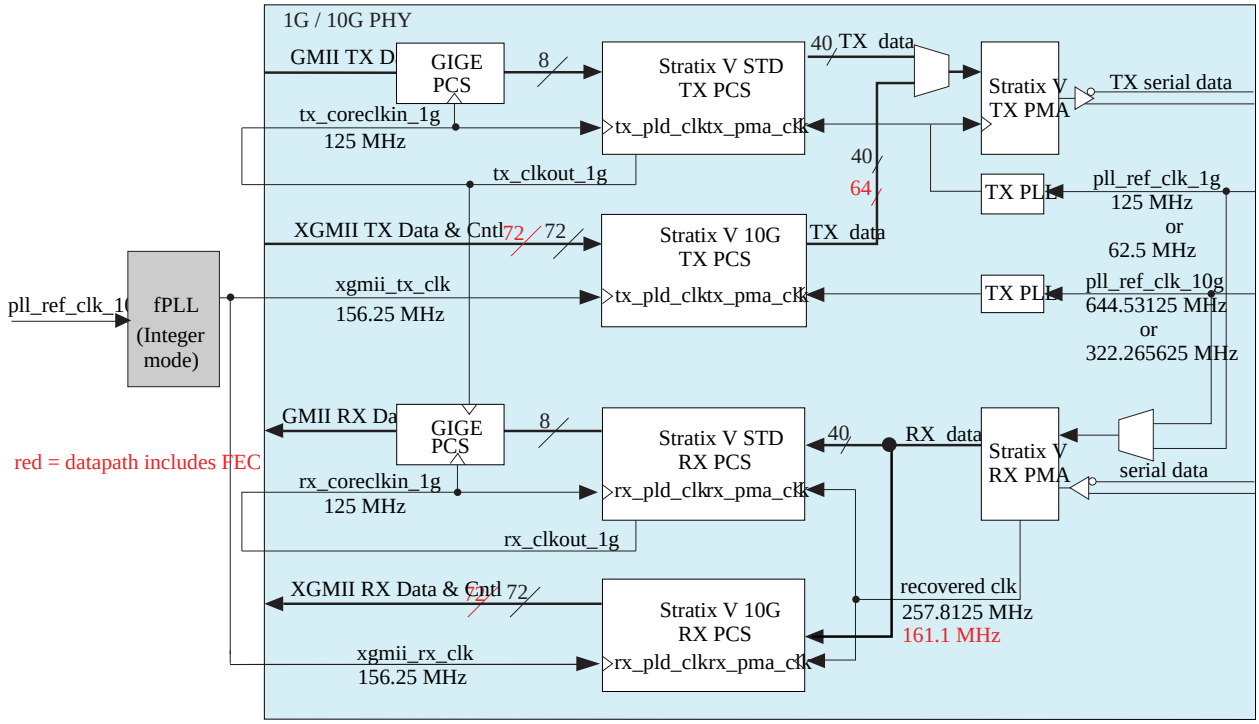
Use the Transceiver PHY Reset Controller IP Core to automatically control the transceiver reset sequence. This reset controller also has manual overrides for the TX and RX analog and digital circuits to allow you to reset individual channels upon reconfiguration.

If you instantiate multiple channels within a transceiver bank they share TX PLLs. If a reset is applied to this PLL, it will affect all channels. Altera recommends leaving the TX PLL free-running after the start-up reset sequence is completed. After a channel is reconfigured you can simply reset the digital portions of that specific channel instead of going through the entire reset sequence. If you are not using the sequencer and the data link is lost, you must assert the `rx_digitalreset` when the link recovers. For more information about reset, refer to the "Transceiver PHY Reset IP Core" chapter in the *Altera Transceiver PHY IP Core User Guide*.

`Phy_mgmt_clk_reset` is the Avalon-MM reset signal. `Phy_mgmt_clk_reset` is also an input to the Transceiver PHY Reset Controller IP Core which is a separately instantiated module not included in the 1G/10GbE and 10GBASE-KR variants. The Transceiver PHY Reset Controller IP Core resets the TX PLL and RX analog circuits and the TX and RX digital circuits. When complete, the Reset Controller asserts the `tx_ready` and `rx_ready` signals.

The following figure provides an overview of the clocking for this IP core.

Figure 5-3: Clocks for Standard and 10G PCS and TX PLLs



The following table describes the clock and reset signals.

Table 5-6: Clock and Reset Signals

Signal Name	Direction	Description
`rx_recovered_clk`	Output	The RX clock which is recovered from the received data. You can use this clock as a reference to lock an external clock source. Its frequency is 125 or 156.25 MHz. For 10G PCS, its frequency is 257.8125 MHz.
`tx_clkout_1g`	Output	GMII TX clock for the 1G TX and RX parallel data source interface. The frequency is 125 MHz.
`rx_clkout_1g`	Output	GMII RX clock for the 1G RX parallel data source interface. The frequency is 125 MHz.
`rx_coreclkin_1g`	Input	Clock to drive the read side of the RX phase compensation FIFO in the Standard PCS. The frequency is 125 MHz.
`tx_coreclkin_1g`	Input	Clock to drive the write side of the TX phase compensation FIFO in the Standard PCS. The frequency is 125 MHz.
`pll_ref_clk_1g`	Input	Reference clock for the PMA block for the 1G mode. Its frequency is 125 or 62.5 MHz.

Signal Name	Direction	Description
pll_ref_clk_10g	Input	Reference clock for the PMA block in 10G mode. Its frequency is 644.53125 or 322.265625 MHz.
pll_powerdown_1g	Input	Resets the 1Gb TX PLLs.
pll_powerdown_10g	Input	Resets the 10Gb TX PLLs.
tx_analogreset	Input	Resets the analog TX portion of the transceiver PHY.
tx_digitalreset	Input	Resets the digital TX portion of the transceiver PHY.
rx_analogreset	Input	Resets the analog RX portion of the transceiver PHY.
rx_digitalreset	Input	Resets the digital RX portion of the transceiver PHY.
usr_seq_rest	Input	Resets the sequencer.

1G/10GbE PHY Data Interfaces

The following table describes the signals in the XGMII and GMII interfaces. The MAC drives the TX XGMII and GMII signals to the 1G/10GbE PHY. The 1G/10GbE PHY drives the RX XGMII or GMII signals to the MAC.

Table 5-7: SGMII and GMII Signals

Signal Name	Direction	Description
1G/10GbE XGMII Data Interface		
xgmii_tx_dc[71:0]	Input	XGMII data and control for 8 lanes. Each lane consists of 8 bits of data and 1 bit of control.
xgmii_tx_clk	Input	Clock for single data rate (SDR) XGMII TX interface to the MAC. It should connect to xgmii_rx_clk. The frequency is 156.25 MHz irrespective of 1588 being enabled or disabled. Driven from the MAC.
xgmii_rx_dc[71:0]	Output	RX XGMII data and control for 8 lanes. Each lane consists of 8 bits of data and 1 bit of control.
xgmii_rx_clk	Input	Clock for SDR XGMII RX interface to the MAC. The frequency is 156.25 MHz irrespective of 1588 being enabled or disabled.
1G/10GbE GMII Data Interface		
gmii_tx_d[7:0]	Input	TX data for 1G mode. Synchronized to tx_clkout_1g clock. The TX PCS 8B/10B module encodes this data which is sent to link partner.

Signal Name	Direction	Description
gmii_rx_d[7:0]	Output	RX data for 1G mode. Synchronized to tx_clkout_1g clock. The RX PCS 8B/10B decoders decodes this data and sends it to the MAC.
gmii_tx_en	Input	When asserted, indicates the start of a new frame. It should remain asserted until the last byte of data on the frame is present on gmii_tx_d.
gmii_tx_err	Input	When asserted, indicates an error. May be asserted at any time during a frame transfer to indicate an error in that frame.
gmii_rx_err	Output	When asserted, indicates an error. May be asserted at any time during a frame transfer to indicate an error in that frame.
gmii_rx_dv	Output	When asserted, indicates the start of a new frame. It remains asserted until the last byte of data on the frame is present on gmii_rx_d.
led_char_err	Output	10-bit character error. Asserted for one rx_clkout_1g cycle when an erroneous 10-bit character is detected.
led_link	Output	When asserted, indicates successful link synchronization at 1Gb. This signal is not used at 10Gb
led_disp_err	Output	Disparity error signal indicating a 10-bit running disparity error. Asserted for one rx_clkout_1g cycle when a disparity error is detected. A running disparity error indicates that more than the previous and perhaps the current received group had an error.
led_an	Output	Clause 37 Auto-negotiation status. The PCS function asserts this signal when auto-negotiation completes.

<table>
<tr><th>Signal Name</th><th>Direction</th><th>Description</th></tr>
<tr><td>led_panel_link</td><td>Output</td><td>When asserted, this signal indicates the following behavior:
<table>
<tr><th>Mode</th><th>Signal Behavior</th></tr>
<tr><td>1000 Base-X without auto-negotiation</td><td>When asserted, indicates successful link synchronization.</td></tr>
<tr><td>1000 Base-X with auto-negotiation</td><td>Clause 37 Auto-negotiation status. The PCS function asserts this signal when auto-negotiation completes.</td></tr>
<tr><td>SGMII mode without auto-negotiation</td><td>When asserted, indicates successful link synchronization.</td></tr>
<tr><td>SGMII mode (MAC) auto-negotiation</td><td>Clause 37 Auto-negotiation status. The PCS function asserts this signal when auto-negotiation completes. Also refer to the description of partner_ability [15].</td></tr>
<tr><td>SGMII mode (PHY) auto-negotiation</td><td>Clause 37 Auto-negotiation status. The PCS function asserts this signal when auto-negotiation completes. Also refer to the description of dev_ability [15].</td></tr>
</table></td></tr>
</table>

XGMII Mapping to Standard SDR XGMII Data

Table 5-8: TX XGMII Mapping to Standard SDR XGMII Interface

The 72-bit TX XGMII data bus format is different than the standard SDR XGMII interface. This table shows the mapping of this non-standard format to the standard SDR XGMII interface.

Signal Name	SDR XGMII Signal Name	Description
xgmii_tx_dc[7:0]	xgmii_sdr_data[7:0]	Lane 0 data
xgmii_tx_dc[8]	xgmii_sdr_ctrl[0]	Lane 0 control
xgmii_tx_dc[16:9]	xgmii_sdr_data[15:8]	Lane 1 data
xgmii_tx_dc[17]	xgmii_sdr_ctrl[1]	Lane 1 control
xgmii_tx_dc[25:18]	xgmii_sdr_data[23:16]	Lane 2 data

Signal Name	SDR XGMII Signal Name	Description
xgmii_tx_dc[26]	xgmii_sdr_ctrl[2]	Lane 2 control
xgmii_tx_dc[34:27]	xgmii_sdr_data[31:24]	Lane 3 data
xgmii_tx_dc[35]	xgmii_sdr_ctrl[3]	Lane 3 control
xgmii_tx_dc[43:36]	xgmii_sdr_data[39:32]	Lane 4 data
xgmii_tx_dc[44]	xgmii_sdr_ctrl[4]	Lane 4 control
xgmii_tx_dc[52:45]	xgmii_sdr_data[47:40]	Lane 5 data
xgmii_tx_dc[53]	xgmii_sdr_ctrl[5]	Lane 5 control
xgmii_tx_dc[61:54]	xgmii_sdr_data[55:48]	Lane 6 data
xgmii_tx_dc[62]	xgmii_sdr_ctrl[6]	Lane 6 control
xgmii_tx_dc[70:63]	xgmii_sdr_data[63:56]	Lane 7 data
xgmii_tx_dc[71]	xgmii_sdr_ctrl[7]	Lane 7 control

Table 5-9: RX XGMII Mapping to Standard SDR XGMII Interface

The 72-bit RX XGMII data bus format is different from the standard SDR XGMII interface. This table shows the mapping of this non-standard format to the standard SDR XGMII interface.

Signal Name	XGMII Signal Name	Description
xgmii_rx_dc[7:0]	xgmii_sdr_data[7:0]	Lane 0 data
xgmii_rx_dc[8]	xgmii_sdr_ctrl[0]	Lane 0 control
xgmii_rx_dc[16:9]	xgmii_sdr_data[15:8]	Lane 1 data
xgmii_rx_dc[17]	xgmii_sdr_ctrl[1]	Lane 1 control
xgmii_rx_dc[25:18]	xgmii_sdr_data[23:16]	Lane 2 data
xgmii_rx_dc[26]	xgmii_sdr_ctrl[2]	Lane 2 control
xgmii_rx_dc[34:27]	xgmii_sdr_data[31:24]	Lane 3 data
xgmii_rx_dc[35]	xgmii_sdr_ctrl[3]	Lane 3 control
xgmii_rx_dc[43:36]	xgmii_sdr_data[39:32]	Lane 4 data
xgmii_rx_dc[44]	xgmii_sdr_ctrl[4]	Lane 4 control
xgmii_rx_dc[52:45]	xgmii_sdr_data[47:40]	Lane 5 data
xgmii_rx_dc[53]	xgmii_sdr_ctrl[5]	Lane 5 control
xgmii_rx_dc[61:54]	xgmii_sdr_data[55:48]	Lane 6 data
xgmii_rx_dc[62]	xgmii_sdr_ctrl[6]	Lane 6 control
xgmii_rx_dc[70:63]	xgmii_sdr_data[63:56]	Lane 7 data
xgmii_rx_dc[71]	xgmii_sdr_ctrl[7]	Lane 7 control

MII Interface Signals

The following signals are available when you turn on the **Expose MII interface** parameter.

Table 5-10: MII Interface Signals

Signal Name	Direction	Description
mii_tx_d[3:0]	Input	TX data to be encoded and sent to link partner.
mii_tx_en	Input	MII transmit control signal.
mii_tx_err	Input	MII transmit error signal.
mii_tx_clkena	Output	Clock enabled signal from PHY to MAC. Following are the effective rates: • For 100 Mbps: 25 MHz • For 10 Mbps: 2.5 MHz
mii_tx_clkena_half_rate	Output	Clock enabled signal from PHY to MAC. Following are the effective rates: • For 100 Mbps: 12.5 MHz • For 10 Mbps: 1.25 MHz
mii_rx_d[3:0]	Output	RX data to be encoded and sent to link partner.
mii_rx_dv	Output	MII receive control signal.
mii_rx_err	Output	MII receive error signal.
mii_rx_clkena	Output	Clock enabled signal from PHY to MAC. Following are the effective rates: • For 100 Mbps: 25 MHz • For 10 Mbps: 2.5 MHz
mii_rx_clkena_half_rate	Output	Clock enabled signal from PHY to MAC. Following are the effective rates: • For 100 Mbps: 12.5 MHz • For 10 Mbps: 1.25 MHz
mii_speed_sel[1:0]	Output	This signal indicates the current speed of the PHY. • 2'b00: 10 Gbps • 2'b01: 1 Gbps • 2'b10: 100 Mbps • 2'b11: 10 Mbps

Serial Data Interface

Table 5-11: Serial Data Signals

Signal Name	Direction	Description
rx_serial_data	Input	RX serial input data
tx_serial_data	Output	TX serial output data

1G/10GbE Control and Status Interfaces

The 10GBASE-KR XGMII and GMII interface signals drive data to and from PHY.

Table 5-12: Control and Status Signals

Signal Name	Direction	Description
rx_block_lock	Output	Asserted to indicate that the block synchronizer has established synchronization at 10G.
rx_hi_ber	Output	Asserted by the BER monitor block to indicate a Sync Header high bit error rate greater than 10^{-4}.
pll_locked	Output	When asserted, indicates the TX PLL is locked.
rx_is_lockedtodata	Output	When asserted, indicates the RX channel is locked to input data.
tx_cal_busy	Output	When asserted, indicates that the initial TX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP. You must hold the channel in reset until calibration completes.
rx_cal_busy	Output	When asserted, indicates that the initial RX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP.
calc_clk_1g	Input	This clock is used for calculating the latency of the soft 1G PCS block. This clock is only required for when you enable 1588 in 1G mode.
rx_sync_status	Output	When asserted, indicates the word aligner has aligned to in incoming word alignment pattern.
tx_pcfifo_error_1g	Output	When asserted, indicates that the Standard PCS TX phase compensation FIFO is full.
rx_pcfifo_error_1g	Output	When asserted, indicates that the Standard PCS RX phase compensation FIFO is full.

Signal Name	Direction	Description
lcl_rf	Input	When asserted, indicates a Remote Fault (RF).The MAC sends this fault signal to its link partner. Bit D13 of the Auto Negotiation Advanced Remote Fault register (0xC2) records this error.
tm_in_trigger[3:0]	Input	This is an optional signal that can be used for hardware testing by using an oscilloscope or logic analyzer to trigger events. If unused, tie this signal to 1'b0.
tm_out_trigger[3:0]	Output	This is an optional signal that can be used for hardware testing by using an oscilloscope or logic analyzer to trigger events. You can ignore this signal if not used.
rx_rlv	Output	When asserted, indicates a run length violation.
rx_clkslip	Input	When you turn this signal on, the deserializer skips one serial bit or the serial clock is paused for one cycle to achieve word alignment. As a result, the period of the parallel clock can be extended by 1 unit interval (UI). This is an optional control input signal.
rx_latency_adj_1g[21:0]	Output	When you enable 1588, this signal outputs the real time latency in GMII clock cycles (125 MHz) for the RX PCS and PMA datapath for 1G mode. Bits 0 to 9 represent fractional number of clock cycles. Bits 10 to 21 represent number of clock cycles.
tx_latency_adj_1g[21:0]	Output	When you enable 1588, this signal outputs real time latency in GMII clock cycles (125 MHz) for the TX PCS and PMA datapath for 1G mode. Bits 0 to 9 represent fractional number of clock cycles. Bits 10 to 21 represent number of clock cycles.
rx_latency_adj_10g[15:0]	Output	When you enable 1588, this signal outputs the real time latency in XGMII clock cycles (156.25 MHz) for the RX PCS and PMA datapath for 10G mode. Bits 0 to 9 represent fractional number of clock cycles. Bits 10 to 15 represent number of clock cycles.
tx_latency_adj_10g[15:0]	Output	When you enable 1588, this signal outputs real time latency in XGMII clock cycles (156.25 MHz) for the TX PCS and PMA datapath for 10G mode. Bits 0 to 9 represent fractional number of clock cycles. Bits 10 to 15 represent number of clock cycles.
rx_data_ready	Output	When asserted, indicates that the MAC can begin sending data to the 10GBASE-KRPHY IP Core.

Register Interface Signals

The Avalon-MM master interface signals provide access to all registers.

Refer to the *Typical Slave Read and Write Transfers* and *Master Transfers* sections in the *Avalon Memory-Mapped Interfaces* chapter of the *Avalon Interface Specifications* for timing diagrams.

Table 5-13: Avalon-MM Interface Signals

Signal Name	Direction	Description
`mgmt_clk`	Input	The clock signal that controls the Avalon-MM PHY management, interface. If you plan to use the same clock for the PHY management interface and transceiver reconfiguration, you must restrict the frequency range to 100-125 MHz to meet the specification for the transceiver reconfiguration clock.
`mgmt_clk_reset`	Input	Resets the PHY management interface. This signal is active high and level sensitive.
`mgmt_addr[7:0]`	Input	8-bit Avalon-MM address.
`mgmt_writedata[31:0]`	Input	Input data.
`mgmt_readdata[31:0]`	Output	Output data.
`mgmt_write`	Input	Write signal. Active high.
`mgmt_read`	Input	Read signal. Active high.
`mgmt_waitrequest`	Output	When asserted, indicates that the Avalon-MM slave interface is unable to respond to a read or write request. When asserted, control signals to the Avalon-MM slave interface must remain constant.

Related Information

Avalon Interface Specifications

1G/10GbE PHY Register Definitions

You can access the 1G/10GbE registers using the Avalon-MM PHY management interface with word addresses and a 32-bit embedded processor. A single address space provides access to all registers.

Notes:

- Unless otherwise indicated, the default value of all registers is 0.
- Writing to reserved or undefined register addresses may have undefined side effects.
- To avoid any unspecified bits to be erroneously overwritten, you must perform read-modify-writes to change the register values.

Table 5-14: 1G/10GbE Register Definitions

Addr	Bit	R/W	Name	Description
0xB0	0	RW	`Reset SEQ`	When set to 1, resets the sequencer. This bit must be used in conjunction with SEQ Force Mode[2:0] . This reset self clears.
	1			Reserved.
	2	RW	`Disable LF Timer`	When set to 1, disables the Link Fault timer. When set to 0, the Link Fault timer is enabled.
	6:4	RW	`SEQ Force Mode[2:0]`	Forces the sequencer to a specific protocol. Allows you to change speeds if you have turned on **Enable automatic speed detection** in the GUI. You must write the Reset SEQ bit to 1 for the Force to take effect. The following encodings are defined: • 3'b000: No force • 3'b001: GigE • 3'b010: Reserved • 3'b011: Reserved • 3'b100: 10GBASE-R • 3'b101: Reserved • Others: Reserved
0xB1	0	RO	`SEQ Link Ready`	When asserted, the sequencer is indicating that the link is ready.

Related Information

Avalon Interface Specifications

PMA Registers

The PMA registers allow you to reset the PMA and provide status information.

Table 5-15: PMA Registers - Reset and Status

The following PMA registers allow you to reset the PMA and provide status information.

Addr	Bit	Access	Name	Description
0x22	0	RO	`pma_tx_pll_is_locked`	Indicates that the TX PLL is locked to the input reference clock.

Addr	Bit	Access	Name	Description
0x44	1	RW	reset_tx_digital	Writing a 1 causes the internal TX digital reset signal to be asserted. You must write a 0 to clear the reset condition.
	2	RW	reset_rx_analog	Writing a 1 causes the internal RX analog reset signal to be asserted. You must write a 0 to clear the reset condition.
	3	RW	reset_rx_digital	Writing a 1 causes the internal RX digital reset signal to be asserted. You must write a 0 to clear the reset condition.
0x61	[31:0]	RW	phy_serial_loopback	Writing a 1 puts the channel in serial loopback mode.
0x64	[31:0]	RW	pma_rx_set_locktodata	When set, programs the RX CDR PLL to lock to the incoming data.
0x65	[31:0]	RW	pma_rx_set_locktoref	When set, programs the RX clock data recovery (CDR) PLL to lock to the reference clock.
0x66	[31:0]	RO	pma_rx_is_lockedtodata	When asserted, indicates that the RX CDR PLL is locked to the RX data, and that the RX CDR has changed from LTR to LTD mode.
0x67	[31:0]	RO	pma_rx_is_lockedtoref	When asserted, indicates that the RX CDR PLL is locked to the reference clock.

Table 5-16: PMA Registers - TX and RX Serial Data Interface

The following PMA registers allow you to customize the TX and RX serial data interface

Address	Bit	R/W	Name	Description
0xA8	0	RW	tx_invpolarity	When set to 1, the TX interface inverts the polarity of the TX data. Inverted TX data is output from the 8B/10B encoder.
	1	RW	rx_invpolarity	When set to 1, the RX channels inverts the polarity of the received data. Inverted RX data is input to the 8B/10B decoder.
	2	RW	rx_bitreversal_enable	When set to 1, enables bit reversal on the RX interface. The RX data is input to the word aligner.
	3	RW	rx_bytereversal_enable	When set, enables byte reversal on the RX interface. The RX data is input to the byte deserializer.
	4	RW	force_electrical_idle	When set to 1, forces the TX outputs to electrical idle.

Address	Bit	R/W	Name	Description
0xA9	0	R	rx_syncstatus	When set to 1, indicates that the word aligner is synchronized to incoming data.
	1	R	rx_patterndetect	When set to 1, indicates the 1G word aligner has detected a comma.
	2	R	rx_rlv	When set to 1, indicates a run length violation.
	3	R	rx_rmfifodatainserted	When set to 1, indicates the rate match FIFO inserted code group.
	4	R	rx_rmfifodatadeleted	When set to 1, indicates that rate match FIFO deleted code group.
	5	R	rx_disperr	When set to 1, indicates an RX 8B/10B disparity error.
	6	R	rx_errdetect	When set to 1, indicates an RX 8B/10B error detected.

PCS Registers

Table 5-17: PCS Registers

These registers provide PCS status information.

Addr	Bit	Access	Name	Description
0x800	31:0	RW	Indirect_addr	Because the PHY implements a single channel, this register must remain at the default value of 0 to specify logical channel 0.
0x881	2	RW	RCLR_ERRBLK_CNT	Error Block Counter clear register. When set to 1, clears the RCLR_ERRBLK_CNT register. When set to 0, normal operation continues.
	3	RW	RCLR_BER_COUNT	BER Counter clear register. When set to 1, clears the RCLR_BER_COUNT register. When set to 0, normal operation continues.
0x882	1	RO	HI_BER	High BER status. When set to 1, the PCS is reporting a high BER. When set to 0, the PCS is not reporting a high BER.
	2	RO	BLOCK_LOCK	Block lock status. When set to 1, the PCS is locked to received blocks. When set to 0, the PCS is not locked to received blocks.
	3	RO	TX_FULL	When set to 1, the TX_FIFO is full.
	4	RO	RX_FULL	When set to 1, the RX_FIFO is full.
	5	RO	RX_SYNC_HEAD_ERROR	When set to 1, indicates an RX synchronization error.
	6	RO	RX_SCRAMBLER_ERROR	When set to 1, indicates an RX scrambler error.
	7	RO	Rx_DATA_READY	When set to 1, indicates the PHY is ready to receive data.

Table 5-18: Pattern Generator Registers

1G/10Gbps Ethernet PHY IP core supports the 10G PCS Pattern Generator.

Offset	Bits	R/W	Name	Description
0x12D	[15:0]	R/W	Seed A for PRP	Bits [15:0] of seed A for the pseudo-random pattern.
0x12E	[15:0]			Bits [31:16] of seed A for the pseudo-random pattern.
0x12F	[15:0]			Bits [47:21] of seed A for the pseudo-random pattern.
0x130	[9:0]			Bits [57:48] of seed A for the pseudo-random pattern.
0x131	[15:0]	R/W	Seed B for PRP	Bits [15:0] of seed B for the pseudo-random pattern.
0x132	[15:0]			Bits [31:16] of seed B for the pseudo-random pattern.
0x133	[15:0]			Bits [47:32] of seed B for the pseudo-random pattern.
0x134	[9:0]			Bits [57:48] of seed B for the pseudo-random pattern.

Offset	Bits	R/W	Name	Description
0x135	[15:12]	R/W	Square Wave Pattern	Specifies the number of consecutive 1s and 0s. The following values are available: 1, 4, 5, 6, 8, and 10.
	[10]	R/W	TX PRBS 7 Enable	Enables the PRBS-7 polynomial in the transmitter.
	[8]	R/W	TX PRBS 23 Enable	Enables the PRBS-23 polynomial in the transmitter.
	[6]	R/W	TX PRBS 9 Enable	Enables the PRBS-9 polynomial in the transmitter.
	[4]	R/W	TX PRBS 31 Enable	Enables the PRBS-31 Polynomial in the transmitter.
	[3]	R/W	TX Test Enable	Enables the pattern generator in the transmitter.
	[1]	R/W	TX Test Pattern Select	Selects between the square wave or pseudo-random pattern generator. The following encodings are defined: • 1'b1: Square wave • 1'b0: Pseudo-random pattern or PRBS
	[0]	R/W	Data Pattern Select	Selects the data pattern for the pseudo-random pattern. The following encodings are defined: • 1'b1: Two Local Faults. Two, 32-bit ordered sets are XORd with the pseudo-random pattern. • 1'b0: All 0's
0x137	[2]	R/W	TX PRBS Clock Enable	Enables the transmitter PRBS clock.
	[1]	R/W	TX Square Wave Clock Enable	Enables the square wave clock.

Offset	Bits	R/W	Name	Description
0x15E	[14]	R/W	RX PRBS 7 Enable	Enables the PRBS-7 polynomial in the receiver.
	[13]	R/W	RX PRBS 23 Enable	Enables the PRBS-23 polynomial in the receiver.
	[12]	R/W	RX PRBS 9 Enable	Enables the PRBS-9 polynomial in the receiver.
	[11]	R/W	RX PRBS 31 Enable	Enables the PRBS-31 polynomial in the receiver.
	[10]	R/W	RX Test Enable	Enables the PRBS pattern verifier in the receiver.
0x164	[10]	R/W	RX PRBS Clock Enable	Enables the receiver PRBS Clock.
0x169	[0]	R/W	RX Test Pattern Select	Selects between a square wave or pseudo-random pattern. The following encodings are defined: • 1'b1: Square wave • 1'b0: Pseudo-random pattern or PRBS

1G/10GbE GMII PCS Registers

This topic describes the GMII PCS registers.

Table 5-19: GMII PCS Registers

Addr	Bit	R/W	Name	Description
0x90	9	RW	RESTART_AUTO_ NEGOTIATION	Set this bit to 1 to restart the Clause 37 Auto-Negotiation sequence. For normal operation, set this bit to 0 which is the default value. This bit is self-clearing.
	12	RW	AUTO_NEGOTIATION_ ENABLE	Set this bit to 1 to enable Clause 37 Auto-Negotiation. The default value is 1.
	15	RW	Reset	Set this bit to 1 to generate a synchronous reset pulse which resets all the PCS state machines, comma detection function, and the 8B/10B encoder and decoder. For normal operation, set this bit to 0. This bit self clears.

Addr	Bit	R/W	Name	Description
0x91	2	R	LINK_STATUS	A value of 1 indicates that a valid link is operating. A value of 0 indicates an invalid link. If link synchronization is lost, this bit is 0.
	3	R	AUTO_NEGOTIATION_ ABILITY	A value of 1 indicates that the PCS function supports Clause 37 Auto-Negotiation.
	5	R	AUTO_NEGOTIATION_ COMPLETE	A value of 1 indicates the following status: • The Auto-Negotiation process is complete. • The Auto-Negotiation control registers are valid.
0x94	5	RW	FD	Full-duplex mode enable for the local device. Set to 1 for full-duplex support.
	6	RW	HD	Half-duplex mode enable for the local device. Set to 1 for half-duplex support. This bit should always be set to 0.
	8:7	RW	PS2,PS1	Pause support for local device. The following encodings are defined for PS1/PS2: • 2'b00: Pause is not supported • 2'b01: Asymmetric pause toward link partner • 2'b10: Symmetric pause • 2'b11: Pause is supported on TX and RX
	13:12	RW	RF2,RF1	Remote fault condition for local device. The following encodings are defined for RF1/RF2: • 2'b00: No error, link is valid (reset condition) • 2'b01: Offline • 2'b10: Failure condition • 2'b11: Auto-negotiation error
	14	RO	ACK	Acknowledge for local device. A value of 1 indicates that the device has received three consecutive matching ability values from its link partner.
	15	RW	NP	Next page. In the device ability register, this bit is always set to 0.
0x94 (SGMII mode)	14	RO	ACK	Local device acknowledge. Value as specified in IEEE 802.3z standard.

Addr	Bit	R/W	Name	Description
0x95	5	R	FD	Full-duplex mode enable for the link partner. This bit should always be 1 because only full duplex is supported.
	6	R	HD	Half-duplex mode enable for the link partner. A value of 1 indicates support for half duplex. This bit should always be 0 because half-duplex mode is not supported.
	8:7	R	PS2,PS1	Specifies pause support for link partner. The following encodings are defined for PS1/PS2: • 2'b00: Pause is not supported • 2'b01: Asymmetric pause toward link partner • 2'b10: Symmetric pause • 2'b11: Pause is supported on TX and RX
	13:12	R	RF2,RF1	Remote fault condition for link partner. The following encodings are defined for RF1/RF2: • 2'b00: No error, link is valid (reset condition) • 2'b01: Offline • 2'b10: Failure condition • 2'b11: Auto-negotiation error
	14	R	ACK	Acknowledge for link partner. A value of 1 indicates that the device has received three consecutive matching ability values from its link partner.
	15	R	NP	Next page. When set to 0, the link partner has a Next Page to send. When set to 1, the link partner does not a Next Page. Next Page is not supported in Auto Negotiation.

Addr	Bit	R/W	Name	Description
0x95 (SGMII mode)	11:10	RO	Speed [1:0]	Link partner interface speed: • 2'b00: copper interface speed is 10 Mbps • 2'b01: copper interface speed is 100 Mbps • 2'b10: copper interface speed is 1 Gigabit • 2'b11: reserved
	12	RO	COPPER_DUPLEX_STATUS	Link partner capability: • 1: copper interface is capable of operating in full-duplex mode • 0: copper interface is capable of operating in half-duplex mode **Note:** The PHY IP Core does not support half duplex operation because it is not supported in SGMII mode of the 1G/10G PHY IP core.
	14	RO	ACK	Link partner acknowledge. Value as specified in IEEE 802.3z standard.
	15	RO	COPPER_LINK_STATUS	Link partner status at 1Gb: • 1'b1: copper interface link is up • 1'b0: copper interface link is down
0x96	0	R	LINK_PARTNER_AUTO_NEGOTIATION_ABLE	Setting to 1, indicates that the link partner supports auto negotiation. The default value is 0.
	1	R	PAGE_RECEIVE	A value of 1 indicates that a new page has been received with new partner ability available in the register partner ability. The default value is 0 when the system management agent performs a read access.

Addr	Bit	R/W	Name	Description
0xA4	0	RW	SGMII_ENA	Determines the PCS function operating mode. Setting this bit to 1 enables SGMII mode. Setting this bit to 0 enables 1000BASE-X Gigabit mode.
	1	RW	USE_SGMII_AN	In SGMII mode, setting this bit to 1 configures the PCS with the link partner abilities advertised during auto-negotiation. If this bit is set to 0, then PCS should be configured with the SGMII_SPEED and SGMII_DUPLEX bits.
	3:2	RW	SGMII_SPEED[1:0]	When the PCS operates in SGMII mode (SGMII_ENA = 1) and is not programmed for automatic configuration (USE_SGMII_AN = 0) , the following encodings specify the speed: • 2'b00: 10 Mbps • 2'b01: 100 Mbps • 2'b10: 1Gigabit • 2'b11: Reserved These bits are only valid if you enable the SGMII mode and not the auto-negotiation mode.
	4	RW	SGMII_DUPLEX	Setting this bit to 1 enables half duplex mode for 10/100 Mbps speed. This bit is only valid if you enable the SGMII mode and not the auto-negotiation mode. **Note:** The PHY IP Core does not support half duplex operation because it is not supported in SGMII mode of the 1G/10G PHY IP core.

GIGE PMA Registers

The PMA registers allow you to customize the TX and RX serial data interface.

Table 5-20: GIGE PMA Registers

Address	Bit	R/W	Name	Description
0xA8	0	RW	tx_invpolarity	When set to 1, the TX interface inverts the polarity of the TX data. Inverted TX data is input to the 8B/10B encoder.
	1	RW	rx_invpolarity	When set to 1, the RX channels inverts the polarity of the received data. Inverted RX data is input to the 8B/10B decoder.
	2	RW	rx_bitreversal_enable	When set to 1, enables bit reversal on the RX interface. The RX data is input to the word aligner.
	3	RW	rx_bytereversal_enable	When set, enables byte reversal on the RX interface. The RX data is input to the byte deserializer.
	4	RW	force_electrical_idle	When set to 1, forces the TX outputs to electrical idle.
0xA9	0	R	rx_syncstatus	When set to 1, indicates that the word aligner is synchronized to incoming data.
	1	R	rx_patterndetect	When set to 1, indicates the 1G word aligner has detected a comma.
	2	R	rx_rlv	When set to 1, indicates a run length violation.
	3	R	rx_rmfifodatainserted	When set to 1, indicates the rate match FIFO inserted code group.
	4	R	rx_rmfifodatadeleted	When set to 1, indicates that rate match FIFO deleted code group.
	5	R	rx_disperr	When set to 1, indicates an RX 8B/10B disparity error.
	6	R	rx_errdetect	When set to 1, indicates an RX 8B/10B error detected.

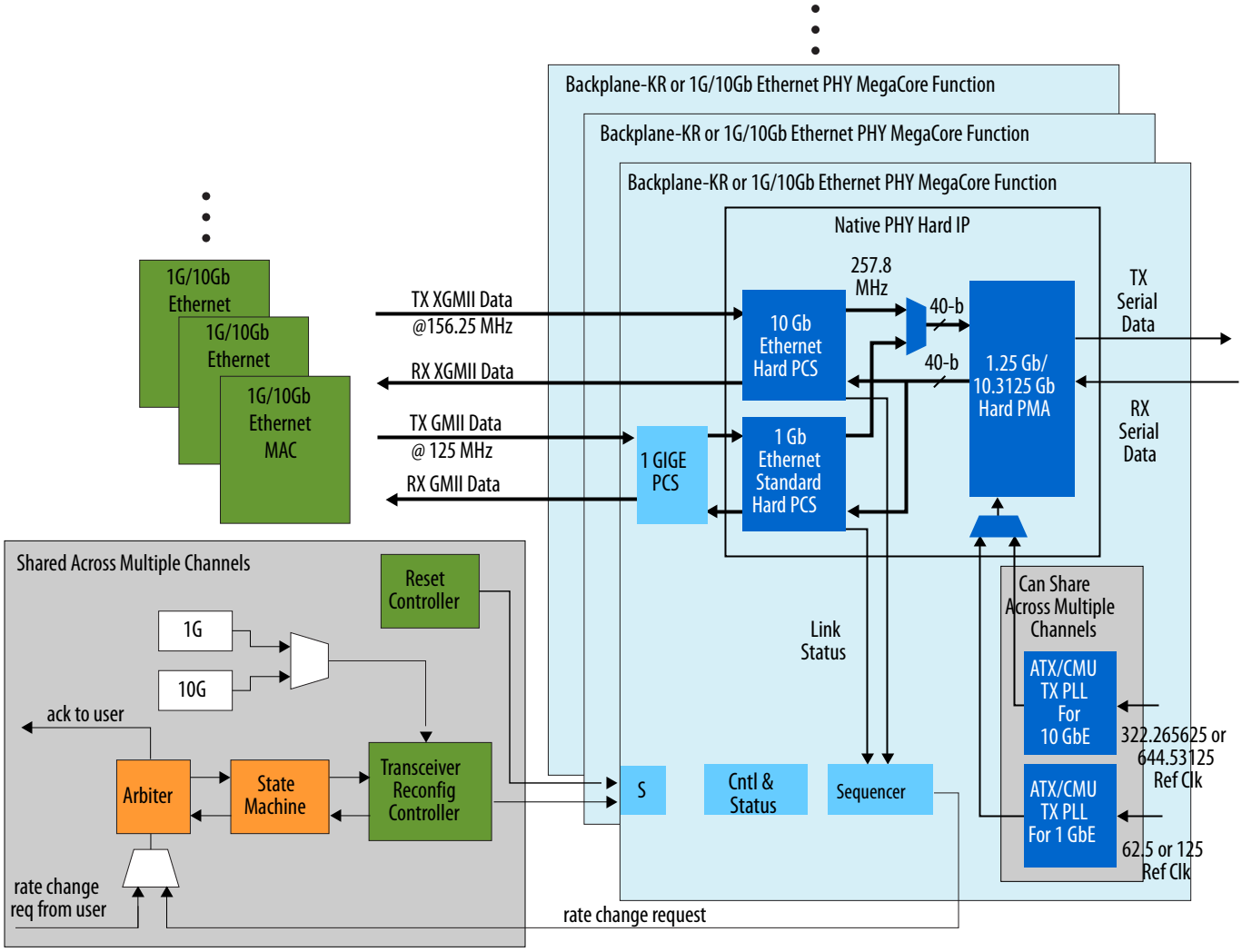
1G/10GbE Dynamic Reconfiguration from 1G to 10GbE

This topic illustrates the necessary logic to reconfigure between the 1G and 10G data rates.

The following figure illustrates the necessary modules to create a design that can dynamically change between 1G and 10GbE operation on a channel-by-channel basis. In this figure, the colors have the following meanings:

- Green-Altera- Cores available Intel Quartus Prime IP Library, including the 1G/10Gb Ethernet MAC, the Reset Controller, and Transceiver Reconfiguration Controller.
- Orange-Arbitration Logic Requirements Logic you must design, including the Arbiter and State Machine. Refer to**1G/10GbE PHY Arbitration Logic Requirements** on page 5-29 and **1G/10GbE PHY State Machine Logic Requirements** on page 5-30 for a description of this logic.
- White-1G and 10G settings files that you must generate. Refer to **Creating a 1G/10GbE Design** on page 5-31 for more information.
- Blue-The 1G/10GbE PHY IP core available in the Intel Quartus Prime IP Library.

Figure 5-4: Block Diagram for Reconfiguration Example



1G/10GbE PHY Arbitration Logic Requirements

This topic describes the arbitration functionality that you must implement.

The arbiter should implement the following logic. You can modify this logic based on your system requirements:

1. Accept requests from the sequencer (if **Enable automatic speed detection** is turned **On** in the GUI) . Prioritize requests to meet system requirements. Requests should consist of the following two buses:

- Channel number—specifies the requested channel
- Mode—specifies 1G or 10G mode for the corresponding channel

2. Select a channel for reconfiguration and send an ack/busy signal to the requestor. The requestor should deassert its request signal when the ack/busy is received.
3. Pass the selected channel and rate information to the state machine for processing.
4. Wait for a done signal from the state machine indicating that the reconfiguration process is complete and it is ready to service another request.

1G/10GbE PHY State Machine Logic Requirements

The state machine should implement the following logic. You can modify this logic based on your system requirements:

1. Wait for `reconfig_busy` from the Transceiver Reconfiguration Controller to be deasserted and the `tx_ready` and `rx_ready` signals from the Transceiver PHY Reset Controller to be asserted. These conditions indicate that the system is ready to service a reconfiguration request.
2. Set the appropriate channel for reconfiguration.
3. Initiate the MIF streaming process. The state machine should also select the appropriate MIF (stored in the ROMs) to stream based on the requested mode.
4. Wait for the `reconfig_busy` signal from the Transceiver Reconfiguration Controller to assert and then deassert indicating the reconfiguration process is complete.
5. Toggle the digital resets for the reconfigured channel and wait for the link to be ready.
6. Deassert the `ack/busy` signal for the selected channel. Deassertion of `ack/busy` indicates to the arbiter that the reconfiguration process is complete and the system is ready to service another request.

Editing a 1G/10GbE MIF File

This topic shows how to edit a 1G/10GbE MIF file to change between 1G and 10Gb Ethernet.

The MIF format contains all bit settings for the transceiver PMA and PCS. Because the 1G/10GbE PHY IP Core only requires PCS reconfiguration for a rate change, the PMA settings should not change. Removing the PMA settings from the MIF file also prevents an unintended overwrite of PMA parameters set through other assignments. A few simple edits to the MIF file removes the PMA settings. Complete the following steps to edit the MIF file:

1. Replace line 17 with `"13: 0001000000010110; -- PMA - RX changed to removed CTLE"`.
2. Replace line 20 with `"16: 0010100000011001; -- PMA - RX continued"`.
3. Replace line 4 with `"4: 0001000000000000; -- PMA - TX"`.
4. Remove lines 7-10. These lines contain the TX settings (V_{OD}, post-tap, pre-tap).
5. Renumber the lines starting with the old line 11.
6. Change the depth at the top of the file from 168 to 164.

Example 5-1: Edits to a MIF to Remove PMA Settings

```
MIF_A
WIDTH=16;
DEPTH=168;
ADDRESS_RADIX=UNS
DATA_RADIX=BIN;

0: 0000000000100001; -- Start of MIF opcode - FAMILY - Stratix V
1: 0000000000000010; -- Type of MIF opcode
2: 0000000000000011; -- RefClk switch opcode
3: 0000000000000100; -- CGB PLL switch opcode
4: 0011000000000000; -- PMA - TX
5: 0010100001100000;
6: 0000001111100000;
7: 1000000111010100;
8: 0000000000000000;
9: 0000001111000000;
10: 0100000000110010;
11: 0010100000001100; -- PMA - RX (PLL section
12: 0110000000000000;
13: 0000000000000000;
14: 0000110101100000;
15: 0000000000000010;
16: 0100000000000000;
17: 0100000000010110;-- PMA - RX
18: 0000111010010110;
19: 0100000000010001;
20: 0000000000000000;
21: 0000000000000000;
22: 0010110000000000;
23: 0000000110000001;
24: 0100000100001111;
25: 0001000101110011;
26: 0000100000110000; -- PMA -- COM
```

```
MIF_B;
WIDTH=16;
DEPTH=164;
ADDRESS_RADIX=UNS
DATA_RADIX=BIN;

0: 0000000000100001; -- Start of MIF opcode - FAMILY- Stratix V
1: 0000000000000010; -- Type of MIF opcode
2: 0000000000000011; -- RefClk switch opcode
3: 0000000000000100; -- CGB PLL switch opcode
4: 0010000000000000; -- PMA - TX
5: 0010100001100000;
6: 0000001111100000;

7: 0010100000001100; -- PMA - RX (PLL section)
8: 0110000000000000;
9: 0000000000000000;
10: 0000110101100000;
11: 0000000000000010;
12: 0100000000000000;
13: 0001000000010110; -- PMA - RX removed CTLE
14: 0000111010010110;
15: 0100000000010001;
16: 0010100000011001; -- PMA - RX continued
17: 0000000000000000;
18: 0010110000000000;
19: 0000000110000001;
20: 0100000100001111;
21: 0001000101110011;
22: 0000100000110000; -- PMA -- COM
```

Creating a 1G/10GbE Design

Here are the steps you must take to create a 1G/10GbE design using this PHY.

1. Generate the 1G/10GbE PHY with the required parameterization.
2. Generate a Transceiver Reconfiguration Controller with the correct number of reconfiguration interfaces based on the number of channels you are using. This controller is connected to all the transceiver channels. It implements the reconfiguration process.
3. Generate a Transceiver Reset Controller.
4. Create arbitration logic that prioritizes simultaneous reconfiguration requests from multiple channels. This logic should also acknowledge the channel being serviced causing the requestor to deassert its request signal.
5. Create a state machine that controls the reconfiguration process. The state machine should:
   a. Receive the prioritized reconfiguration request from the arbiter
   b. Put the Transceiver Reconfiguration Controller into MIF streaming mode.
   c. Select the correct MIF and stream it into the appropriate channel.
   d. Wait for the reconfiguration process to end and provide status signal to arbiter.
6. Generate one ROM for each required configuration.
7. Create a MIF for each configuration and associate each MIF with a ROM created in Step 6. For example, create a MIF for 1G with 1588 and a MIF for 10G with 1588. These MIFs are the two configurations used in the MIF streaming process.

8. Generate a fractional PLL to create the 156.25 MHz XGMII clock from the 10G reference clock.
9. Instantiate the PHY in your design based on the required number of channels.
10. To complete the system, connect all the blocks.

Dynamic Reconfiguration Interface Signals

You can use the dynamic reconfiguration interface signals to dynamically change between 1G,10G data rates and AN or LT mode. These signals also used to update TX coefficients during Link Training..

Table 5-21: Dynamic Reconfiguration Interface Signals

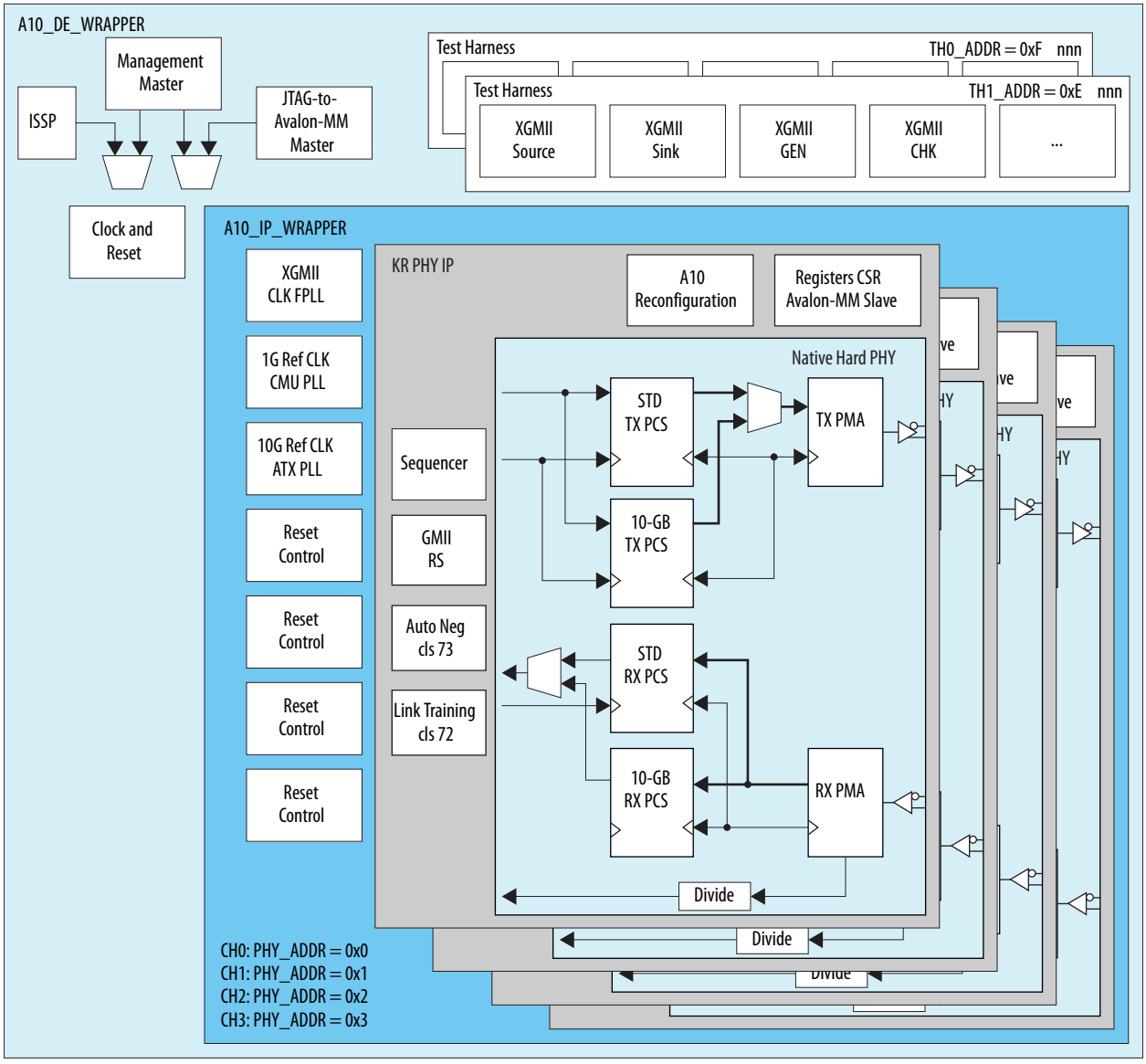
Signal Name	Direction	Description
`reconfig_to_xcvr [(<n>70-1):0]`	Input	Reconfiguration signals from the Reconfiguration Design Example. <*n*> grows linearly with the number of reconfiguration interfaces.
`reconfig_from_xcvr [(<n>46-1):0]`	Output	Reconfiguration signals to the Reconfiguration Design Example. <*n*> grows linearly with the number of reconfiguration interfaces.
`rc_busy`	Input	When asserted, indicates that reconfiguration is in progress.
`lt_start_rc`	Output	When asserted, starts the TX PMA equalization reconfiguration.
`main_rc[5:0]`	Output	The main TX equalization tap value which is the same as V_{OD}. The following example mappings to the V_{OD} settings are defined: • 6'b111111: FIR_MAIN_12P6MA • 6'b111110: FIR_MAIN_12P4MA • 6'b000001: FIR_MAIN_P2MA • 6'b000000: FIR_MAIN_DISABLED
`post_rc[4:0]`	Output	The post-cursor TX equalization tap value. This signal translates to the first post-tap settings. The following example mappings are defined: • 5'b11111: FIR_1PT_6P2MA • 5'b11110: FIR_1PT_6P0MA • 5'b00001: FIR_1PT_P2MA • 5'b00000: FIR_1PT_DISABLED
`pre_rc[3:0]`	Output	The pre-cursor TX equalization tap value. This signal translates to pre-tap settings. The following example mappings are defined: • 4'b1111: FIR_PRE_3P0MA • 4'b1110: FIR_PRE_P28MA • 4'b0001: FIR_PRE_P2MA • 4'b0000: FIR_PRE_DISABLED

Signal Name	Direction	Description
tap_to_upd[2:0]	Output	Specifies the TX equalization tap to update to optimize signal quality. The following encodings are defined: • 3'b100: main tap • 3'b010: post-tap • 3'b001: pre-tap
seq_start_rc	Output	When asserted, starts PCS reconfiguration.
pcs_mode_rc[5:0]	Output	Specifies the PCS mode for reconfig using 1-hot encoding. The following modes are defined: • 6'b000001: Auto-Negotiation mode • 6'b000010: Link Training mode • 6'b000100: 10GBASE-KR data mode • 6'b001000: GigE data mode • 6'b010000: Reserved • 6'b100000:10G data mode with FEC
dfe_start_rc	Output	When asserted, starts the RX DFE equalization of the PMA.
dfe_mode[1:0]	Output	Specifies the DFE operation mode. Valid at the rising edge of the def_start_rc signal and held until the falling edge of the rc_busy signal. The following encodings are defined: • 2'b00: Disable DFE • 2'b01: DFE triggered mode • 2'b10: Reserved • def_start_rcd'b11: Reserved
ctle_start_rc	Output	When asserted, starts continuous time-linear equalization (CTLE) reconfiguration.
ctle_mode[1:0]	Output	Specifies CTLE mode. These signals are valid at the rising edge of the ctle_start_rc signal and held until the falling edge of the rc_busy signal. The following encodings are defined: • 2'b00: ctle_rc[3:0] drives the value of CTLE set during link training • 2'b01: Reserved • 2b'10: Reserved • 2'b11: Reserved
ctle_rc[3:0]	Output	RX CTLE value. This signal is valid at the rising edge of the ctle_start_rc signal and held until the falling edge of the rc_busy signal. The valid range of values is 4'b0000-4'b1111.

Signal Name	Direction	Description
mode_1g_10gbar	Input	This signal indicates the requested mode for the channel. A 1 indicates 1G mode and a 0 indicates 10G mode. This signal is only used when the sequencer which performs automatic speed detection is disabled.
en_lcl_rxeq	Output	This signal is not used. You can leave this unconnected.
rxeq_done	Input	Link training requires RX equalization to be complete. Tie this signal to 1 to indicate that RX equalization is complete.

Design Example

Figure 5-5: PHY-Only Design Example with Two Backplane Ethernet and Two Line-Side (1G/10G) Ethernet Channels



Related Information

- **Arria 10 Transceiver PHY Design Examples**
- **10-Gbps Ethernet MAC IP Function User Guide.**
  For more information about latency in the MAC as part of the Precision Time Protocol implementation.

Simulation Support

The 1G/10GbE and 10GBASE-KR PHY IP core supports the following Intel-supported simulators for this Quartus Prime software release:

- ModelSim* Verilog
- ModelSim VHDL
- VCS Verilog
- VCS VHDL
- NCSIM Verilog
- NCSIM VHDL simulation

When you generate a 1G/10GbE or 10GBASE-KR PHY IP core, the Quartus Prime software optionally generates an IP functional simulation model.

TimeQuest Timing Constraints

To pass timing analysis, you must decouple the clocks in different time domains. The necessary Synopsys Design Constraints File (**.sdc**) timing constraints are included in the top-level wrapper file.

Acronyms

This table defines some commonly used Ethernet acronyms.

Table 5-22: Ethernet Acronyms

Acronym	Definition
AN	Auto-Negotiation in Ethernet as described in Clause 73 of IEEE 802.3ap-2007.
BER	Bit Error Rate.
DME	Differential Manchester Encoding.
FEC	Forward error correction.
GMII	Gigabit Media Independent Interface.
KR	Short hand notation for Backplane Ethernet with 64b/66b encoding.
LD	Local Device.
LT	Link training in backplane Ethernet Clause 72 for 10GBASE-KR and 40GBASE-KR4.
LP	Link partner, to which the LD is connected.
MAC	Media Access Control.
MII	Media independent interface.
OSI	Open System Interconnection.
PCS	Physical Coding Sublayer.

Acronym	Definition
PHY	Physical Layer in OSI 7-layer architecture, also in Intel device scope is: PCS + PMA.
PMA	Physical Medium Attachment.
PMD	Physical Medium Dependent.
SGMII	Serial Gigabit Media Independent Interface.
WAN	Wide Area Network.
XAUI	10 Gigabit Attachment Unit Interface.

1G/2.5G/5G/10G Multi-rate Ethernet PHY IP Core 6

2020.06.02

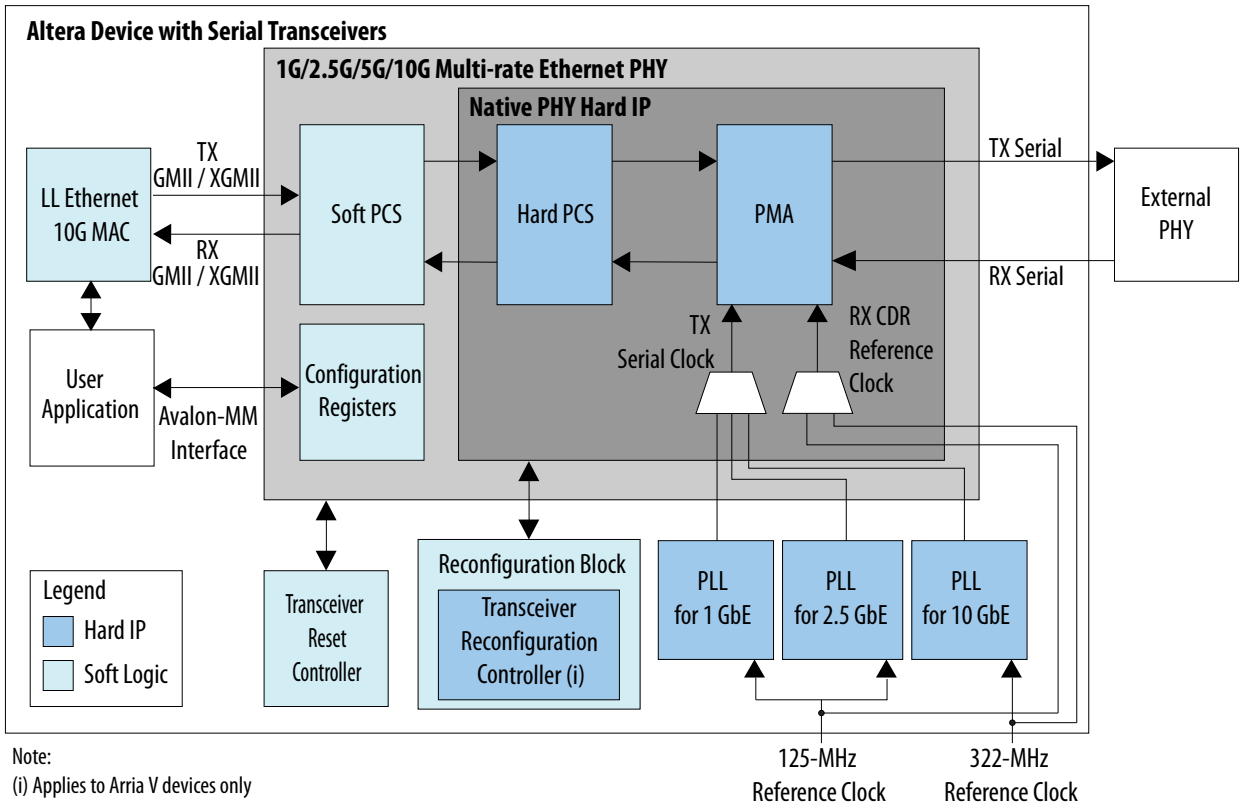
UG-01080 Subscribe Send Feedback

About the 1G/2.5G/5G/10G Multi-rate Ethernet PHY Intel FPGA IP Core

The 1G/2.5G/5G/10G Multi-rate Ethernet PHY Intel FPGA IP core implements the Ethernet protocol as defined in Clause 36 of the *IEEE 802.3 2005 Standard*. The PHY IP core consists of a physical coding sublayer (PCS) function and an embedded physical media attachment (PMA). You can dynamically switch the PHY operating speed.

Note: Intel FPGAs implement and support the required Media Access Control (MAC) and PHY (PCS +PMA) IP to interface in a chip-to-chip or chip-to-module channel with external MGBASE-T and NBASE-T PHY standard devices. You are required to use an external PHY device to drive any copper media.

Figure 6-1: Block Diagram of the PHY IP Core



Features

Table 6-1: PHY Features

Feature	Description
Multiple operating speeds	1G, 2.5G, 5G, and 10G.
MAC-side interface	16-bit GMII for 1G and 2.5G.
	32-bit XGMII for 1G/2.5G/5G/10G (USXGMII).
	64-bit XGMII for 10G.
Network-side interface	1.25 Gbps for 1G.
	3.125 Gbps for 2.5G.
	10.3125 Gbps for 1G/2.5G/5G/10G (USXGMII).
Avalon Memory-Mapped interface	Provides access to the configuration registers of the PHY.
PCS function	1000BASE-X for 1G and 2.5G.
	10GBASE-R for 10G.
	USXGMII PCS for 1G/2.5G/5G/10G

Feature	Description
Auto-negotiation	Implements clause 37. Supported in 1GbE only. USXGMII Auto-negotiation supported in the 1G/2.5G/5G/10G (USXGMII) configuration.
IEEE 1588v2	Provides the required latency to the MAC if the MAC enables the IEEE 1588v2 feature.
Sync-E	Provides the clock for Sync-E implementation.

Release Information

Table 6-2: PHY Release Information

Item	Description
Version	16.0
Release Date	May 2016
Ordering Codes	IP-10GMRPHY
Product ID	00E4
Vendor ID	6AF7
Open Core Plus	Supported

Device Family Support

Device Family	Operating Mode	Support Level
Arria® V GX/GT/SX/ST	2.5G 1G/2.5G	Final
Other device families		No support

Resource Utilization

The following estimates are obtained by compiling the PHY IP core with the Intel Quartus Prime software.

Table 6-3: Resource Utilization

Device	Speed	ALMs	ALUTs	Logic Registers	Memory Block
Arria V	1G/2.5G	550	750	1200	2 (M10K)
	1G/2.5G with IEEE 1588v2 enabled	1200	1850	2550	2 (M10K)

Using the IP Core

The Intel FPGA IP Library is installed as part of the Intel Quartus Prime installation process. You can select the 1G/2.5G/5G/10G Multi-rate Ethernet IP core from the library and parameterize it using the IP parameter editor.

Parameter Settings

You customize the PHY IP core by specifying the parameters in the parameter editor in the Intel Quartus Prime software. The parameter editor enables only the parameters that are applicable to the selected speed.

Table 6-4: Multi-rate Ethernet PHY IP Core Parameters

Name	Value	Description
Speed	2.5G 1G/2.5G	The operating speed of the PHY.
Enable IEEE 1588 Precision Time Protocol	On, Off	Select this parameter for the PHY to provide latency information to the MAC. The MAC requires this information if it enables the IEEE 1588v2 feature. This parameter is enabled only for 2.5G and 1G/2.5G.
PHY ID (32 bit)	32-bit value	An optional 32-bit unique identifier: • Bits 3 to 24 of the Organizationally Unique Identifier (OUI) assigned by the IEEE • 6-bit model number • 4-bit revision number If unused, do not change the default value, which is 0x00000000.
Reference clock frequency for 10 GbE (MHz)	322.265625, 644.53125	Specify the frequency of the reference clock for 10GbE.
Selected clock network for 1GbE	`x1`, `xN`	Select the clock network for the 1GbE TX PLL. This parameter applies to Arria V devices only.
Selected clock network for 2.5GbE	`x1`, `xN`	Select the clock network for the 2.5GbE TX PLL. This parameter applies to Arria V devices only.

Timing Constraints

Constrain the PHY based on the fastest speed. For example, if you configure the PHY as 1G/2.5G, constrain it based on 2.5G.

Changing the PHY's Speed

You can change the PHY's speed through the reconfiguration block.

1. The user application initiates the speed change by writing to the corresponding register of the reconfiguration block.
2. The reconfiguration block performs the following steps:
   - In Arria V devices:
     1. Sets the `xcvr_mode` signal of the 1G/2.5/10G Multi-rate Ethernet PHY to the requested speed.
     2. Selects the corresponding transceiver PLL.
     3. Configures the transceiver using the configuration settings embedded in the reconfiguration block.
3. The reconfiguration block triggers the PHY reset through the transceiver reset controller.

Configuration Registers

Register Map

You can access the 16-bit/32-bit configuration registers via the Avalon memory-mapped interface.

Table 6-5: Register Map

Address Range	Usage	Bit	Configuration
0x00 : 0x1F	1000BASE-X/SGMII	16	2.5G, 1G/2.5G, 1G/2.5G/10G (MGBASE-T)
0x400 : 0x41F	USXGMII	32	10M/100M/1G/2.5G/5G/10G (USXGMII)
0x461	Serial Loopback	32	10M/100M/1G/2.5G/5G/10G (USXGMII)

Configuration Registers

You can access the 16-bit configuration registers via the Avalon memory-mapped interface. These configuration registers apply only to 2.5G and 1G/2.5G operating modes.

Observe the following guidelines when accessing the registers:

- Do not write to reserved or undefined registers.
- When writing to the registers, perform read-modify-write to ensure that reserved or undefined register bits are not overwritten.

Table 6-6: PHY Register Definitions

Addr	Name	Description	Access	HW Reset Value
0x00	control	• Bit [15]: RESET. Set this bit to 1 to trigger a soft reset. The PHY clears the bit when the reset is completed. The register values remain intact during the reset.	RWC	0
		• Bit[14]: LOOPBACK. Set this bit to 1 to enable loopback on the serial interface.	RW	0
		• Bit [12]: AUTO_NEGOTIATION_ENABLE. Set this bit to 1 to enable auto-negotiation. Auto-negotiation is supported only in 1GbE. Therefore, set this bit to 0 when you switch to a speed other than 1GbE.	RW	0
		• Bit [9]: RESTART_AUTO_NEGOTIATION. Set this bit to 1 to restart auto-negotiation. The PHY clears the bit as soon as auto-negotiation is restarted.	RWC	0
		• The rest of the bits are reserved.	—	—
0x01	status	• Bit [5]: AUTO_NEGOTIATION_COMPLETE. A value of "1" indicates that the auto-negotiation is completed.	RO	0
		• Bit [3]: AUTO_NEGOTIATION_ABILITY. A value of "1" indicates that the PCS function supports auto-negotiation.	RO	1
		• Bit [2]: LINK_STATUS. A value of "0" indicates that the link is lost. Value of 1 indicates that the link is established.	RO	0
		• The rest of the bits are reserved.	—	—
0x02:0x03	phy_identifier	The value set in the **PHY_IDENTIFIER** parameter.	RO	Value of PHY_IDENTIFIER parameter

Addr	Name	Description	Access	HW Reset Value
0x04	dev_ability	Use this register to advertise the device abilities during auto-negotiation.	—	—

		• Bits [13:12]: RF. Specify the remote fault.
		  • 00: No error.
		  • 01: Link failure.
		  • 10: Offline.
		  • 11: Auto-negotiation error.
			RW	00

		• Bits [8:7]: PS. Specify the PAUSE support.
		  • 00: No PAUSE.
		  • 01: Symmetric PAUSE.
		  • 10: Asymmetric PAUSE towards the link partner.
		  • 11: Asymmetric and symmetric PAUSE towards the link device.
			RW	11

		• Bit [5]: FD. Ensure that this bit is always set to 1.	RW	1
		• The rest of the bits are reserved.	—	—

Addr	Name	Description	Access	HW Reset Value
0x05	partner_ability	The device abilities of the link partner during auto-negotiation.	—	—
		• Bit [14]: ACK. A value of "1" indicates that the link partner has received three consecutive matching ability values from the device.	RO	0

		• Bits [13:12]: RF. The remote fault.
		  • 00: No error.
		  • 01: Link failure.
		  • 10: Offline.
		  • 11: Auto-negotiation error.
			RO	0

		• Bits [8:7]: PS. The PAUSE support.
		  • 00: No PAUSE.
		  • 01: Symmetric PAUSE.
		  • 10: Asymmetric PAUSE towards the link partner.
		  • 11: Asymmetric and symmetric PAUSE towards the link device.
			RO	0

		• Bit [6]: HD. A value of "1" indicates that half-duplex is supported.	RO	0
		• Bit [5]: FD. A value of "1" indicates that full-duplex is supported.	RO	0
		• The rest of the bits are reserved.	—	—
0x06	an_expansion	The PCS capabilities and auto-negotiation status.	—	—
		Bit [1]: PAGE_RECEIVE. A value of "1" indicates that the partner_ability register has been updated. This bit is automatically cleared once it is read.	RO	0
		Bit [0]: LINK_PARTNER_AUTO_NEGOTIATION_ABLE. A value of "1" indicates that the link partner supports auto-negotiation.	RO	0
0x07	device_next_page	The PHY does not support the next page feature. These registers are always set to 0.	RO	0
0x08	partner_next_page		RO	0
0x09:0x0F	Reserved	—	—	—

Addr	Name	Description	Access	HW Reset Value
0x10	scratch	Provides a memory location to test read and write operations.	RW	0
0x11	rev	The current version of the PHY IP core.	RO	Current version of the PHY
0x12:0x13	link_timer	21-bit auto-negotiation link timer. • Offset 0x12: link_timer[15:0]. Bits [8:0] are always be set to 0. • Offset 0x13: link_timer[20:16] occupies the lower 5 bits. The remaining 11 bits are reserved and must always be set to 0.	RW	0
0x14:0x1F	Reserved	—	—	—

Addr	Name	Description	Access	HW Reset Value
0x400	usxgmii_control	Control Register	—	—
		Bit [0]: USXGMII_ENA: • 0: 10GBASE-R mode • 1: USXGMII mode	RW	0x0
		Bit [1]: USXGMII_AN_ENA is used when USXGMII_ENA is set to 1: • 0: Disables USXGMII Auto-Negotiation and manually configures the operating speed with the USXGMII_SPEED register. • 1: Enables USXGMII Auto-Negotiation, and automatically configures operating speed with link partner ability advertised during USXGMII Auto-Negotiation.	RW	0x1
		Bit [4:2]: USXGMII_SPEED is the operating speed of the PHY in USXGMII mode and USE_USXGMII_AN is set to 0. • 3'b000: Reserved • 3'b001: Reserved • 3'b010: 1G • 3'b011: 10G • 3'b100: 2.5G • 3'b101: 5G • 3'b110: Reserved • 3'b111: Reserved	RW	0x0
		Bit [8:5]: Reserved	—	—
		Bit [9]: RESTART_AUTO_NEGOTIATION Write 1 to restart Auto-Negotiation sequence The bit is cleared by hardware when Auto-Negotiation is restarted.	RWC (hardware self-clear)	0x0
		Bit [15:10]: Reserved	—	—
		Bit [30:16]: Reserved	—	—

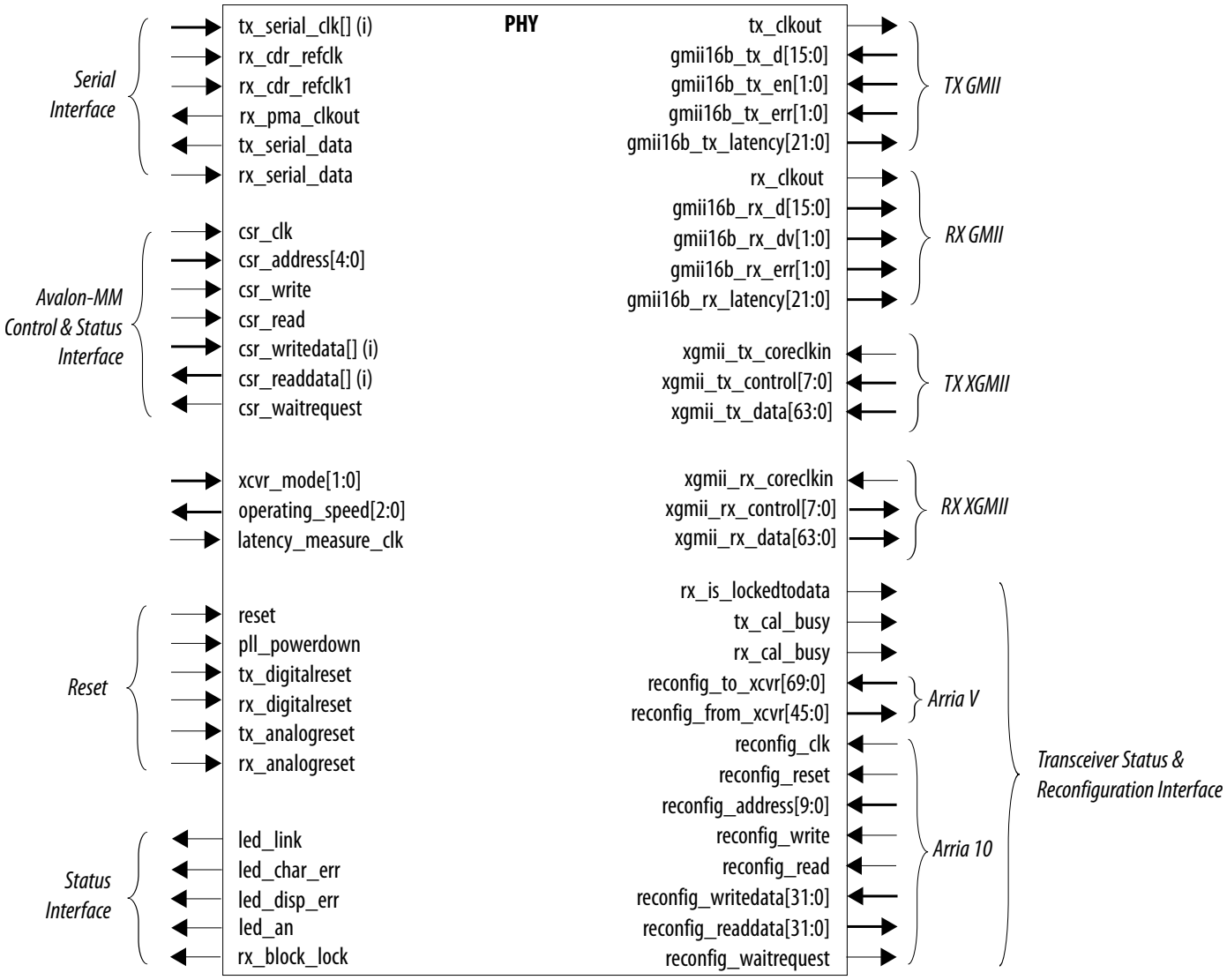
Addr	Name	Description	Access	HW Reset Value
0x401	usxgmii_status	Status Register	—	—
		Bit [1:0]: Reserved	—	—
		Bit [2]: LINK_STATUS indicates link status for USXGMII all speeds • 1: Link is established • 0: Link synchronization is lost, a 0 is latched	RO	0x0
		Bit [3]: Reserved	—	—
		Bit [4]: Reserved	—	—
		Bit [5]: AUTO_NEGOTIATION_COMPLETE A value of 1 indicates the Auto-Negotiation process is completed.	RO	0x0
		Bit [15:6]: Reserved	—	—
		Bit [31:16]: Reserved	—	—
0x402:0x404	Reserved	—	—	—

Addr	Name	Description	Access	HW Reset Value
0x405	usxgmii_partner_ability	Device abilities advertised to the link partner during Auto-Negotiation	—	—
		Bit [0]: Reserved	—	—
		Bit [6:1]: Reserved	—	—
		Bit [7]: EEE_CLOCK_STOP_CAPABILITY Indicates whether or not energy efficient ethernet (EEE) clock stop is supported. • 0: Not supported • 1: Supported	RO	0x0
		Bit [8]: EEE_CAPABILITY Indicates whether or not EEE is supported. • 0: Not supported • 1: Supported	RO	0x0
		Bit [11:9]: SPEED • 3'b000: 10M • 3'b001: 100M • 3'b010: 1G • 3'b011: 10G • 3'b100: 2.5G • 3'b101: 5G • 3'b110: Reserved • 3'b111: Reserved	RO	0x0
		Bit [12]: DUPLEX Indicates the duplex mode. • 0: Half duplex • 1: Full duplex	RO	0x0
		Bit [13]: Reserved	—	—
		Bit [14]: ACKNOWLEDGE A value of 1 indicates that the device has received three consecutive matching ability values from its link partner.	RO	0x0
		Bit [15]: LINK Indicates the link status. • 0: Link down • 1: Link up	RO	0x0
		Bit [31:16]: Reserved	—	—

Addr	Name	Description	Access	HW Reset Value
0x406:0x411	Reserved	—	—	—
0x412	usxgmii_link_timer	Auto-Negotiation link timer. Sets the link timer value in bit [19:14] from 0 to 2 ms in approximately 0.05 ms steps. You must program the link timer to ensure that it matches the link timer value of the external NBASE-T PHY IP Core. The reset value sets the link timer to approximately 1.6 ms. Bits [13:0] are reserved and always set to 0.	[19:14]: RW [13:0]: RO	[19:14]: 0x1F [13:0]: 0x0
0x413:0x41F	Reserved	—	—	—
0x461	phy_serial_loopback	Configures the transceiver serial loopback in the PMA from TX to RX.	—	—
		Bit [0] • 0: Disables the PHY serial loopback • 1: Enables the PHY serial loopback	RW	0x0
		Bit [15:1]: Reserved	—	—
		Bit [31:16]: Reserved	—	—

Interface Signals

Figure 6-2: PHY Interface Signals



Note: (i) The width depends on the PHY's operating mode.

Clock and Reset Signals

Table 6-7: Clock and Reset Signals

Signal Name	Direction	Width	Description
Clock signals			

Signal Name	Direction	Width	Description
tx_clkout	Output	1	GMII TX clock, derived from tx_serial_clk[1:0]. Provides 156.25 MHz timing reference for 2.5GbE; 62.5 MHz for 1GbE.
rx_clkout	Output	1	GMII RX clock, derived from tx_serial_clk[1:0]. Provides 156.25 MHz timing reference for 2.5GbE; 62.5 MHz for 1GbE.
csr_clk	Input	1	Clock for the control and status Avalon memory-mapped interface. Intel recommends 125 – 156.25 MHz for this clock.
xgmii_tx_coreclkin	Input	1	XGMII TX clock. Provides 156.25 MHz timing reference for 10GbE. Synchronous to tx_serial_clk with zero ppm.
xgmii_rx_coreclkin	Input	1	XGMII RX clock. Provides 156.25 MHz timing reference for 10GbE.
latency_measure_clk	Input	1	Sampling clock for measuring the latency of the 16-bit GMII datapath. This clock operates at 80 MHz and is available only when the IEEE 1588v2 feature is enabled.
tx_serial_clk	Input	1-3	Serial clock from transceiver PLLs. • 2.5GbE: Connect bit [0] to the transceiver PLL. This clock operates at 1562.5 MHz. • 1GbE: Connect bit [1] to the transceiver PLL. This clock operates at 625 MHz. • 10GbE: Connect bit [2] to the transceiver PLL. This clock operates at 5156.25 MHz.
rx_cdr_refclk	Input	1	125 MHz RX CDR reference clock for 1GbE and 2.5GbE
rx_cdr_refclk_1	Input	1	RX CDR reference clock for 10GbE. The frequency of this clock can be either 322.265625 MHz or 644.53125 MHz, as specified by the **Reference clock frequency for 10 GbE (MHz)** parameter setting.

Signal Name	Direction	Width	Description
rx_pma_clkout	Output	1	Recovered clock from CDR, operates at the following frequency: • 1GbE: 125 MHz • 2.5GbE: 312.5 MHz • 10GbE: 322.265625 MHz
Reset signals			
reset	Input	1	Active-high global reset. Assert this signal to trigger an asynchronous global reset.
tx_analogreset	Input	1	Connect this signal to the Transceiver PHY Reset Controller IP core. When asserted, triggers an asynchronous reset to the analog block on the TX path.
tx_digitalreset	Input	1	Connect this signal to the Transceiver PHY Reset Controller IP core. When asserted, triggers an asynchronous reset to the digital logic on the TX path.
rx_analogreset	Input	1	Connect this signal to the Transceiver PHY Reset Controller IP core. When asserted, triggers an asynchronous reset to the receiver CDR.
rx_digitalreset	Input	1	Connect this signal to the Transceiver PHY Reset Controller IP core. When asserted, triggers an asynchronous reset to the digital logic on the RX path.
pll_powerdown	Input	2	Assert this signal to power down the CMU PLLs in Arria V transceivers. CMU PLLs provide high-speed serial and low-speed parallel clocks to the transceiver channels.

Operating Mode and Speed Signals

Table 6-8: Transceiver Mode and Operating Speed Signals

Signal Name	Direction	Width	Description
`xcvr_mode`	Input	2	Connect this signal to the reconfiguration block. Use the values below to set the speed: • 0x0 = 1G • 0x1 = 2.5G • 0x3 = 10G
`operating_speed`	Output	3	Connect this signal to the MAC. This signal provides the current operating speed of the PHY: • 0x0 = 10G • 0x1 = 1G • 0x4 = 2.5G • 0x5 = 5G

GMII Signals

The 16-bit TX and RX GMII supports 1GbE and 2.5GbE at 62.5 MHz and 156.25 MHz respectively.

Table 6-9: GMII Signals

Signal Name	Direction	Width	Description
TX GMII signals—synchronous to `tx_clkout`			
`gmii16b_tx_d`	Input	16	TX data from the MAC. The MAC sends the lower byte first followed by the upper byte.
`gmii16b_tx_en`	Input	2	When asserted, indicates the start of a new frame from the MAC. Bit[0] corresponds to `gmii16b_tx_d`[7:0]; bit[1] corresponds to `gmii16b_tx_d`[15:8]. This signal remains asserted until the PHY receives the last byte of the data frame.
`gmii16b_tx_err`	Input	2	When asserted, indicates an error. Bit[0] corresponds to `gmii16b_tx_err`[7:0]; bit[1] corresponds to `gmii16b_tx_err`[15:8]. The bits can be asserted at any time during a frame transfer to indicate an error in the current frame.

Signal Name	Direction	Width	Description
gmii16b_tx_latency	Output	22	The latency of the PHY excluding the PMA block on the TX datapath: • Bits [21:10]: The number of clock cycles. • Bits [9:0]: The fractional number of clock cycles. This signal is available when only the **Enable IEEE 1588 Precision Time Protocol** parameter is selected.
RX GMII signals—synchronous to rx_clkout			
gmii16b_rx_d	Output	16	RX data to the MAC. The PHY sends the lower byte first followed by the upper byte. Rate matching is done by the PHY on the RX data from the RX recovered clock to rx_clkout.
gmii16b_rx_err	Output	2	When asserted, indicates an error. Bit[0] corresponds to gmii16b_rx_err[7:0]; bit[1] corresponds to gmii16b_rx_err[15:8]. The bits can be asserted at any time during a frame transfer to indicate an error in the current frame.
gmii16b_rx_dv	Output	2	When asserted, indicates the start of a new frame. Bit[0] corresponds to gmii16b_rx_d[7:0]; bit[1] corresponds to gmii16b_rx_d[15:8]. This signal remains asserted until the PHY sends the last byte of the data frame.
gmii16b_rx_latency	Output	22	The latency of the PHY excluding the PMA block on the RX datapath: • Bits [21:10]: The number of clock cycles. • Bits [9:0]: The fractional number of clock cycles. This signal is available only when the **Enable IEEE 1588 Precision Time Protocol** parameter is selected.

XGMII Signals

The XGMII supports 10GbE at 156.25 MHz.

Table 6-10: XGMII Signals

Signal Name	Direction	Width	Description
TX XGMII signals—synchronous to `xgmii_tx_coreclkin`			
`xgmii_tx_data`	Input	64, 32	TX data from the MAC. The MAC sends the data in the following order: bits[7:0], bits[15:8], and so forth. The width is: • 64 bits for 1G/2.5G/10G configurations.
`xgmii_tx_control`	Input	8, 4	TX control from the MAC: • `xgmii_tx_control`[0] corresponds to `xgmii_tx_data`[7:0] • `xgmii_tx_control`[1] corresponds to `xgmii_tx_data`[15:8] • and so forth. The width is: • 8 bits for 1G/2.5G/10G configurations.
`xgmii_tx_valid`	Input	1	Indicates valid data on `xgmii_tx_control` and `xgmii_tx_data` from the MAC. Your logic/MAC must toggle the valid data as shown below: Speed: 1G — Asserted once every 10 clock cycles; 2.5G — Asserted once every 4 clock cycles; 5G — Asserted once every 2 clock cycles; 10G — Asserted in every clock cycle
RX XGMII signals—synchronous to `xgmii_rx_coreclkin`			
`xgmii_rx_data`	Output	64, 32	RX data to the MAC. The PHY sends the data in the following order: bits[7:0], bits[15:8], and so forth. The width is: • 64 bits for 1G/2.5G/10G configurations.

Signal Name	Direction	Width	Description
xgmii_rx_control	Output	8, 4	RX control to the MAC. • xgmii_rx_control[0] corresponds to xgmii_rx_data[7:0] • xgmii_rx_control[1] corresponds to xgmii_rx_data[15:8] • and so forth. The width is: • 8 bits for 1G/2.5G/10G configurations.
xgmii_rx_valid	Output	1	Indicates valid data on xgmii_rx_control and xgmii_rx_data from the MAC. The toggle rate from the PHY is shown in the table below. **Note:** The toggle rate may vary when the start of a packet is received or when rate match occurs inside the PHY. You should not expect the valid data pattern to be fixed. Speed / Toggle Rate: 1G: Asserted once every 10 clock cycles 2.5G: Asserted once every 4 clock cycles 5G: Asserted once every 2 clock cycles 10G: Asserted in every clock cycle

Status Signals

Table 6-11: Status Signals

Signal Name	Direction	Clock Domain	Width	Description
led_char_err	Output	Synchronous to rx_clkout	1	Asserted when a 10-bit character error is detected in the RX data.
led_link	Output	Synchronous to tx_clkout	1	Asserted when the link synchronization for 1GbE or 2.5GbE is successful

Signal Name	Direction	Clock Domain	Width	Description
led_disp_err	Output	Synchronous to rx_clkout	1	Asserted when a 10-bit running disparity error is detected in the RX data.
led_an	Output	Synchronous to rx_clkout	1	Asserted when auto-negotiation is completed.
rx_block_lock	Output	Synchronous to rx_clkout	1	Asserted when the link synchronization for 10GbE is successful.

Serial Interface Signals

The serial interface connects to an external device.

Table 6-12: Serial Interface Signals

Signal Name	Direction	Width	Description
tx_serial_data	Output	1	TX data.
rx_serial_data	Input	1	RX data.

Transceiver Status and Reconfiguration Signals

Table 6-13: Control and Status Signals

Signal Name	Direction	Width	Description
rx_is_lockedtodata	Output	1	Asserted when the CDR is locked to the RX data.
tx_cal_busy	Output	1	Asserted when TX calibration is in progress.
rx_cal_busy	Output	1	Asserted when RX calibration is in progress.
Transceiver reconfiguration signals for Arria V devices			
reconfig_to_xcvr	Input	70	Reconfiguration signals from the reconfiguration block.
reconfig_from_xcvr	Output	46	Reconfiguration signals to the reconfiguration block.

Avalon Memory-Mapped Interface Signals

The Avalon memory-mapped interface is an Avalon memory-mapped interface slave port. This interface uses word addressing and provides access to the 16-bit configuration registers of the PHY.

Table 6-14: Avalon Memory-Mapped Interface Signals

Signal Name	Direction	Width	Description
csr_address	Input	5, 11	Use this bus to specify the register address to read from or write to. The width is: • 5 bits for 2.5G and 1G/2.5G configurations.
csr_read	Input	1	Assert this signal to request a read operation.
csr_readdata	Output	16, 32	Data read from the specified register. The data is valid only when the csr_waitrequest signal is deasserted. The width is: • 16 bits for 2.5G and 1G/2.5G configurations.
csr_write	Input	1	Assert this signal to request a write operation.
csr_writedata	Input	16, 32	Data to be written to the specified register. The data is written only when the csr_waitrequest signal is deasserted. The width is: • 16 bits for 2.5G and 1G/2.5G configurations.
csr_waitrequest	Output	1	When asserted, indicates that the PHY is busy and not ready to accept any read or write requests. • When you have requested for a read or write, keep the control signals to the Avalon memory-mapped interface constant while this signal is asserted. The request is complete when it is deasserted. • This signal can be high or low during idle cycles and reset. Therefore, the user application must not make any assumption of its assertion state during these periods.

XAUI PHY IP Core 7

2020.06.02

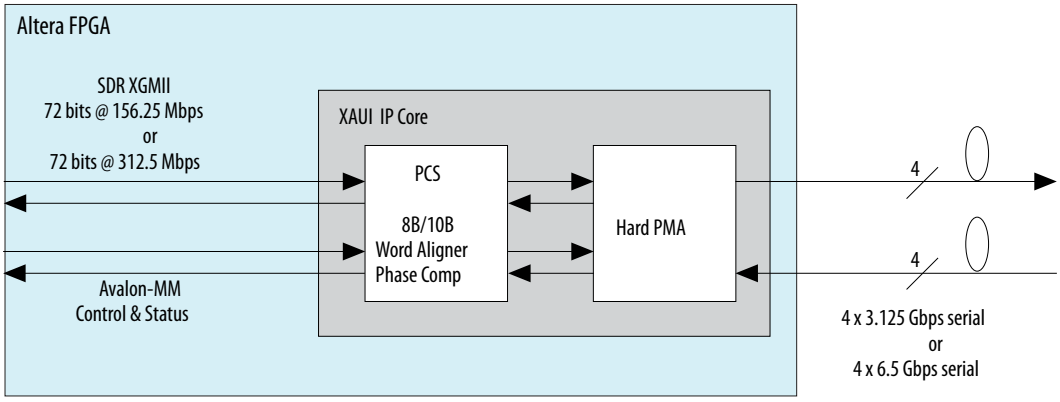
UG-01080

Subscribe Send Feedback

The Altera XAUI PHY IP Core implements the IEEE 802.3 Clause 48 specification to extend the operational distance of the XGMII interface and reduce the number of interface signals.

XAUI extends the physical separation possible between the 10 Gbps Ethernet MAC function and the Ethernet standard PHY component to one meter. The XAUI IP Core accepts 72-bit data (single data rate–SDR XGMII) from the application layer at either 156.25 Mbps or 312.5 Mbps. The serial interface runs at either 4 × 3.125 Gbps or 4 × 6.25 Gbps (DDR XAUI option).

Figure 7-1: XAUI PHY IP Core



For Stratix IV GX and GT devices, you can choose a hard XAUI physical coding sublayer (PCS) and physical media attachment (PMA), or a soft XAUI PCS and PMA in low latency mode. You can also combine both hard and soft PCS configurations in the same device, using all channels in a transceiver bank. The PCS is only available in soft logic for Stratix V devices.

For more detailed information about the XAUI transceiver channel datapath, clocking, and channel placement, refer to the "*XAUI*" section in the *Transceiver Configurations in Stratix V Devices* chapter of the *Stratix V Device Handbook*.

Related Information

- **IEEE 802.3 Clause 48**
- **Transceiver Configurations in Stratix V Devices**

XAUI PHY Release Information

This section provides information about this release of the XAUI PHY IP Core.

Table 7-1: XAUI Release Information

Item	Description
Version	13.1
Release Date	November 2013
Ordering Codes[4]	P-XAUIPCS (primary)–Soft PCS IPR-XAUIPCS (renewal)–Soft PCS
Product ID	00D7
Vendor ID	6AF7

XAUI PHY Device Family Support

This section describes device family support for the IP core.

IP cores provide either final or preliminary support for target Altera device families. These terms have the following definitions:

- Final support—Verified with final timing models for this device.

Preliminary support—Verified with preliminary timing models for this device.

Table 7-2: Device Family Support

Device Family	Support
XAUI	
Arria II GX -Hard PCS and PMA	Final
Arria II GZ-Hard PCS and PMA	Final
Arria V GX-Soft PCS + PMA	Final
Arria V SoC-Soft PCS + PMA	Final
Arria V GZ devices-Soft PCS + PMA	Final
Cyclone IV GX-Hard PCS and PMA	Final
Cyclone V-Soft PCS + PMA	Final
Cyclone V SoC-Soft PCS + hard PMA	Final

[4] No ordering codes or license files are required for the hard PCS and PMA PHY in Arria II GX, Cyclone IV GX, or Stratix IV GX or GT devices.

Device Family	Support
HardCopy® IV	Final
Stratix IV GX and GT devices-Soft or hard PCS and PMA	Final
Stratix V devices-Soft PCS + PMA	Final
Other device families	No support
DXAUI	
Stratix IV GX and GT	Final
Other device families	No support

XAUI PHY Performance and Resource Utilization for Stratix IV Devices

This section describes performance and resource utilization for Stratix IV Devices

The following table shows the typical expected device resource utilization for different configurations using the current version of the Intel Quartus Prime software targeting a Stratix IV GX (EP4SG230KF40C2ES) device. The numbers of combinational ALUTs, logic registers, and memory bits are rounded to the nearest 100.

Table 7-3: XAUI PHY Performance and Resource Utilization—Stratix IV GX Device

Implementation	Number of 3.125 Gbps Channels	Combinational ALUTS	Dedicated Logic Registers	Memory Bits
Soft XAUI	4	4500	3200	5100
Hard XAUI	4	2000	1300	0

XAUI PHY Performance and Resource Utilization for Arria V GZ and Stratix V Devices

This section describes performance and resource utilization for Arria V GZ and Stratix V Devices.

For the Arria V GZ (5AGZME5K2F40C3) device, the XAUI PHY uses 1% of ALMs and less than 1% of M20K memory, primary and secondary logic registers. Resource utilization is similar for Stratix V devices.

Parameterizing the XAUI PHY

Complete the following steps to configure the XAUI PHY IP Core:

1. Under **Tools** > **IP Catalog**, select the device family of your choice.
2. Under **Tools** > **IP Catalog** > **Interfaces** > **Ethernet** select **XAUI PHY**.
3. Use the tabs on the MegaWizard Plug-In Manager to select the options required for the protocol.
4. Refer the following topics to learn more about the parameters:

a. General Parameters
b. Analog Parameters
c. Advanced Options Parameters

5. Click **Finish** to generate your customized XAUI PHY IP Core.

XAUI PHY General Parameters

This section describes the settings available on **General Options** tab.

Table 7-4: General Options

Name	Value	Description
Device family	**Arria II GX** **Arria V** **Arria V GZ** **Cyclone IV GX** **Cyclone V** **HardCopy IV** **Stratix IV** **Stratix V**	The target device family.
Starting channel number	**0-124**	The physical starting channel number in the Altera device for channel 0 of this XAUI PHY. In Arria II GX, Cyclone IV GX, HardCopy IV, and Stratix IV devices, this starting channel number must be 0 or a multiple of 4. In Arria V GZ and Stratix V devices, logical lane 0 should be assigned to either physical transceiver channel 1 or channel 4 of a transceiver bank. However, if you have already created a PCB with a different lane assignment for logical lane 0, you can use the workaound shown in **Example 7-1** to remove this restriction. Assignment of the starting channel number is required for serial transceiver dynamic reconfiguration. Check logical channel 0 restrictions in Cyclone 5 and Arria 5.

Name	Value	Description
XAUI interface type	**Hard XAUI** **Soft XAUI** **DDR XAUII**	The following 3 interface types are available: • **Hard XAUI**–Implements the PCS and PMA in hard logic. Available for Arria II, Cyclone IV, HardCopy IV, and Stratix IV devices. • **Soft XAUI**–Implements the PCS in soft logic and the PMA in hard logic. Available for HardCopy IV, Stratix IV, Arria V, Cyclone V, and Stratix V devices. • **DDR XAUI**–Implements the PCS in soft logic and the PMA in hard logic. Both the application and serial interfaces run at twice the frequency of the Soft XAUI options. Available for HardCopy IV Stratix IV devices. All interface types include 4 channels.
Data rate	Device Dependent	Specifies the data rate.
PLL type	CMU ATX	You can select either the CMU or ATX PLL. The CMU PLL has a larger frequency range than the ATX PLL. The ATX PLL is designed to improve jitter performance and achieves lower channel-to-channel skew; however, it supports a narrower range of data rates and reference clock frequencies. Another advantage of the ATX PLL is that it does not use a transceiver channel, while the CMU PLL does. This parameter is available for the soft PCS and DDR XAUI. The ATX PLL is not available for all devices.
Base data rate	1 × **Lane rate** 2 × **Lane rate** 4 × **Lane rate**	The **base data rate** is the frequency of the clock input to the PLL. Select a **base data rate** that minimizes the number of PLLs required to generate all the clock s required for data transmission. By selecting an appropriate **base data rate**, you can change data rates by changing the divider used by the clock generation block. This parameter is available for Stratix V devices.
Number of XAUI interfaces	1	Specifies the number of XAUI interfaces. Only 1 is available in the current release.

Example 7-1 shows how to remove the restriction on logical lane 0 channel assignment in Stratix V devices by redefining the pma_bonding_master parameter using the Intel Quartus Prime Assignment

Editor. In this example, the pma_bonding_master was originally assigned to physical channel 1. (The original assignment could also have been to physical channel 4.) The to parameter reassigns the pma_bonding_master to the XAUI instance name shown in quotation marks. You must substitute the instance name from your design for the instance name shown in quotation marks

Example 7-1: Overriding Logical Lane 0 Channel Assignment Restrictions in Stratix V Devices

```
set_parameter -name pma_bonding_master "\"1\"" -to "<xaui
instance name>|sv_xcvr_xaui:alt_xaui_phy|sv_xcvr_low_latency_phy_nr:
alt_pma_0|sv_xcvr_custom_native:sv_xcvr_custom_inst|sv_xcvr_native:
gen.sv_xcvr_native_insts[0].gen_bonded_group.sv_xcvr_native_inst"
```

XAUI PHY Analog Parameters

This section describes the analog parameters for the IP core.

Click on the appropriate link to specify the analog options for your device:

- **XAUI PHY Analog Parameters for Arria II GX, Cyclone IV GX, HardCopy IV and Stratix IV Devices** on page 7-6

Related Information

- **Analog Settings for Arria V Devices** on page 20-2
- **Analog Settings for Arria V GZ Devices** on page 20-11
- **Analog Settings for Cyclone V Devices** on page 20-26
- **Analog Settings for Stratix V Devices** on page 20-35

XAUI PHY Analog Parameters for Arria II GX, Cyclone IV GX, HardCopy IV and Stratix IV Devices

This section describes parameters for the Arria II GX, Cyclone IV GX, and Stratix IV devices; specify your analog options on the **Analog Options** tab.

Table 7-5: PMA Analog Options

Name	Value	Description
Transmitter termination resistance	**OCT_85_OHMS** **OCT_100_OHMS** **OCT_120_OHMS** **OCT_150_OHMS**	Indicates the value of the termination resistor for the transmitter.
Transmitter VOD control setting	**0–7**	Sets V_{OD} for the various TX buffers.
Pre-emphasis pre-tap setting	**0–7**	Sets the amount of pre-emphasis on the TX buffer. Available for Stratix IV.

Name	Value	Description
Invert the pre-emphasis pre-tap polarity setting	**On** **Off**	Determines whether or not the pre-emphasis control signal for the pre-tap is inverted. If you turn this option on, the pre-emphasis control signal is inverted. Available for HardCopy IV and Stratix IV devices.
Pre-emphasis first post-tap setting	**0–15**	Sets the amount of pre-emphasis for the 1st post-tap.
Pre-emphasis second post-tap setting	**0–7**	Sets the amount of pre-emphasis for the 2nd post-tap. Available for HardCopy IV and Stratix IV devices.
Invert the pre-emphasis second post-tap polarity	**On** **Off**	Determines whether or not the pre-emphasis control signal for the second post-tap is inverted. If you turn this option on, the pre-emphasis control signa is inverted. Available for HardCopy IV and Stratix IV devices.
Receiver common mode voltage	Tri-state 0.82V 1.1v	Specifies the RX common mode voltage.
Receiver termination resistanc	**OCT_85_OHMS** **OCT_100_OHMS** **OCT_120_OHMS** **OCT_150_OHMS**	Indicates the value of the termination resistor for the receiver. Cyclone IV supports 100 and 150.
Receiver DC gain	**0–4**	Sets the equalization DC gain using one of the following settings: • 0–0 dB • 1–3 dB • 2–6 dB • 3–9 dB • 4–12 dB
Receiver static equalizer setting	**0–15**	This option sets the equalizer control settings. The equalizer uses a pass band filter. Specifying a low value passes low frequencies. Specifying a high value passes high frequencies. Available for HardCopy IV and Stratix IV devices.

Advanced Options Parameters

This section describes the settings available on the **Advanced Options** tab.

Table 7-6: Advanced Options

Name	Value	Description
Include control and status ports	**On/Off**	If you turn this option on, the top-level IP core include the status signals and digital resets shown in XAUI Top-Level Signals—Soft PCS and PMA and XAUI Top-Level Signals–Hard IP PCS and PMA . If you turn this option off, you can access control and status information using Avalon-MM interface to the control and status registers. The default setting is off.
External PMA control and configuration	**On/Off**	If you turn this option on, the PMA signals are brought up to the top level of the XAUI IP Core. This option is useful if your design includes multiple instantiations of the XAUI PHY IP Core. To save FPGA resources, you can instantiate the Low Latency PHY Controller and Transceiver Reconfiguration Controller IP Cores separately in your design to avoid having these IP cores instantiated in each instance of the XAUI PHY IP Core. If you turn this option off, the PMA signals remain internal to the core. The default setting is off. This option is available for Arria II GX, HardCopy IV and Stratix IV devices. In these devices, this option must be turned **On** to fit 2 hard XAUI instances in adjacent transceiver quads that share the same calibration block. In addition, the instances must share powerdown signals.
Enable rx_recovered_clk pin	**On/Off**	When you turn this option on, the RX recovered clock signal is an output signal.

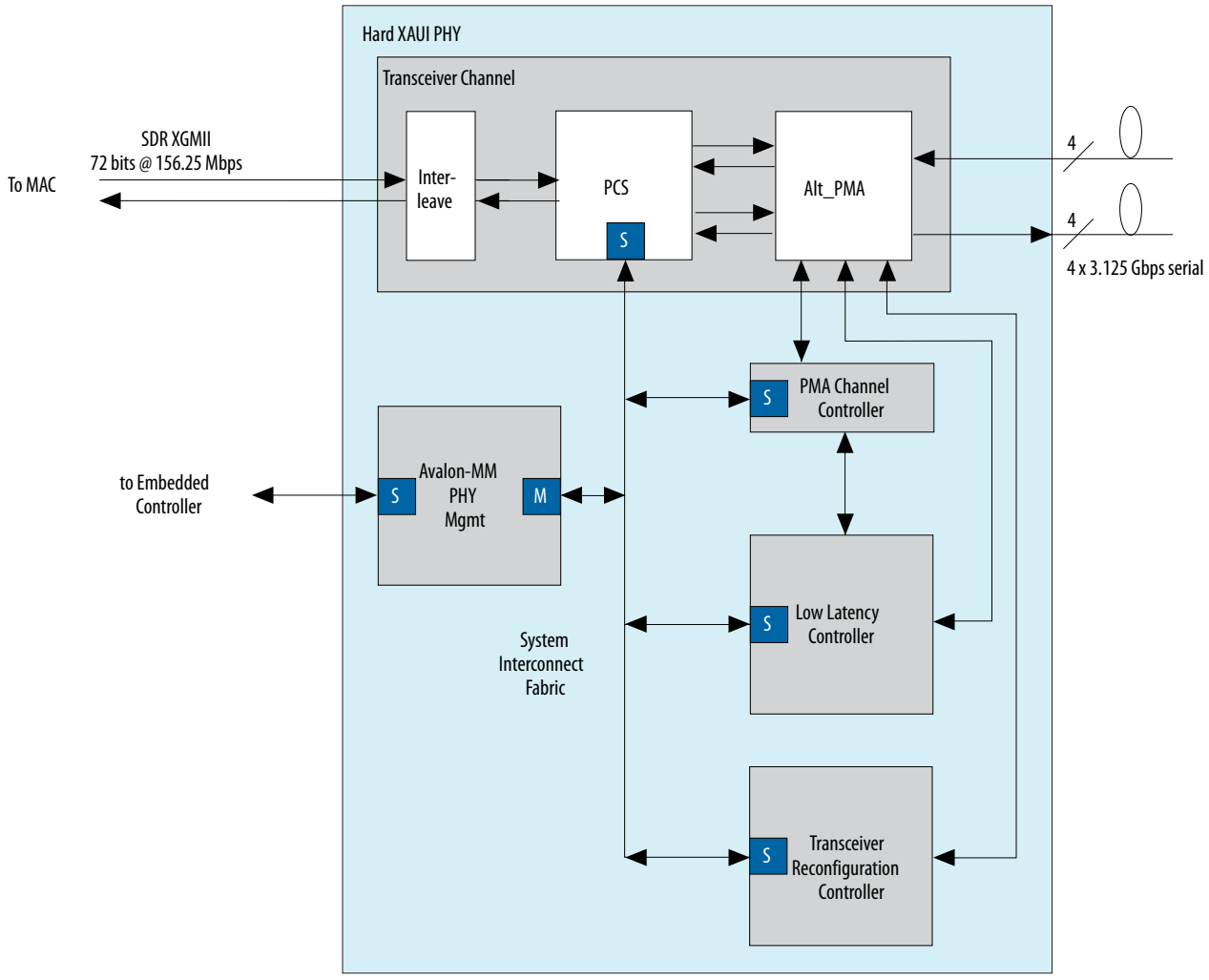
XAUI PHY Configurations

This section describes configurations of the IP core.

The following figure illustrates one configuration of the XAUI IP Core. As this figure illustrates, if your variant includes a single instantiation of the XAUI IP Core, the transceiver reconfiguration control logic is included in the XAUI PHY IP Core. For Arria V, Cyclone V, and Stratix V devices the Transceiver Reconfi-

guration Controller must always be external. Refer to Chapter 16, Transceiver Reconfiguration Controller IP Core for more information about this IP core. The Transceiver Reconfiguration Controller is always separately instantiated in Stratix V and Arria V GZ devices.

Figure 7-2: XAUI PHY with Internal Transceiver Reconfiguration Control



Related Information

Transceiver Reconfiguration Controller IP Core Overview on page 17-1

XAUI PHY Ports

This section describes the ports for the IP core.

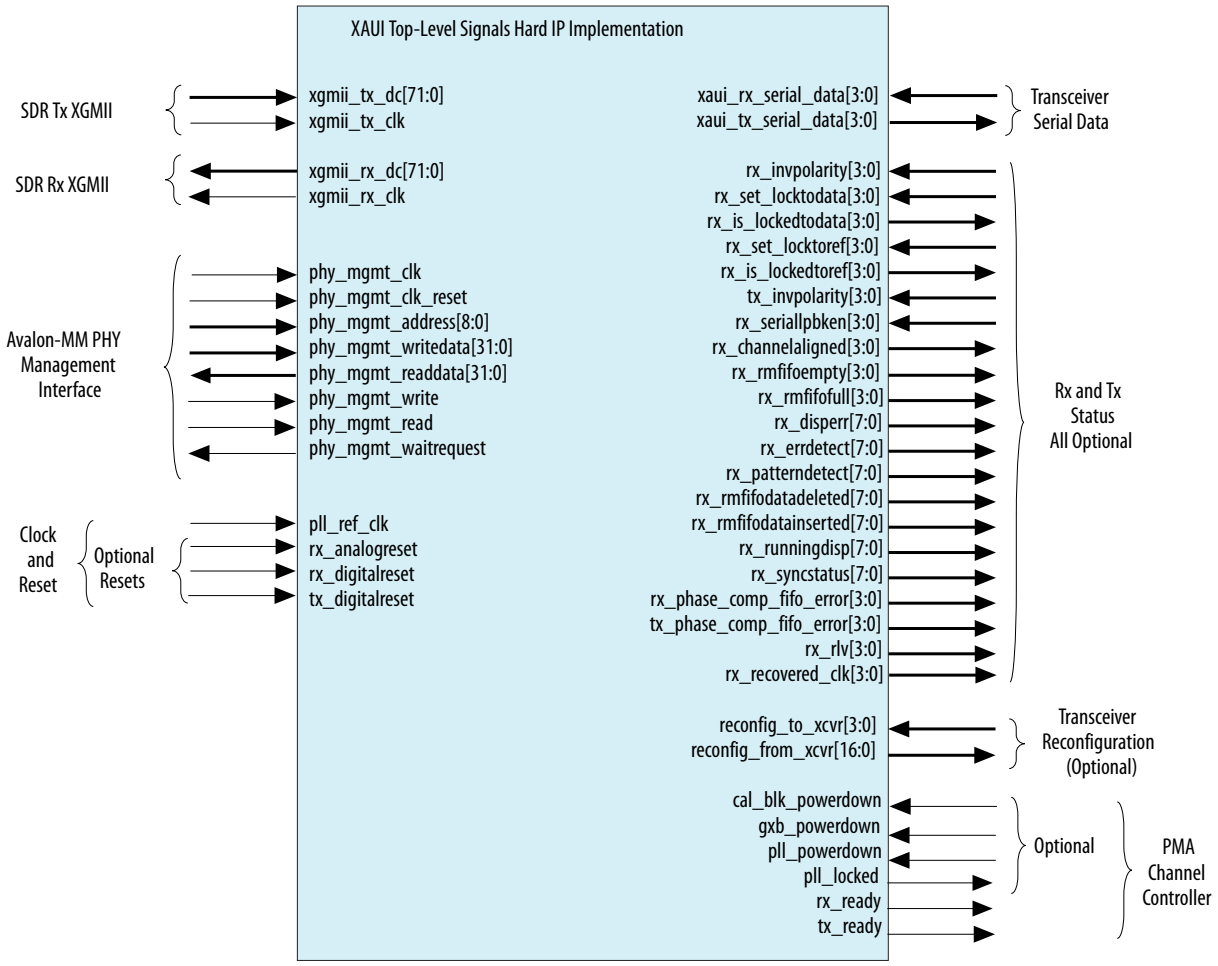
Figure 7-3 illustrates the top-level signals of the XAUI PHY IP Core for the hard IP implementation. This variant is available for Arria II GX, Cyclone IV GX, HardCopy IV and Stratix IV GX devices.**Figure 7-4** illustrates the top-level signals of the XAUI PHY IP Core for the soft IP implementation. With the exception of the optional signals available for debugging and the signals for dynamic reconfiguration of

the transceivers, the top-level signals of the two variants is nearly identical. The DDR XAUI soft IP signals and behavior are the same as the soft IP implementation.

The block diagram shown in the MegaWizard Plug-In Manager GUI labels the external pins with the interface type and places the interface name inside the box. The interface type and name are used to define component interfaces in the **_hw.tcl**. If you turn on Show signals, the block diagram displays all top-level signal names.

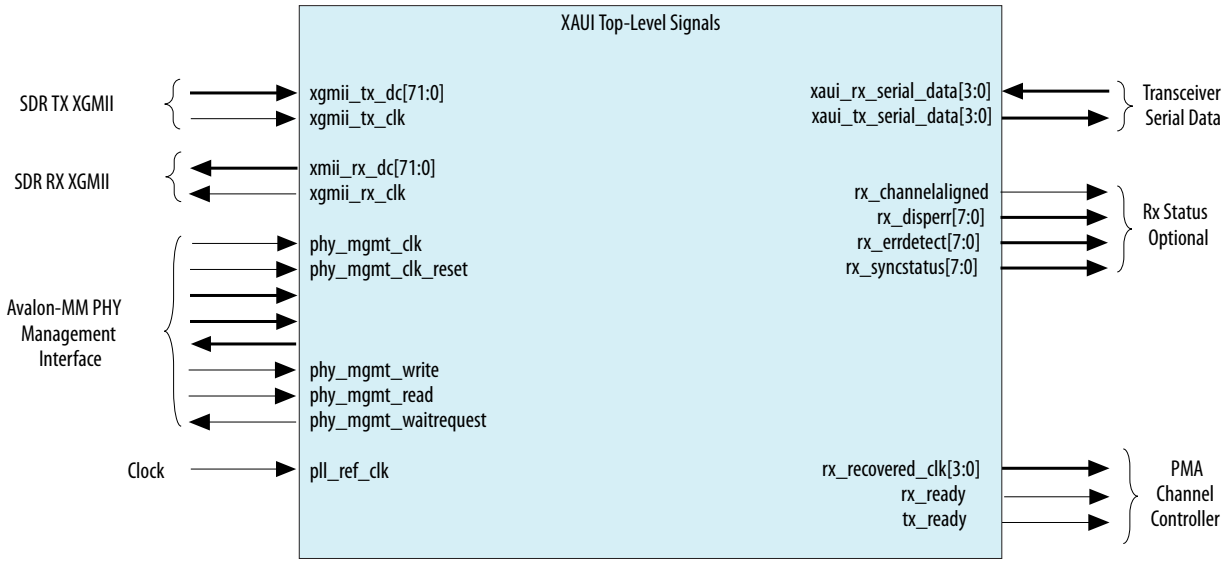
For more information about **_hw.tcl** files refer to refer to the Component Interface Tcl Reference chapter in volume 1 of the Intel Quartus Prime Handbook.

Figure 7-3: XAUI Top-Level Signals–Hard IP PCS and PMA



The following figure illustrates the top-level signals of the XAUI PHY IP Core for the soft IP implementation for both the single and DDR rates.

Figure 7-4: XAUI Top-Level Signals—Soft PCS and PMA



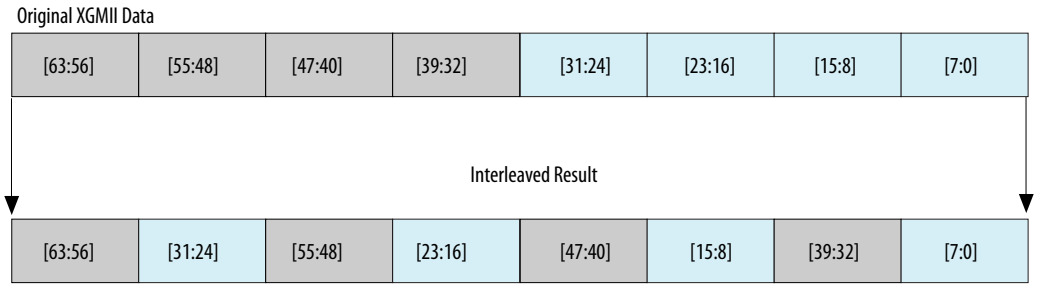
XAUI PHY Data Interfaces

The XAUI PCS interface to the FPGA fabric uses a SDR XGMII interface. This interface implements a simple version of Avalon-ST protocol. The interface does not include ready or valid signals; consequently, the sources always drive data and the sinks must always be ready to receive data.

For more information about the Avalon-ST protocol, including timing diagrams, refer to the *Avalon Interface Specifications*.

Depending on the parameters you choose, the application interface runs at either 156.25 Mbps or 312.5 Mbps. At either frequency, data is only driven on the rising edge of clock. To meet the bandwidth requirements, the datapath is eight bytes wide with eight control bits, instead of the standard four bytes of data and four bits of control. The XAUI IP Core treats the datapath as two, 32-bit data buses and includes logic to interleave them, starting with the low-order bytes.

Figure 7-5: Interleaved SDR XGMII Data Mapping



For the DDR XAUI variant, the start of control character (0xFB) is aligned to either byte 0 or byte 5.

Figure 7-6: Byte 0 Start of Frame Transmission Example

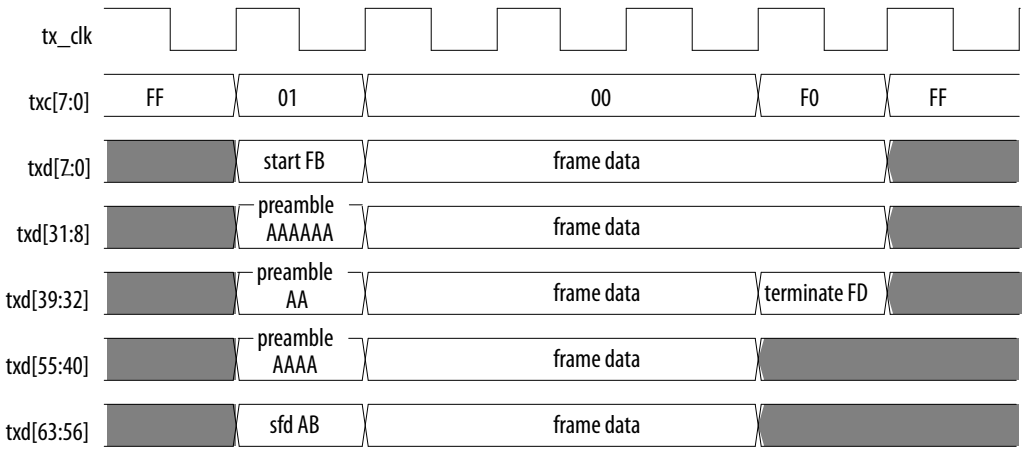
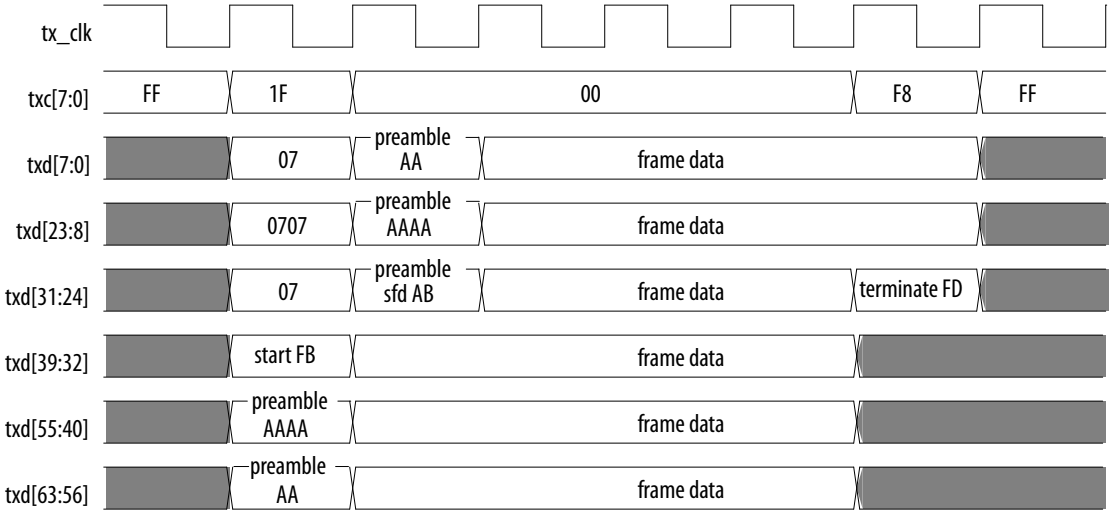


Figure 7-7: Byte 5 Start of Frame Transmission Example



Related Information

Avalon Interface Specifications

SDR XGMII TX Interface

This section describes the signals in the SDR TX XGMII interface.

Table 7-7: SDR TX XGMII Interface

Signal Name	Direction	Description
xgmii_tx_dc[71:0]	Output	Contains 4 lanes of data and control for XGMII. Each lane consists of 16 bits of data and 2 bits of control. • Lane 0–[7:0]/[8], [43:36]/[44] • Lane 1–[16:9]/[17], [52:45]/[53] • Lane 2–[25:18]/[26], [61:54]/[62] • Lane 3–[34:27]/[35],[70:63]/[71]
xgmii_tx_clk	Input	The XGMII SDR TX clock which runs at 156.25 MHz or 312.5 for the DDR variant.

SDR XGMII RX Interface

This section describes the signals in the SDR RX XGMII interface.

Table 7-8: SDR RX XGMII Interface

Signal Name	Direction	Description
xgmii_rx_dc_[71:0]	Input	Contains 4 lanes of data and control for XGMII. Each lane consists of 16 bits of data and 2 bits of control. • Lane 0–[7:0]/[8], [43:36]/[44] • Lane 1–[16:9]/[17], [52:45]/[53] • Lane 2–[25:18]/[26], [61:54]/[62] • Lane 3–[34:27]/[35],[70:63]/[71]
xgmii_rx_clk	Output	The XGMII SDR RX clock which runs at 156.25 MHz.

Transceiver Serial Data Interface

This section describes the signals in the XAUI transceiver serial data interface.

The XAUI transceiver serial data interface has four lanes of serial data for both the TX and RX interfaces. This interface runs at 3.125 GHz or 6.25 GHz depending on the variant you choose. There is no separate clock signal because it is encoded in the data.

Table 7-9: Serial Data Interface

Signal Name	Direction	Description
xaui_rx_serial_data[3:0]	Input	Serial input data.
xaui_tx_serial_data[3:0]	Output	Serial output data.

XAUI PHY Clocks, Reset, and Powerdown Interfaces

This section describes the clocks, reset, and oowerdown interfaces.

Figure 7-8: Clock Inputs and Outputs for IP Core with Hard PCS

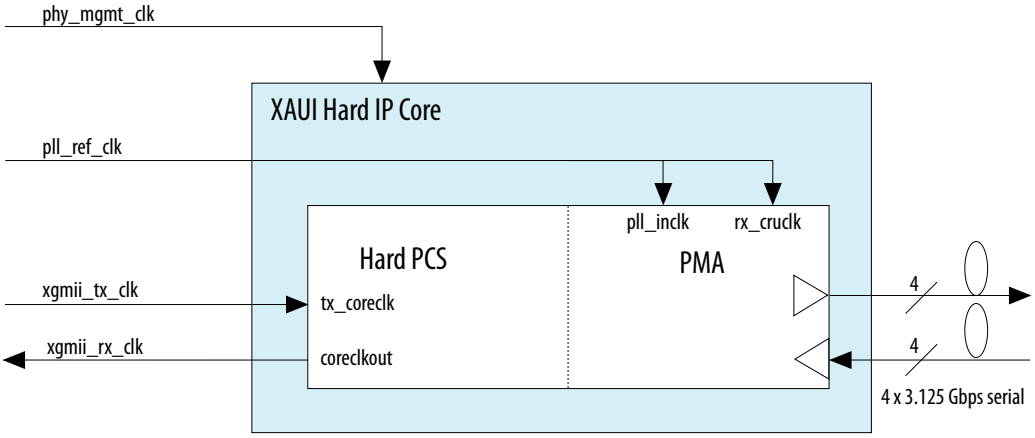


Figure 7-9: Clock Inputs and Outputs for IP Core with Soft PCS

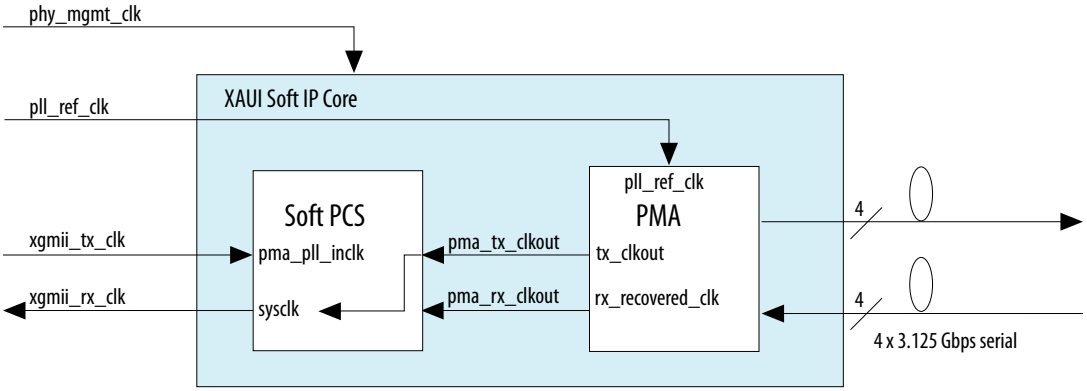


Table 7-10: Optional Clock and Reset Signals

Signal Name	Direction	Description
`pll_ref_clk`	Input	This is a 156.25 MHz reference clock that is used by the TX PLL and CDR logic.
`rx_analogreset`	Input	This signal resets the analog CDR and deserializer logic in the RX channel. It is available only for the hard IP implementation.
`rx_digitalreset`	Input	PCS RX digital reset signal. It is available only for the hard IP implementation.

Signal Name	Direction	Description
tx_digitalreset	Input	PCS TX digital reset signal. If your design includes bonded TX PCS channels, refer to Timing Constraints for Reset Signals when Using Bonded PCS Channels for a SDC constraint you must include in your design. It is available only for the hard IP implementation.
xgmii_tx_clk	Input	The XGMII TX clock which runs at 156.25 MHz. Connect xgmii_tx_clk to xgmii_rx_clk to guarantee this clock is within 150 ppm of the transceiver reference clock.
xgmii_rx_clk	Output	This clock is generated by the same reference clock that is used to generate the transceiver clock. Its frequency is 156.25 MHz. Use this clock for the MAC interface to minimize the size of the FIFO between the MAX and SDR XGMII RX interface.

Refer to Transceiver Reconfiguration Controller for additional information about reset.

Related Information

Transceiver Reconfiguration Controller IP Core Overview on page 17-1

XAUI PHY PMA Channel Controller Interface

This sectiondescribes the signals in the PMA channel controller interface.

Table 7-11: PMA Channel Controller Signals

Signal Name	Direction	Description
cal_blk_powerdown	Input	Powers down the calibration block. A high-to-low transition on this signal restarts calibration. Only available in Arria II GX, HardCopy IV, and Stratix IV GX, and Stratix IV GT devices.
gxb_powerdown	Input	When asserted, powers down the entire transceiver block. Only available in Arria II GX, HardCopy IV, and Stratix IV GX, and Stratix IV GT devices.
pll_powerdown	Input	Powers down the CMU PLL. Only available in Arria II GX, HardCopy IV, and Stratix IV GX, and Stratix IV GT devices.
pll_locked	Output	Indicates CMU PLL is locked. Only available in Arria II GX, HardCopy IV, and Stratix IV GX, and Stratix IV GT devices.
rx_recovered_clk[3:0]	Output	This is the RX clock which is recovered from the received data stream.

Signal Name	Direction	Description
rx_ready	Output	Indicates PMA RX has exited the reset state and the transceiver can receive data.
tx_ready	Output	Indicates PMA TX has exited the reset state and the transceiver can transmit data.

XAUI PHY Optional PMA Control and Status Interface

You can access the state of the optional PMA control and status signals available in the soft IP implementation using the Avalon-MM PHY Management interface to read the control and status registers which are detailed in XAUI PHY IP Core Registers . However, in some cases, you may need to know the instantaneous value of a signal to ensure correct functioning of the XAUI PHY. In such cases, you can include the required signal in the top-level module of your XAUI PHY IP Core.

Table 7-12: Optional Control and Status Signals—Soft IP Implementation

Signal Name	Direction	Description
rx_channelaligned	Output	When asserted, indicates that all 4 RX channels are aligned.
rx_disperr[7:0]	Output	Received 10-bit code or data group has a disparity error. It is paired with rx_errdetect which is also asserted when a disparity error occurs. The rx_disperr signal is 2 bits wide per channel for a total of 8 bits per XAUI link.
rx_errdetect[7:0]	Output	When asserted, indicates an 8B/10B code group violation. It is asserted 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, a disparity error, or both. The rx_errdetect signal is 2 bits wide per channel for a total of 8 bits per XAUI link.
rx_syncstatus[7:0]	Output	Synchronization indication. RX synchronization is indicated on the rx_syncstatus port of each channel. The rx_syncstatus signal is 2 bits per channel for a total of 8 bits per hard XAUI link. The rx_syncstatus signal is 1 bit per channel for a total of 4 bits per soft XAUI link.
rx_is_lockedtodata[3:0]	Output	When asserted indicates that the RX CDR PLL is locked to the incoming data.

Signal Name	Direction	Description
rx_is_lockedtoref[3:0]	Output	When asserted indicates that the RX CDR PLL is locked to the reference clock.
tx_clk312_5	Output	This is the clock used for the SDR XGMII interface.

You can access the state of the PMA control and status signals available in the hard IP implementation using the Avalon-MM PHY Management interface to read the control and status registers which are detailed in XAUI PHY IP Core Registers. However, in some cases, you may need to know the instantaneous value of a signal to ensure correct functioning of the XAUI PHY. In such cases, you can include the required signal in the top-level module of your XAUI PHY IP Core.

Table 7-13: Optional Control and Status Signals—Hard IP Implementation, Stratix IV GX Devices

Name	Direction	Description
rx_invpolarity[3:0]	Input	Dynamically reverse the polarity of every bit of the RX data at the input of the word aligner.
rx_set_locktodata[3:0]	Input	Force the CDR circuitry to lock to the received data.
rx_is_lockedtodata[3:0]	Output	When asserted, indicates the RX channel is locked to input data.
rx_set_locktoref[3:0]	Input	Force the receiver CDR to lock to the phase and frequency of the input reference clock.
rx_is_lockedtoref[3:0]	Output	When asserted, indicates the RX channel is locked to input reference clock.
tx_invpolarity[3:0]	Input	Dynamically reverse the polarity the data word input to the serializer in the TX datapath.
rx_seriallpbken	Input	Serial loopback enable. • 1: Enables serial loopback • 0: Disables serial loopback This signal is asynchronous to the receiver. The status of the serial loopback option is recorded by the PMA channel controller, word address 0x061.
rx_channelaligned	Output	When asserted indicates that the RX channel is aligned.
pll_locked	Output	In LTR mode, indicates that the receiver CDR has locked to the phase and frequency of the input reference clock.
rx_rmfifoempty[3:0]	Output	Status flag that indicates 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 RX datapath.

Name	Direction	Description
rx_rmfifofull[3:0]	Output	Status flag that indicates 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 RX data.
rx_disperr[7:0]	Output	Received 10-bit code or data group has a disparity error. It is paired with rx_errdetect which is also asserted when a disparity error occurs. The rx_disperr signal is 2 bits wide per channel for a total of 8 bits per XAUI link.
rx_errdetect[7:0]	Output	Transceiver 8B/10B code group violation or disparity error indicator. If either signal is asserted, a code group violation or disparity error was detected on the associated received code group. Use the rx_disperr signal to determine whether this signal indicates a code group violation or a disparity error. The rx_errdetect signal is 2 bits wide per channel for a total of 8 bits per XAUI link.
rx_patterndetect[7:0]	Output	Indicates that the word alignment pattern programmed has been detected in the current word boundary. The rx_patterndetect signal is 2 bits wide per channel for a total of 8 bits per XAUI link.
rx_rmfifodatadeleted[7:0]	Output	Status flag that is asserted when the rate match block deletes a ||R|| column. The flag is asserted for one clock cycle per deleted ||R|| column.
rx_rmfifodatainserted[7:0]	Output	Status flag that is asserted when the rate match block inserts a ||R|| column. The flag is asserted for one clock cycle per inserted ||R|| column.
rx_runningdisp[7:0]	Output	Asserted when the current running disparity of the 8B/10B decoded byte is negative. Low when the current running disparity of the 8B/10B decoded byte is positive.
rx_syncstatus[7:0]	Output	Synchronization indication. RX synchronization is indicated on the rx_syncstatus port of each channel. The rx_syncstatus signal is 2 bits wide per channel for a total of 8 bits per XAUI link.
rx_phase_comp_fifo_error[3:0]	Output	Indicates a RX phase comp FIFO overflow or underrun condition.
tx_phase_comp_fifo_error[3:0]	Output	Indicates a TX phase compensation FIFO overflow or underrun condition.
rx_rlv[3:0]	Output	Asserted if the number of continuous 1s and 0s exceeds the number that was set in the run-length option. The rx_rlv signal is asynchronous to the RX datapath and is asserted for a minimum of 2 recovered clock cycles.
rx_recovered_clk	Output	This is the RX clock which is recovered from the received data stream.

XAUI PHY Register Interface and Register Descriptions

This section describes the register interface and descriptions for the IP core.

The Avalon-MM PHY management interface provides access to the XAUI PHY IP Core PCS, PMA, and transceiver reconfiguration registers.

Table 7-14: Avalon-MM PHY Management Interface

Signal Name	Direction	Description
phy_mgmt_clk	Input	Avalon-MM clock input. There is no frequency restriction for Stratix V devices; however, if you plan to use the same clock for the PHY management interface and transceiver reconfiguration, you must restrict the frequency range of phy_mgmt_clk to 100–150 MHz to meet the specification for the transceiver reconfiguration clock. For Arria II GX, Cyclone IV GX, HardCopy IV, and Stratix IV GX the frequency range is 37.5–50 MHz.
phy_mgmt_clk_reset	Input	Global reset signal that resets the entire XAUI PHY. This signal is active high and level sensitive.
phy_mgmt_addr[8:0]	Input	9-bit Avalon-MM address.
phy_mgmt_writedata[31:0]	Input	32-bit input data.
phy_mgmt_readdata[31:0]	Output	32-bit output data.
phy_mgmt_write	Input	Write signal. Asserted high.
phy_mgmt_read	Input	Read signal. Asserted high.
phy_mgmt_waitrequest	Output	When asserted, indicates that the Avalon-MM slave interface is unable to respond to a read or write request. When asserted, control signals to the Avalon-MM slave interface must remain constant.

For more information about the Avalon-MM interface, including timing diagrams, refer to the *Avalon Interface Specifications*.

The following table specifies the registers that you can access using the Avalon-MM PHY management interface using word addresses and a 32-bit embedded processor. A single address space provides access to all registers.

Note: Writing to reserved or undefined register addresses may have undefined side effects.

Table 7-15: XAUI PHY IP Core Registers

Word Addr	Bits	R/W	Register Name	Description
PMA Common Control and Status Registers				
0x021	[31:0]	RW	`cal_blk_powerdown`	Writing a 1 to channel <*n*> powers down the calibration block for channel <*n*>. This register is not available for Stratix V devices.
0x022	[31:0]	R	`pma_tx_pll_is_locked`	Bit[P] indicates that the TX CMU PLL (P) is locked to the input reference clock. There is typically one `pma_tx_pll_is_locked` bit per system. This register is not available for Arria V, Arria V GZ, Cyclone V, or Stratix V devices.
Reset Control Registers–Automatic Reset Controller				
0x041	[31:0]	RW	`reset_ch_bitmask`	Bit mask for reset registers at addresses 0x042 and 0x044. The default value is all 1s. Channel <*n*> can be reset when bit<*n*> = 1.
0x042	[1:0]	W	`reset_control`(write)	Writing a 1 to bit 0 initiates a TX digital reset using the reset controller module. The reset affects channels enabled in the `reset_ch_bitmask`. Writing a 1 to bit 1 initiates a RX digital reset of channels enabled in the `reset_ch_bitmask`. This bit self-clears.
		R	`reset_status`(read)	Reading bit 0 returns the status of the reset controller TX ready bit. Reading bit 1 returns the status of the reset controller RX ready bit. This bit self-clears.
Reset Controls –Manual Mode				

Word Addr	Bits	R/W	Register Name	Description
0x044	[31:4,0]	RW	Reserved	It is safe to write 0s to reserved bits.
	[1]	RW	`reset_tx_digital`	Writing a 1 causes the internal TX digital reset signal to be asserted, resetting all channels enabled in `reset_ch_bitmask`. You must write a 0 to clear the reset condition.
	[2]	RW	`reset_rx_analog`	Writing a 1 causes the internal RX analog reset signal to be asserted, resetting the RX analog logic of all channels enabled in `reset_ch_bitmask`. You must write a 0 to clear the reset condition.
	[3]	RW	`reset_rx_digital`	Writing a 1 causes the RX digital reset signal to be asserted, resetting the RX digital channels enabled in `reset_ch_bitmask`. You must write a 0 to clear the reset condition.
PMA Control and Status Registers				
0x061	[31:0]	RW	`phy_serial_loopback`	Writing a 1 to channel <*n*> puts channel <*n*> in serial loopback mode. For information about pre- or post-CDR serial loopback modes, refer to Loopback Modes.
0x064	[31:0]	RW	`pma_rx_set_locktodata`	When set, programs the RX CDR PLL to lock to the incoming data. Bit <*n*> corresponds to channel <*n*>.
0x065	[31:0]	RW	`pma_rx_set_locktoref`	When set, programs the RX CDR PLL to lock to the reference clock. Bit <*n*> corresponds to channel <*n*>.
0x066	[31:0]	RO	`pma_rx_is_lockedtodata`	When asserted, indicates that the RX CDR PLL is locked to the RX data, and that the RX CDR has changed from LTR to LTD mode. Bit <*n*> corresponds to channel <*n*>.
0x067	[31:0]	RO	`pma_rx_is_lockedtoref`	When asserted, indicates that the RX CDR PLL is locked to the reference clock. Bit <*n*> corresponds to channel <*n*>.
XAUI PCS				
0x082	[31:4]	-	Reserved	-
	[3:0]	RW	`invpolarity[3:0]`	Inverts the polarity of corresponding bit on the RX interface. Bit 0 maps to lane 0 and so on. This register is only available in the hard XAUI implementation. **To block**: Word aligner.

Word Addr	Bits	R/W	Register Name	Description
0x083	[31:4]	-	Reserved	-
	[3:0]	RW	`invpolarity[3:0]`	Inverts the polarity of corresponding bit on the TX interface. Bit 0 maps to lane 0 and so on. This register is only available in the hard XAUI implementation. **To block**: Serializer.
0x084	[31:16]	-	Reserved	-
	[15:8]	R	`patterndetect[7:0]`	When asserted, indicates that the programmed word alignment pattern has been detected in the current word boundary. The RX pattern detect signal is 2 bits wide per channel or 8 bits per XAUI link. Reading the value of the `patterndetect` registers clears the bits.This register is only available in the hard XAUI implementation. **From block**: Word aligner.
	[7:0]		`syncstatus[7:0]`	Records the synchronization status of the corresponding bit. The RX sync status register has 2 bits per channel for a total of 8 bits per hard XAUI link. The RX sync status register has 1 bit per channel for a total of 4 bits per soft XAUI link; soft XAUI uses bits 0–3. Reading the value of the `syncstatus` register clears the bits. **From block**: Word aligner.

Word Addr	Bits	R/W	Register Name	Description
0x085	[31:16]	-	Reserved	-
	[15:8]	R	errdetect[7:0]	When set, indicates that a received 10-bit code group has an 8B/10B code violation or disparity error. It is used along with disperr to differentiate between a code violation error, a disparity error, or both. There are 2 bits per RX channel for a total of 8 bits per XAUI link. Reading the value of the errdetect register clears the bits. **From block**: 8B/10B decoder.
	[7:0]		disperr[7:0]	Indicates that the received 10-bit code or data group has a disparity error. When set, the corresponding errdetect bits are also set. There are 2 bits per RX channel for a total of 8 bits per XAUI link. Reading the value of the errdetect register clears the bits **From block**: 8B/10B decoder.
0x086	[31:8]	-	Reserved	-
	[7:4]	R, sticky	phase_comp_fifo_error[3:0]	Indicates a RX phase compensation FIFO overflow or underrun condition on the corresponding lane. Reading the value of the phase_comp_fifo_error register clears the bits. This register is only available in the hard XAUI implementation **From block**: RX phase compensation FIFO.
	[3:0]		rlv[3:0]	Indicates a run length violation. Asserted if the number of consecutive 1s or 0s exceeds the number that was set in the **Runlength check** option. Bits 0-3 correspond to lanes 0-3, respectively. Reading the value of the RLV register clears the bits. This register is only available in the hard XAUI implementation. **From block**: Word aligner.

Word Addr	Bits	R/W	Register Name	Description
0x087	[31:16]	-	Reserved	-
	[15:8]	R, sticky	rmfifodatainserted[7:0]	When asserted, indicates that the RX rate match block inserted a ||R|| column. Goes high for one clock cycle per inserted ||R|| column. Reading the value of the rmfifodatainserted register clears the bits. This register is only available in the hard XAUI implementation. **From block**: Rate match FIFO.
	[7:0]		rmfifodatadeleted[7:0]	When asserted, indicates that the rate match block has deleted an ||R|| column. The flag goes high for one clock cycle per deleted ||R|| column. There are 2 bits for each lane. Reading the value of the rmfifodatadeleted register clears the bits. This register is only available in the hard XAUI implementation. **From block**: Rate match FIFO.
0x088	[31:8]	-	Reserved	-
	[7:4]	R, sticky	rmfifofull[3:0]	When asserted, indicates that rate match FIFO is full (20 words). Bits 0-3 correspond to lanes 0-3, respectively. Reading the value of the rmfifofull register clears the bits. This register is only available in the hard XAUI implementation **From block**: Rate match FIFO.
	[3:0]		rmfifoempty[3:0]	When asserted, indicates that the rate match FIFO is empty (5 words). Bits 0-3 correspond to lanes 0-3, respectively. Reading the value of the rmfifoempty register clears the bits. This register is only available in the hard XAUI implementation **From block**: Rate match FIFO.

Word Addr	Bits	R/W	Register Name	Description
0x089	[31:3]	-	Reserved	-
	[2:0]	R, sticky	`phase_comp_fifo_error[2:0]`	Indicates a TX phase compensation FIFO overflow or `underrun` condition on the corresponding lane. Reading the value of the `phase_comp_fifo_error` register clears the bits. This register is only available in the hard XAUI implementation **From block**: TX phase compensation FIFO.
0x08a	[0]	RW	`simulation_flag`	Setting this bit to 1 shortens the duration of reset and loss timer when simulating. Altera recommends that you keep this bit set during simulation.

For more information about the individual PCS blocks, refer to the Transceiver Architecture chapters of the appropriate device handbook.

Related Information

- **Loopback Modes** on page 17-59
- **Avalon Interface Specifications**
- **Transceiver Architecture in Arria II Devices**
- **Transceiver Architecture in Arria V Devices**
- **Cyclone IV Transceivers Architecture**
- **Transceiver Architecture in Cyclone V Devices**
- **Transceiver Architecture in HardCopy IV Devices**
- **Transceiver Architecture in Stratix IV Devices**
- **Transceiver Architecture in Stratix V Devices**

XAUI PHY Dynamic Reconfiguration for Arria II GX, Cyclone IV GX, HardCopy IV GX, and Stratix IV GX

The Arria II GX, Cyclone IV GX, HardCopy IV GX, and Stratix IV GX use the ALTGX_RECONFIG Mega function for transceiver reconfiguration.

For more information about the ALTGX_RECONFIG Megafunction, refer to *ALTGX_RECONFIG Megafunction User Guide for Stratix IV Devices* in volume 2 of the *Stratix IV Device Handbook*.

If your XAUI PHY IP Core includes a single transceiver quad, these signals are internal to the core. If your design uses more than one quad, the reconfiguration signals are external.

Table 7-16: Dynamic Reconfiguration Interface Arria II GX, Cyclone IV GX, HardCopy IV GX, and Stratix IV GX devices

Signal Name	Direction	Description
`reconfig_to_xcvr[3:0]`	Input	Reconfiguration signals from the Transceiver Reconfiguration IP Core to the XAUI transceiver.
`reconfig_from_xcvr[<n>:0]`	Output	Reconfiguration signals from the XAUI transceiver to the Transceiver Reconfiguration IP Core. The size of this bus is depends on the device. For the soft PCS in Stratix IV GX and GT devices, <*n*> = 68 bits. For hard XAUI variants, <*n*> = 16. For Stratix V devices, the number of bits depends on the number of channels specified. Refer to Chapter 16, Transceiver Reconfiguration Controller IP Core for more information.

Related Information

- **Transceiver Reconfiguration Controller IP Core Overview** on page 17-1
- **ALTGX_RECONFIG Megafunction User Guide for Stratix IV Devices**

XAUI PHY Dynamic Reconfiguration for Arria V, Arria V GZ, Cyclone V and Stratix V Devices

The Arria V, Arria V GZ, Cyclone V, and Stratix V devices use the Transceiver Reconfiguration Controller IP Core for dynamic reconfiguration.

For more information about this IP core, refer to Chapter 16, Transceiver Reconfiguration Controller IP Core.

Each channel and each TX PLL have separate dynamic reconfiguration interfaces. The MegaWizard Plug-In Manager provides informational messages on the connectivity of these interfaces.

Example 7-2: Informational Messages for the Transceiver Reconfiguration Interface

```
PHY IP will require 8 reconfiguration interfaces for connection to the
external reconfiguration controller.Reconfiguration interface offsets 0-3
are connected to the transceiver channels.Reconfiguration interface offsets
4-7 are connected to the transmit PLLs.
```

Although you must initially create a separate reconfiguration interface for each channel and TX PLL in your design, when the Intel Quartus Prime software compiles your design, it reduces the number of reconfiguration interfaces by merging reconfiguration interfaces. The synthesized design typically includes a reconfiguration interface for at least three channels because three channels share an Avalon-MM slave interface which connects to the Transceiver Reconfiguration Controller IP Core. Conversely, you cannot connect the three channels that share an Avalon-MM interface to different Transceiver Reconfiguration Controller IP Cores. Doing so causes a Fitter error. For more information, refer to "Transceiver Reconfiguration Controller to PHY IP Connectivity".

Related Information

- **Transceiver Reconfiguration Controller to PHY IP Connectivity** on page 17-57
- **Transceiver Reconfiguration Controller IP Core Overview** on page 17-1

Logical Lane Assignment Restriction

If you are using ×6 or ×N bonding, transceiver dynamic reconfiguration requires that you assign the starting channel number.

Logical channel 0 should be assigned to either physical transceiver channel 1 or channel 4 of a transceiver bank. However, if you have already created a PCB with a different lane assignment for logical lane 0, you can use the workaround shown in the following example to remove this restriction. This redefines the `pma_bonding_master` parameter using the Intel Quartus Prime Assignment Editor. In this example, the `pma_bonding_master` was originally assigned to physical channel 1. (The original assignment could also have been to physical channel 4.) The to parameter reassigns the `pma_bonding_master` to the PHY IP instance name. You must substitute the instance name from your design for the instance name shown in quotation marks

Example 7-3: Overriding Logical Channel 0 Channel Assignment Restrictions in Stratix V Devices for ×6 or ×N Bonding

```
set_parameter -name pma_bonding_master "\"1\"" -to "<PHY IP instance name>"
```

Related Information

- **Transceiver Reconfiguration Controller to PHY IP Connectivity** on page 17-57
- **Transceiver Reconfiguration Controller IP Core Overview** on page 17-1

XAUI PHY Dynamic Reconfiguration Interface Signals

This section describes the signals in the reconfiguration interface. This interface uses the Avalon-MM PHY Management interface clock.

Table 7-17: Reconfiguration Interface

Signal Name	Direction	Description
`reconfig_to_xcvr [(<n>70)-1:0]`	Input	Reconfiguration signals from the Transceiver Reconfiguration Controller. <*n*> grows linearly with the number of reconfiguration interfaces. <*n*> initially includes the total number transceiver channels and TX PLLs before optimization/merging.
`reconfig_from_xcvr [(<n>46)-1:0]`	Output	Reconfiguration signals to the Transceiver Reconfiguration Controller. <*n*> grows linearly with the number of reconfiguration interfaces. <*n*> initially includes the total number transceiver channels before optimization/merging.

Related Information

- **Transceiver Reconfiguration Controller to PHY IP Connectivity** on page 17-57
- **Transceiver Reconfiguration Controller IP Core Overview** on page 17-1

SDC Timing Constraints

The SDC timing constraints and approaches to identify false paths listed for Stratix V Native PHY IP apply to all other transceiver PHYs listed in this user guide. Refer to *SDC Timing Constraints of Stratix V Native PHY* for details.

Related Information

SDC Timing Constraints of Stratix V Native PHY on page 13-72
This section describes SDC examples and approaches to identify false timing paths.

Simulation Files and Example Testbench

Refer to "Running a Simulation Testbench" for a description of the directories and files that the Intel Quartus Prime software creates automatically when you generate your XAUI PHY IP Core.

Refer to the *Intel FPGA Wiki* for an example testbench that you can use as a starting point in creating your own verification environment.

Related Information

- **Running a Simulation Testbench** on page 1-6
- **Intel FPGA Wiki**

Interlaken PHY IP Core 8

2020.06.02

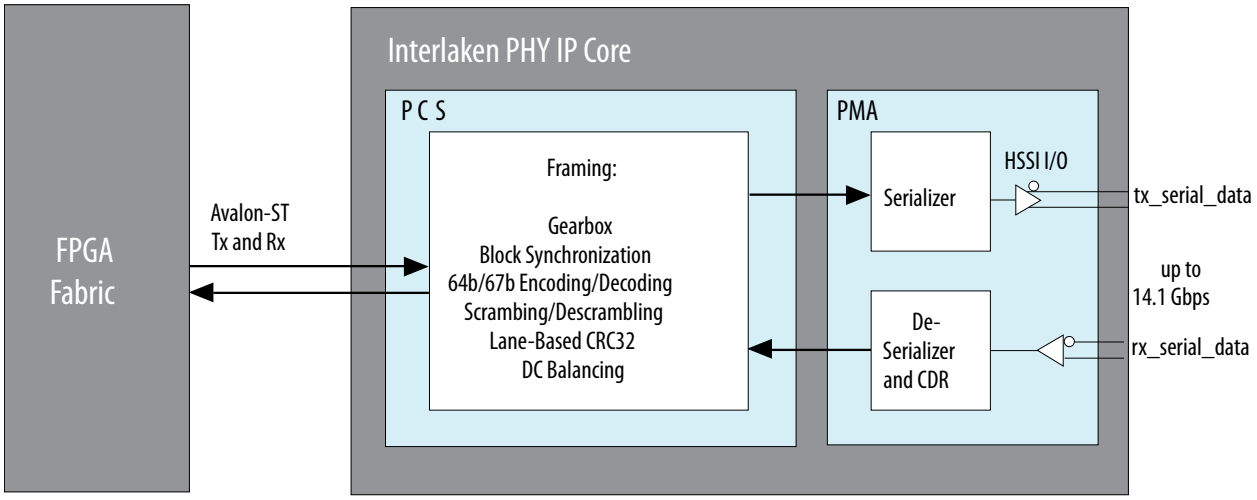
UG-01080

Subscribe Send Feedback

The Altera Interlaken PHY IP Core implements *Interlaken Protocol Specification, Rev 1.2*.

Interlaken is a high speed serial communication protocol for chip-to-chip packet transfers. It supports multiple instances, each with 1 to 24 lanes running at 10.3125 Gbps or greater in Arria V GZ and Stratix V devices. The key advantages of Interlaken are scalability and its low I/O count compared to earlier protocols such as SPI 4.2. Other key features include flow control, low overhead framing, and extensive integrity checking. The Interlaken physical coding sublayer (PCS) transmits and receives Avalon-ST data on its FPGA fabric interface. It transmits and receives high speed differential serial data using the PCML I/O standard.

Figure 8-1: Interlaken PHY IP Core



For more information about the Avalon-ST protocol, including timing diagrams, refer to the *Avalon Interface Specifications*.

Interlaken operates on 64-bit data words and 3 control bits, which are striped round robin across the lanes to reduce latency. Striping renders the interface independent of exact lane count. The protocol accepts packets on 256 logical channels and is expandable to accommodate up to 65,536 logical channels. Packets are split into small bursts which can optionally be interleaved. The burst semantics include integrity checking and per channel flow control.

The Interlaken PCS supports the following framing functions on a per lane basis:

- Gearbox
- Block synchronization
- Metaframe generation and synchronizatio
- 64b/67b encoding and decoding
- Scrambling and descrambling
- Lane-based CRC32
- Disparity DC balancing

For more detailed information about the Interlaken transceiver channel datapath, clocking, and channel placement in Stratix V devices, refer to the "*Interlaken*" section in the *Transceiver Configurations in Stratix V Devices* chapter of the *Stratix V Device Handbook*.

For more detailed information about the Interlaken transceiver channel datapath, clocking, and channel placement in Arria V GZ devices, refer to the "*Interlaken*" section in the *Transceiver Configurations in Arria V Devices* chapter of the *Arria V Device Handbook*.

Refer to *PHY IP Design Flow with Interlaken for Stratix V Devices* for a reference design that implements the Interlaken protocol in a Stratix V device.

Related Information

- **Interlaken Protocol Specification, Rev 1.2**
- **Avalon Interface Specifications**
- **Transceiver Configurations in Stratix V Devices**
- **Transceiver Configurations in Arria V Devices**
- **PHY IP Design Flow with Interlaken for Stratix V Devices**

Interlaken PHY Device Family Support

This section describes the Interlaken PHY device family support.

IP cores provide either final or preliminary support for target Altera device families. These terms have the following definitions:

- *Final support*—Verified with final timing models for this device.
- *Preliminary support*—Verified with preliminary timing models for this device.

Table 8-1: Device Family Support

Device Family	Support
Arria V GZ devices–Hard PCS + PMA	Final
Stratix V devices–Hard PCS + PMA	Final
Other device families	Not supported

Parameterizing the Interlaken PHY

The Interlaken PHY IP Core is available when you select the **Arria V GZ** or **Stratix V** devices. Complete the following steps to configure the Interlaken PHY IP Core in the MegaWizard Plug-In Manager:

1. Under **Tools** > **IP Catalog**, select the device family of your choice.
2. Under **Tools** > **Interface Protocols** > **Interlaken**, select **Interlaken PHY**.
3. Use the tabs on the MegaWizard Plug-In Manager to select the options required for the protocol.
4. Refer to the following topics to learn more about the parameters:
   a. General Parameters
   b. Optional Port Parameters
   c. Analog Options
5. Click **Finish** to generate your parameterized Interlaken PHY IP Core.

Interlaken PHY General Parameters

This section describes the Interlaken PHY parameters you can set on the **General** tab.

Table 8-2: Interlaken PHY General Options

Parameter	Value	Description
Device family	**Arria V GZ** **Stratix V**	Specifies the device family.
Datapath mode	**Duplex,** **RX,** **TX**	Specifies the mode of operation as **Duplex**, **RX**, or **TX** mode.
Lane rate	**3125 Mbps** **5000 Mbps** **6250 Mbps** **6375 Mbps** **10312.5 Mbps** **10312.5 Mbps** **12500 Mbps** **Custom**	Specifies the lane data rate. The **Input clock frequency** and **Base data rate** parameters update automatically based on the **Lane rate** you specify. Custom, user-defined, lane data rates are now supported. However, the you must choose a lane data rate that results in a standard board oscillator reference clock frequency to drive the `pll_ref_clk` and meet jitter requirements. Choosing a lane data rate that deviates from standard reference clock frequencies may result in custom board oscillator clock frequencies which may be prohibitively expensive or unavailable.
Number of lanes	`1-24`	Specifies the number of lanes in a link over which data is striped.

Parameter	Value	Description
Metaframe length in words	5-8191	Specifies the number of words in a metaframe. The default value is 2048. Although 5 -8191 words are valid metaframe length values, the current Interlaken PHY IP Core implementation requires a minimum of 128 **Metaframe length** for good, stable performance. In simulation, Altera recommends that you use a smaller metaframe length to reduce simulation times.
Input clock frequency	**Lane rate**/<n> **Lane rate**/80 **Lane rate**/64 **Lane rate**/50 **Lane rate**/40 **Lane rate**/32 **Lane rate**/25 **Lane rate**/20 **Lane rate**/16 **Lane rate**/12.5 **Lane rate**/10 **Lane rate**/8	Specifies the frequency of the input reference clock. The default value for the **Input clock frequency** is the **Lane rate** /20. Many reference clock frequencies are available.
PLL type	CMU ATX	Specifies the PLL type. The CMU PLL has a larger frequency range than the ATX PLL. The ATX PLL is designed to improve jitter performance and achieves lower channel-to-channel skew; however, it supports a narrower range of lane data rates and reference clock frequencies. Another advantage of the ATX PLL is that it does not use a transceiver channel, while the CMU PLL does. Because the CMU PLL is more versatile, it is specified as the default setting.

Parameter	Value	Description
Base data rate	1 × **Lane rate** 2 × **Lane rate** 3 × **Lane rate**	This option allows you to specify a **Base data rate** to minimize the number of PLLs required to generate the clocks necessary for data transmission at different frequencies. Depending on the Lane rate you specify, the default **Base data rate** can be either 1, 2, or 4 times the **Lane rate**; however, you can change this value. The default value specified is for backwards compatibility with earlier Intel Quartus Prime software releases.

Interlaken PHY Optional Port Parameters

This section describes the Interlaken PHY optional port parameters you can set on the **Optional Ports** tab.

Table 8-3: Optional Ports

Parameter	Value	Description
Enable RX status signals, (word lock, sync lock, crc32 error) as part of rx_parallel_data	**On/Off**	When you turn this option on, `rx_parallel_data[71:69]` are included in the top-level module. These optional signals report the status of word and synchronization locks and CRC32 errors. Refer to Avalon-ST RX Signals for more information.
Create tx_coreclkin port	**On/Off**	The `tx_coreclkin` drives the write side of TX FIFO. This clock is required for multi-lane synchronization but is optional for single lane Interlaken links. If `tx_coreclkin` is deselected for single lane Interlaken links, `tx_user_clkout` drives the TX side of the write FIFO. You must use the `tx_user_clkout` output port to drive transmit data in the Interlaken MAC.
Create rx_coreclkin port	**On/Off**	When selected `rx_coreclkin` is available as input port which drives the read side of RX FIFO, When deselected `rx_user_clkout`, `rx_clkout` for all bonded receiver lanes, is routed internally to drive the RX read side of FIFO. `rx_user_clkout` is also available as an output port for the Interlaken MAC.

Interlaken PHY Analog Parameters

This section describes the Interlaken PHY analog parameters.

Click on the appropriate link to specify the analog options for your device:

Related Information

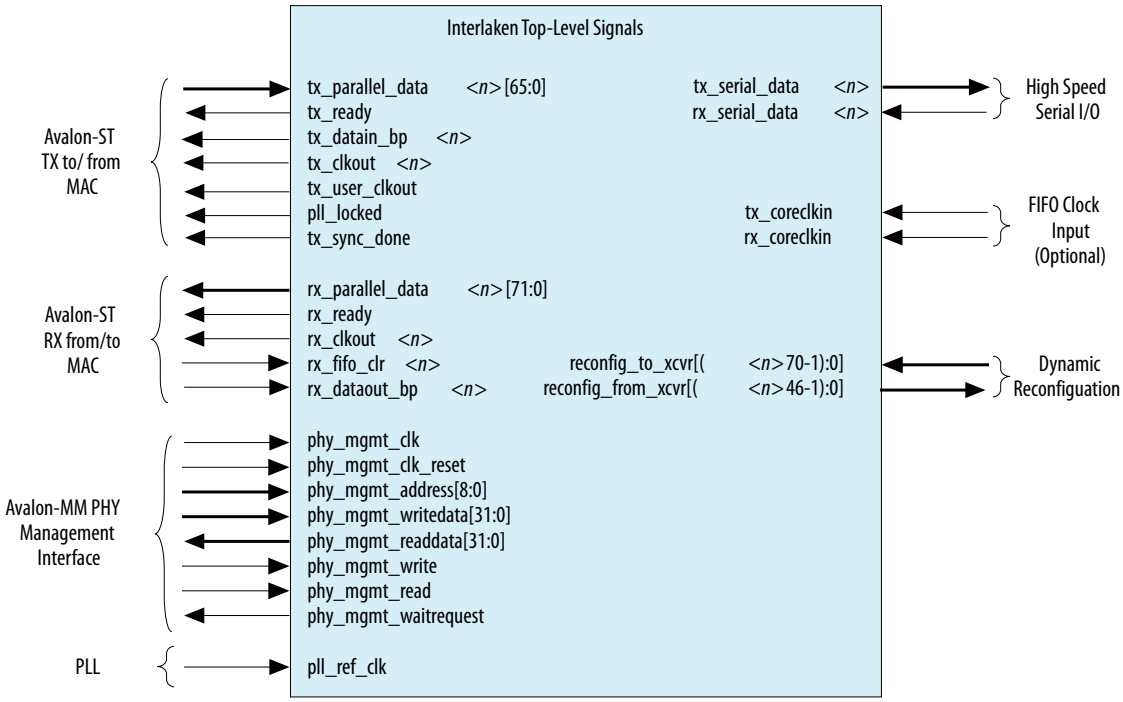
- **Analog Settings for Arria V GZ Devices** on page 20-11
- **Analog Settings for Stratix V Devices** on page 20-35

Interlaken PHY Interfaces

This section describes the Interlaken PHY interfaces.

The following figure illustrates the top-level signals of the Interlaken PHY IP Core; <*n*> is the channel number so that the width of `tx_data` in 4-lane instantiation is [263:0].

Figure 8-2: Top-Level Interlaken PHY Signals



Note: The **block diagram** shown in the GUI labels the external pins with the *interface type* and places the *interface name* inside the box. The interface type and name are used to define interfaces in the **_hw.tcl**. writing.

For more information about **_hw.tcl**, files refer to the *Component Interface Tcl Reference* chapter in volume 1 of the Intel Quartus Prime Handbook.

Related Information

Component Interface Tcl Reference

Interlaken PHY Avalon-ST TX Interface

This section lists the signals in the Avalon-ST TX interface.

Table 8-4: Avalon-ST TX Signals

Signal Name	Direction	Description
tx_parallel_data<n>[63:0]	Input	Avalon-ST data bus driven from the FPGA fabric to the TX PCS. This input should be synchronized to the tx_coreclkin clock domain.
tx_parallel_data<n>[64]	Input	Indicates whether tx_parallel_data<n>[63:0] represents control or data. When deasserted, tx_parallel_data<n>[63:0] is a data word. When asserted, tx_parallel_data<n>[63:0] is a control word. The value of header synchronization bits[65:64] of the Interlaken word identify whether bits[63:0] are a Framing Layer Control/Burst/IDLE Control Word or a data word. The MAC must gray encode the header synchronization bits. The value 2'b10 indicating Burst/IDLE Control Word must be gray encoded to the value 1'b1 for tx_parallel_data<n>[64]. The value 2'b01 indicating data word must be gray encoded to the value 1'b0 for tx_parallel_data<n>[64]. You can also tie header synchronization bit[65] to tx_parallel_data[64] directly.

Signal Name	Direction	Description
tx_parallel_data<n>[65]	Input	When asserted, indicates that tx_parallel_data<n>[63:0] is valid and is ready to be written into the TX FIFO. When deasserted, indicates that tx_parallel_data<n>[63:0] is invalid and is not written into the TX FIFO. This signal is the data valid or write enable port of the TX FIFO. This input must be synchronized to the tx_coreclkin clock domain. The Interlaken MAC should gate tx_parallel_data<n>[65] based on tx_datain_bp<*n*>. Or, you can tie tx_datain_bp<*n*> directly to tx_parallel_data<*n*>[65]. For Quartus II releases before 12.0, you must pre-fill the transmit FIFO so this pin must be 1'b1 when tx_ready is asserted, but before tx_sync_done is asserted to insert the pre-fill pattern. Do not use valid data to pre-fill the transmit FIFO. Use the following Verilog HDL assignment for Quartus II releases prior to 12.0: `assign tx_parallel_data[65] = (!tx_sync_done)?1'b1:tx_datain_bp[0];`
tx_ready	Output	When asserted, indicates that the TX interface has exited the reset state and is ready for service. The tx_ready latency for the TX interface is 0. A 0 latency means that the TX FIFO can accept data on the same clock cycle that tx_ready is asserted. This output is synchronous to the phy_mgmt_clk clock domain. The Interlaken MAC must wait for tx_ready before initiating data transfer (pre-fill pattern or valid user data) on any lanes. The TX FIFO only captures input data from the Interlaken MAC when tx_ready and tx_parallel_data[65] are both asserted. The beginning of the pre-fill stage is marked by the assertion of tx_ready, before tx_sync_done is asserted. The pre-fill stage should terminate when tx_ready is high and tx_sync_done changes from Logic 0 to Logic 1 state. At this point, TX synchronization is complete and valid TX data insertion can begin. TX synchronization is not required for single-lane variants. Use the following Verilog HDL assignment is for Quartus versions earlier than 12.0: `assign tx_parallel_data[65] = (!tx_sync_done)?1'b1:tx_datain_bp[0];`
tx_datain_bp<n>	Output	When asserted, indicates that Interlaken TX lane <*n*> interface is ready to receive data for transmission. In

Signal Name	Direction	Description
		multi-lane configurations, the tx_datain_bp<*n*> signals must be logically Ored. The latency on this Avalon-ST interface is 0 cycles. The Interlaken MAC must only drive valid user data on tx_parallel_data<*n*>[64] and tx_parallel_data<*n*>[63:0] data bus as soon as tx_ready<*n*> and tx_sync_done are both asserted. The tx_datain_bp<*n*> signal is connected to the partial empty threshold of the TX FIFO, so that when tx_datain_bp<*n*> is deasserted the TX FIFO back pressures the Interlaken MAC. Stop sending TX data to the PHY when this signal is deasserted. The Interlaken MAC can continue driving data to the TX FIFO when tx_datain_bp<*n*> is asserted. The Interlaken MAC should gate tx_parallel_data<*n*>[65], which operates as a data_valid signal, based on tx_datain_bp<*n*> . This output is synchronous to the tx_coreclkin clock domain. Or, you can also tie tx_datain_bp<*n*> directly to tx_parallel_data<*n*>[65] . For Quartus II releases prior to 12.0, you must pre-fill the TX FIFO before tx_sync_done can be asserted. Do not use valid data to pre-fill the TX FIFO. Use the following Verilog HDL assignment for Quartus II releases prior to 12.0: `assign tx_parallel_data[65] = (!tx_sync_done)?1'b1:tx_datain_bp[0];`
tx_clkout	Output	For single lane Interlaken links, tx_user_clkout is available when you do not create the optional tx_coreclkin . For Interlaken links with more than 1 lane, tx_coreclkin is required and tx_user_clkout cannot be used. tx_coreclkin must have a minimum frequency of the lane data rate divided by 67. The frequency range for tx_coreclkin is (data rate/40) - (data rate/67). For best results, Altera recommends that tx_coreclkin = (data rate/40).
tx_user_clkout	Output	For single lane Interlaken links, tx_user_clkout is available when you do not create the optional tx_coreclkin. For Interlaken links with more than 1 lane, tx_coreclkin is required and tx_user_clkout cannot be used. You can use a minimum frequency of lane datarate divided by 67 for tx_coreclkin, although Altera recommends that tx_coreclkin frequency of the lane data rate divided by 40 for best performance.

Signal Name	Direction	Description
pll_locked	Output	In multilane Interlaken designs, this signal is the bitwise AND of the individual lane pll_locked signals. This output is synchronous to the phy_mgmt_clk clock domain.
tx_sync_done	Output	When asserted, indicates that all tx_parallel_data lanes are synchronized and ready for valid user data traffic. The Interlaken MAC must wait for this signal to be asserted before initiating valid user data transfers on any lane. This output is synchronous to the tx_coreclkin clock domain. For consistent tx_sync_done performance, Altera recommends using tx_coreclkin and rx_coreclkin frequency of lane (data rate/40). You must invoke a hard reset using mgmt_rst_reset and phy_mgmt_clk_reset to initiate the synchronization sequence on the TX lanes. After tx_sync_done is asserted, you must never allow the TX FIFO to underflow, doing so requires you to hard reset to the Interlaken PHY IP Core. For Quartus versions prior to 12.0, you must pre-fill the TX FIFO before tx_sync_done can be asserted. Use the following Verilog HDL assignment for Quartus II releases prior to 12.0: `assign tx_parallel_data[65] = (!tx_sync_done)?1'b1:tx_datain_bp[0];`

Interlaken PHY Avalon-ST RX Interface

This section lists the signals in the Avalon-ST RX interface.

Table 8-5: Avalon-ST RX Signals

Signal Name	Direction	Description
rx_parallel_data<*n*>[63:0]	Output	Avalon-ST data bus driven from the RX PCS to the FPGA fabric. This output is synchronous to the rx_coreclkin clock domain.

Signal Name	Direction	Description
rx_parallel_data<*n*>[64]	Output	When asserted, indicates that rx_parallel_data<*n*>[63:0] is valid. When deasserted, indicates the rx_parallel_data<*n*>[63:0] is invalid. This output is synchronous to the rx_coreclkin clock domain. The Interlaken PCS implements a gearbox between the PMA and PCS interface. The rx_parallel_data<*n*>[64] port is deasserted whenever the gearbox is in the invalid region. The Interlaken MAC should not read rx_parallel_data<*n*>[65, 63:0] if rx_parallel_data<*n*>[64] is deasserted.
rx_parallel_data<*n*>[65]	Output	Indicates whether rx_parallel_data<*n*>[63:0] represents control or data. When deasserted, rx_parallel_data<*n*>[63:0] is a data word. When asserted, rx_parallelddata<*n*>[63:0] is a control word. This output is synchronous to the rx_coreclkin clock domain. The value of header synchronization bits[65:64] of the Interlaken word identify whether bits[63:0] are Framing Layer Control/Burst/IDLE Word or a data word. The value 2'b10 indicating a Framing Layer Control/Burst/IDLE Word is gray encoded to the value 1'b1 and rx_parallel_data<*n*>[65] is asserted by the Interlaken Receive PCS. The value 2'b01 indicating data word is gray encoded to the value 1'b0 and rx_parallel_data<*n*>[65] is deasserted by the Interlaken Receive PCS. The Framing Layer Control Words (Frame Sync, Scrambler State, Skip, and Diag) are not discarded but are modified (scrambler seed cleared to 0) and sent to the Interlaken MAC for multi-lane alignment and deskew on the lanes.
rx_parallel_data<*n*>[66]	Output	This is an active-high synchronous status signal indicating that block lock (frame synchronization) and frame lock (metaframe boundary delineation) have been achieved. The Interlaken MAC must use this signal to indicate that Metaframe synchronization has been achieved for this lane. You must use this rx_parallel_data[66] as the primary frame synchronization status flag and only use the optional rx_parallel_data[70] as the secondary frame synchronization status flag. This output is synchronous to the rx_coreclkin clock domain. If the RX PCS FIFO reaches the empty state or is in an empty state, rx_parallel_data<*n*>[66] Block Lock and Frame Lock status signals are deasserted in the next clock cycle. rx_parallel_data<*n*>[70] indicating metaframe lock and rx_parallel_data<*n*>[69] indicating that the first Interlaken synchronization word alignment pattern has been received remain asserted.

Signal Name	Direction	Description
rx_parallel_data<*n*>[67]	Output	When asserted, indicates an RX FIFO overflow error.
rx_parallel_data<*n*>[68]	Output	When asserted, indicates that the RX FIFO is partially empty and is still accepting data from the frame synchronizer. This signal is asserted when the RX FIFO fill level is below the rx_fifo_pempty threshold. This output is synchronous to the rx_coreclkin clock domain. To prevent underflow, the Interlaken MAC should begin reading from the RX FIFO when this signal is deasserted, indicating sufficient FIFO contents (RX FIFO level above rx_fifo_pempty threshold). The MAC should continue to read the RX FIFO to prevent overflow as long as this signal is not reasserted. You can assert a FIFO flush using the rx_fifo_clr<*n*> when the receive FIFO overflows. This output is synchronous to the rx_clkout clock domain. Therefore, you must synchronize rx_parallel_data<*n*>[68] to the rx_coreclkin before making the assignment below. You can tie this signal's inverted logic to the rx_dataout_bp<*n*> receive FIFO read enable signal as the following assignment statement illustrates: `assign rx_dataout_bp[0] =!(rx_parallel_data[68]);`
rx_parallel_data<*n*>[69]	Output	When asserted, indicates that the RX FIFO has found the first Interlaken synchronization word alignment pattern. For very short metaframes, this signal may be asserted after the frame synchronizer state machine validates frame synchronization and asserts rx_parallel_data<*n*>[70] because this signal is asserted by the RX FIFO which is the last PCS block in the RX datapath. This output is synchronous to the rx_coreclkin clock domain. This signal is optional. If the RX PCS FIFO reaches the empty state or is in an empty state, rx_parallel_data<*n*>[70] indicating metaframe lock and rx_parallel_data<*n*>[69] indicating that the first Interlaken synchronization word alignment pattern has been received remain asserted, but rx_parallel_data<*n*>[66] block lock and frame lock status signal are deasserted in the next clock cycle.

Signal Name	Direction	Description
rx_parallel_data<*n*>[70]	Output	When asserted, indicates that the RX frame synchronization state machine has found and received 4 consecutive, valid synchronization words. The frame synchronization state machine requires 4 consecutive synchronization words to exit the presync state and enter the synchronized state. You should only use this optional signal as a secondary status flag. The rx_parallel_data[66] signal should be used as the primary frame synchronization status flag. This output is synchronous to the rx_clkout clock domain. This signal is optional. If the RX PCS FIFO reaches an empty state or is in an empty state, rx_parallel_data<*n*>[70] indicating metaframe lock and rx_parallel_data<*n*>[69] indicating that the first Interlaken synchronization word alignment pattern has been received remain asserted but rx_parallel_data<*n*>[66] block lock and frame lock status signal are deasserted in the next clock cycle.
rx_parallel_data<*n*>[71]	Output	When asserted, indicates a CRC32 error in this lane. This signal is optional. This output is synchronous to the rx_clkout clock domain.
rx_ready	Output	When asserted, indicates that the RX interface has exited the reset state and is ready for service. The Interlaken MAC must wait for rx_ready to be asserted before initiating data transfer on any lanes. This output is synchronous to the phy_mgmt_clk domain.
rx_clkout	Output	Output clock from the RX PCS. The frequency of this clock equals the **Lane rate** divided by 40, which is the PMA serialization factor.
rx_fifo_clr<*n*>	Input	When asserted, the RX FIFO is flushed. This signal allows you to clear the FIFO if the receive FIFO overflows or if the Interlaken MAC is not able to achieve multi-lane alignment in the Interlaken MAC's deskew state machine. The rx_fifo_clr signal must be asserted for 4 rx_clkout cycles to successfully flush the RX FIFO. This output is synchronous to the rx_clkout clock domain.

Signal Name	Direction	Description
rx_dataout_bp<*n*>	Input	When asserted, enables reading of data from the RX FIFO. This signal functions as a read enable. The RX interface has a ready latency of 1 cycle so that rx_parralleldata<*n*>[63:0] and rx_parralleldata<*n*>[65] are valid the cycle after rx_dataout_bp<*n*> is asserted. In multi-lane configurations, the rx_dataout_bp<*n*> port signals must not be logically tied together. This output is synchronous to the rx_coreclkin clock domain. You can tie this rx_dataout_bp<*n*> RX FIFO read enable signal to the inverted logic of the rx_parallel_data[68] RX FIFO partially empty signal using the following assignment statement: `assign rx_dataout_bp[0] =! (rx_parallel_data[68]);`
rx_user_clkout	Output	Master channel rx_user_clkout is available when you do not create the optional rx_coreclkin.

Interlaken PHY TX and RX Serial Interface

This section describes the signals in the chip-to-chip serial interface.

Table 8-6: Serial Interface

Signal Name	Direction	Description
tx_serial_data	Output	Differential high speed serial output data using the PCML I/O standard. Clock is embedded in the serial data stream.
rx_serial_data	Input	Differential high speed serial input data using the PCML I/O standard. Clock is recovered from the serial data stream.

Interlaken PHY PLL Interface

This section describes the signals in the PLL interface.

Table 8-7: PLL Interface

Signal Name	Direction	Description
pll_ref_clk	Input	Reference clock for the PHY PLLs. Refer to the **Lane rate** entry in the Table 8-2table for required frequencies. Custom, user-defined, data rates are now supported. However, the you must choose a lane data rate that results in standard board oscillator reference clock frequency to drive the pll_ref_clk and meet jitter requirements. Choosing a lane data rate that deviates from standard reference clock frequencies may result in custom board oscillator clock frequencies which could be unavailable or cost prohibitive.

Interlaken Optional Clocks for Deskew

This section describes the optional clocks that you can create to reduce clock skew.

Table 8-8: Deskew Clocks

Signal Name	Direction	Description
tx_coreclkin	Input	When enabled tx_coreclkin is available as input port which drives the write side of TX FIFO. Altera recommends using this clock to reduce clock skew. The minimum frequency is data rate/67. Using a lower frequency will underflow the TX FIFO causing the Frame Generators to go into a unrecoverable out of alignment state and insert Skip Words into the lane. If the Interlaken TX FIFO underflows, the alignment state machine tries to recover continuously. When disabled, tx_clkout drives the write side the TX FIFO. tx_coreclkin must be used when the number of lanes is greater than 1.
rx_coreclkin	Input	When enabled, rx_coreclkin is available as input port which drives the read side of RX FIFO. Altera recommends using this clock to reduce clock skew. You should use a minimum frequency of lane data rate/ 67 to drive rx_coreclkin. Using a lower frequency overflows the RX FIFO corrupting the received data.When disabled, rx_user_clkout, which is the master rx_clkout for all the bonded receiver lanes, is internally routed to drive the read side the RX FIFO.

Interlaken PHY Register Interface and Register Descriptions

This section describes the register interface and register descriptions.

The Avalon-MM PHY management interface provides access to the Interlaken PCS and PMA registers, resets, error handling, and serial loopback controls. You can use an embedded controller acting as an Avalon-MM master to send read and write commands to this Avalon-MM slave interface.

Table 8-9: Avalon-MM PCS Management Interface

Signal Name	Direction	Description
phy_mgmt_clk	Input	Avalon-MM clock input. There is no frequency restriction for Stratix V devices; however, if you plan to use the same clock for the PHY management interface and transceiver reconfiguration, you must restrict the frequency range of phy_mgmt_clk to 100–150 MHz to meet the specification for the transceiver reconfiguration clock.
phy_mgmt_clk_reset	Input	Global reset signal that resets the entire Interlaken PHY. This signal is active high and level sensitive. When the Interlaken PHY IP connects to the Transceiver PHY Reconfiguration Controller IP Core, the Transceiver PHY Reconfiguration Controller mgmt_rst_reset signal must be simultaneously asserted with the phy_mgmt_clk_reset signal to bring the Frame Generators in the link into alignment. This is a mandatory requirement. Failure to comply to this requirement will result in excessive transmit lane-to-lane skew in the Interlaken link.
phy_mgmt_addr[8:0]	Input	9-bit Avalon-MM address.
phy_mgmt_writedata[31:0]	Input	Input data.
phy_mgmt_readdata[31:0]	Output	Output data.
phy_mgmt_write	Input	Write signal.
phy_mgmt_read	Input	Read signal.
phy_mgmt_waitrequest	Output	When asserted, indicates that the Avalon-MM slave interface is unable to respond to a read or write request. When asserted, control signals to the Avalon-MM slave interface must remain constant.

The following table specifies the registers that you can access using the Avalon-MM PHY management interface using word addresses and a 32-bit embedded processor. A single address space provides access to all registers. Writing to reserved or undefined register addresses may have undefined side effects.

Note: All undefined register bits are reserved.

Table 8-10: Interlaken PHY Registers

Word Addr	Bits	R/W	Register Name	Description
PMA Common Control and Status Registers				
0x022	[<p>-1:0]	RO	`pma_tx_pll_is_locked`	If <*p*> is the PLL number, Bit[<*p*>] indicates that the TX CMU PLL (<*p*>) is locked to the input reference clock. There is typically one `pma_tx_pll_is_locked` bit per system.
Reset Control Registers-Automatic Reset Controller				
0x041	[31:0]	RW	`reset_ch_bitmask`	Reset controller channel bitmask for digital resets. The default value is all 1s. Channel <n> can be reset when bit<*n*> = 1. Channel <*n*> cannot be reset when bit<*n*> = 0. The Interlaken PHY IP requires the use of the embedded reset controller to initiate the correct the reset sequence. A hard reset to `phy_mgmt_clk_reset` and `mgmt_rst_reset` is required for Interlaken PHY IP. Intel does not recommend use of a soft reset or the use of these reset register bits for Interlaken PHY IP.
0x042	[1:0]	WO	`reset_control` (write)	Writing a 1 to bit 0 initiates a TX digital reset using the reset controller module. The reset affects channels enabled in the `reset_ch_bitmask`. Writing a 1 to bit 1 initiates a RX digital reset of channels enabled in the `reset_ch_bitmask`.
		RO	`reset_status`(read)	Reading bit 0 returns the status of the reset controller TX ready bit. Reading bit 1 returns the status of the reset controller RX ready bit.
Reset Controls -Manual Mode				
0x044	-	RW	`reset_fine_control`	You can use the `reset_fine_control` register to create your own reset sequence. The reset control module, illustrated in Transceiver PHY Top-Level Modules, performs a standard reset

Word Addr	Bits	R/W	Register Name	Description
				sequence at power on and whenever the phy_mgmt_clk_reset is asserted. Bits [31:4, 0] are reserved. The Interlaken PHY IP requires the use of the embedded reset controller to initiate the correct the reset sequence. A hard reset to phy_mgmt_clk_reset and mgmt_rst_reset is required for Interlaken PHY IP. Intel does not recommend use of a soft reset or the use of these reset register bits for Interlaken PHY IP.
	[3]	RW	reset_rx_digital	Writing a 1 causes the RX digital reset signal to be asserted, resetting the RX digital channels enabled in reset_ch_bitmask. You must write a 0 to clear the reset condition.
	[2]	RW	reset_rx_analog	Writing a 1 causes the internal RX digital reset signal to be asserted, resetting the RX analog logic of all channels enabled in reset_ch_bitmask. You must write a 0 to clear the reset condition.
	[1]	RW	reset_tx_digital	Writing a 1 causes the internal TX digital reset signal to be asserted, resetting all channels enabled in reset_ch_bitmask. You must write a 0 to clear the reset condition.
PMA Control and Status Registers				
0x061	[31:0]	RW	phy_serial_loopback	Writing a 1 to channel <*n*> puts channel <*n*> in tx to rx serial loopback mode. For information about pre- or post-CDR rx to tx serial loopback modes, refer to Loopback Modes.
0x064	[31:0]	RW	pma_rx_set_locktodata	When set, programs the RX CDR PLL to lock to the incoming data. Bit <*n*> corresponds to channel <*n*>. By default, the Interlaken PHY IP configures the CDR PLL in Auto lock Mode. This bit is part of the CDR PLL Manual Lock Mode which is not the recommended usage.

Word Addr	Bits	R/W	Register Name	Description
0x065	[31:0]	RW	pma_rx_set_locktoref	When set, programs the RX CDR PLL to lock to the reference clock. Bit <*n*> corresponds to channel <*n*>. By default, the Interlaken PHY IP configures the CDR PLL in Auto lock Mode. This bit is part of the CDR PLL Manual Lock Mode which is not the recommended usage.
0x066	[31:0]	RO	pma_rx_is_ lockedtodata	When asserted, indicates that the RX CDR PLL is locked to the RX data, and that the RX CDR has changed from LTR to LTD mode. Bit <*n*> corresponds to channel <*n*>.
00x067	[31:0]	RO	pma_rx_is_ lockedtoref	When asserted, indicates that the RX CDR PLL is locked to the reference clock. Bit <*n*> corresponds to channel <*n*>.
0x080	[31:0]	WO	indirect_addr	Provides for indirect addressing of all PCS control and status registers. Use this register to specify the logical channel address of the PCS channel you want to access.
Device Registers				
	[27]	RO	rx_crc32_err	Asserted by the CRC32 checker to indicate a CRC error in the corresponding RX lane. **From block**: CRC32 checker.
0x081	[25]	RO	rx_sync_lock	Asserted by the frame synchronizer to indicate that 4 frame synchronization words have been received so that the RX lane is synchronized. **From block**: Frame synchronizer.
	[24]	RO	rx_word_lock	Asserted when the first alignment pattern is found. The RX FIFO generates this synchronous signal. **From block**: The RX FIFO generates this synchronous signal.

Related Information

Loopback Modes on page 17-59

Why Transceiver Dynamic Reconfiguration

Dynamic reconfiguration is necessary to calibrate transceivers to compensate for variations due to PVT.

As silicon progresses towards smaller process nodes, circuit performance is affected more by variations due to process, voltage, and temperature (PVT). These process variations result in analog voltages that can be offset from required ranges. Dynamic reconfiguration calibrates transceivers to compensate for variations due to PVT,

Each channel and each TX PLL have separate dynamic reconfiguration interfaces. The MegaWizard Plug-In Manager provides informational messages on the connectivity of these interfaces. The following example shows the messages for a 4-channel Interlaken PHY IP Core.

Example 8-1: Informational Messages for the Transceiver Reconfiguration Interface

```
PHY IP will require 5 reconfiguration interfaces for connection to the
external reconfiguration controller.

Reconfiguration interface offsets 0-3 are connected to the transceiver
channels.

Reconfiguration interface offset 4 is connected to the transmit PLL.
```

Although you must initially create a separate reconfiguration interface for each channel and TX PLL in your design, when the Intel Quartus Prime software compiles your design, it reduces the number of reconfiguration interfaces by merging reconfiguration interfaces. The synthesized design typically includes a reconfiguration interface for at least three channels because three channels share an Avalon-MM slave interface which connects to the Transceiver Reconfiguration Controller IP Core. Conversely, you cannot connect the three channels that share an Avalon-MM interface to different Transceiver Reconfiguration Controller IP cores. Doing so causes a Fitter error. For more information, refer to "Transceiver Reconfiguration Controller to PHY IP Connectivity" .

Dynamic Transceiver Reconfiguration Interface

This section describes the signals in the reconfiguration interface. This interface uses the Avalon-MM PHY Management interface clock.

Table 8-11: Reconfiguration Interface

Signal Name	Direction	Description
`reconfig_to_xcvr [(<n>70)-1:0]`	Input	Reconfiguration signals from the Transceiver Reconfiguration Controller. <*n*> grows linearly with the number of reconfiguration interfaces. <*n*> initially includes the total number transceiver channels and TX PLLs before optimization/merging.

Signal Name	Direction	Description
`reconfig_from_xcvr [(<n>46)-1:0]`	Output	Reconfiguration signals to the Transceiver Reconfiguration Controller. <*n*> grows linearly with the number of reconfiguration interfaces. <*n*> initially includes the total number transceiver channels before optimization/merging.

Note: Transceiver dynamic reconfiguration requires that you assign the starting channel number.

Interlaken PHY TimeQuest Timing Constraints

This section describes the Interlaken PHY TimeQuest timing constraints.

You must add the following TimeQuest constraint to your Synopsys Design Constraints File (.sdc) timing constraint file:

```
derive_pll_clocks -create_base_clocks
```

Note: The SDC timing constraints and approaches to identify false paths listed for Stratix V Native PHY IP apply to all other transceiver PHYs listed in this user guide. Refer to *SDC Timing Constraints of Stratix V Native PHY* for details.

Related Information

SDC Timing Constraints of Stratix V Native PHY on page 13-72
This section describes SDC examples and approaches to identify false timing paths.

Interlaken PHY Simulation Files and Example Testbench

This section describes the Interlaken PHY simulation files and example testbench.

Refer to " Running a Simulation Testbench" for a description of the directories and files that the Intel Quartus Prime software creates automatically when you generate your Interlaken PHY IP Core.

Refer to the Altera Wiki for an example testbench that you can use as a starting point in creating your own verification environment.

Related Information

- **Running a Simulation Testbench** on page 1-6
- **Altera Wiki**

PHY IP Core for PCI Express (PIPE)

9

2020.06.02

UG-01080 Subscribe Send Feedback

The Altera PHY IP Core for PCI Express (PIPE) implements physical coding sublayer (PCS) and physical media attachment (PMA) modules for Gen1, Gen2, and Gen3 data rates.

The Gen1 and Gen2 datapaths are compliant to the *Intel PHY I nterface for PCI Express (PIPE) Architecture PCI Express 2.0* specification. The Gen3 datapath is compliant to the *PHY Interface for the PCI Express Architecture PCI Express 3.0* specification. You must connect this PHY IP Core for PCI Express to a third-party PHY MAC to create a complete PCI Express design.

The PHY IP Core for PCI Express supports ×1, ×2, ×4, or ×8 operation for a total aggregate bandwidth ranging from 2 to 64 Gbps. In Gen1 and Gen2 modes, the PCI Express protocol uses 8B/10B encoding which has a 20% overhead. Gen3 modes uses 128b/130b encoding which has an overhead of less than 1%. The Gen3 PHY initially trains to L0 at the Gen1 data rate using 8B/10B encoding. When the data rate changes to Gen3, the link changes to 128b/130b encoding.

Altera also provides a complete hard IP solution for PCI Express that includes the Transaction, Data Link and PHY MAC. For more information about Altera's complete hard IP solution, refer to the *Stratix V Hard IP for PCI Express IP Core User Guide*.

Figure 9-1 illustrates the top-level blocks of the Gen3 PCI Express PHY (PIPE) for Stratix V GX devices. **Figure 9-2** illustrates the top-level blocks of the Gen1 and Gen2 IP cores. As these figures illustrate, the PIPE interface connects to a third-party MAC PHY implemented using soft logic in the FPGA fabric. The reconfiguration buses connect to the Transceiver Reconfiguration Controller IP Core. For more information about this component, refer to Transceiver Reconfiguration Controller IP Core. An embedded processor connects to an Avalon-MM PHY management interface for control and status updates.

Figure 9-1: Gen3 PCI Express PHY (PIPE) with Hard IP PCS and PMA in Arria V GZ and Stratix V GX Devices

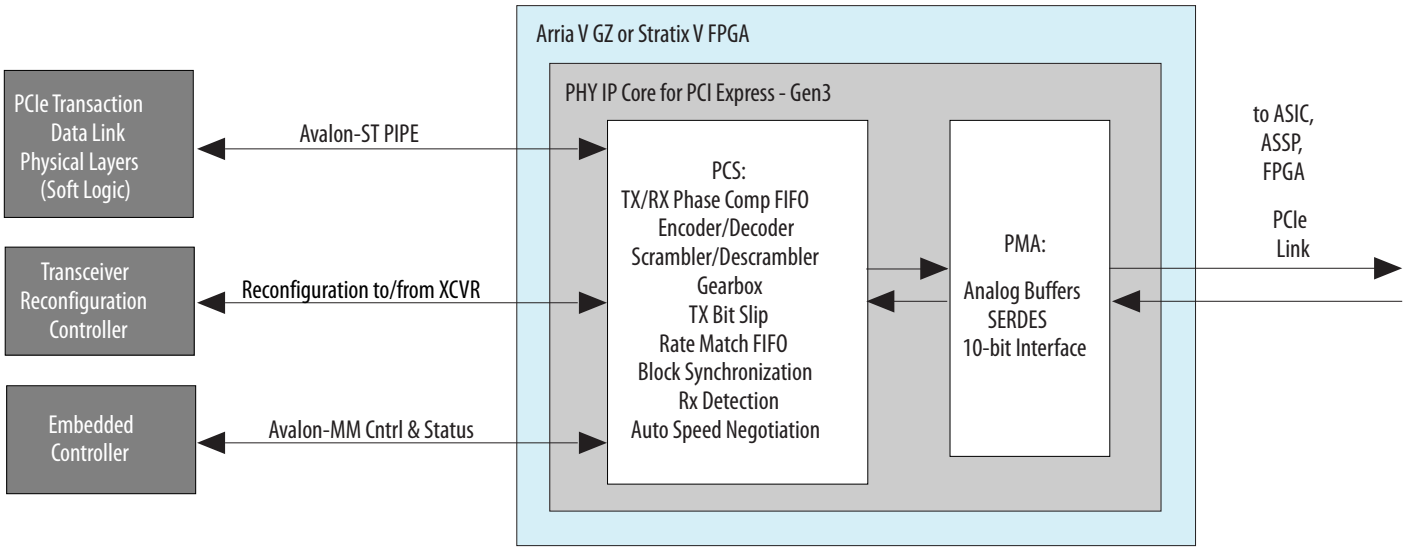
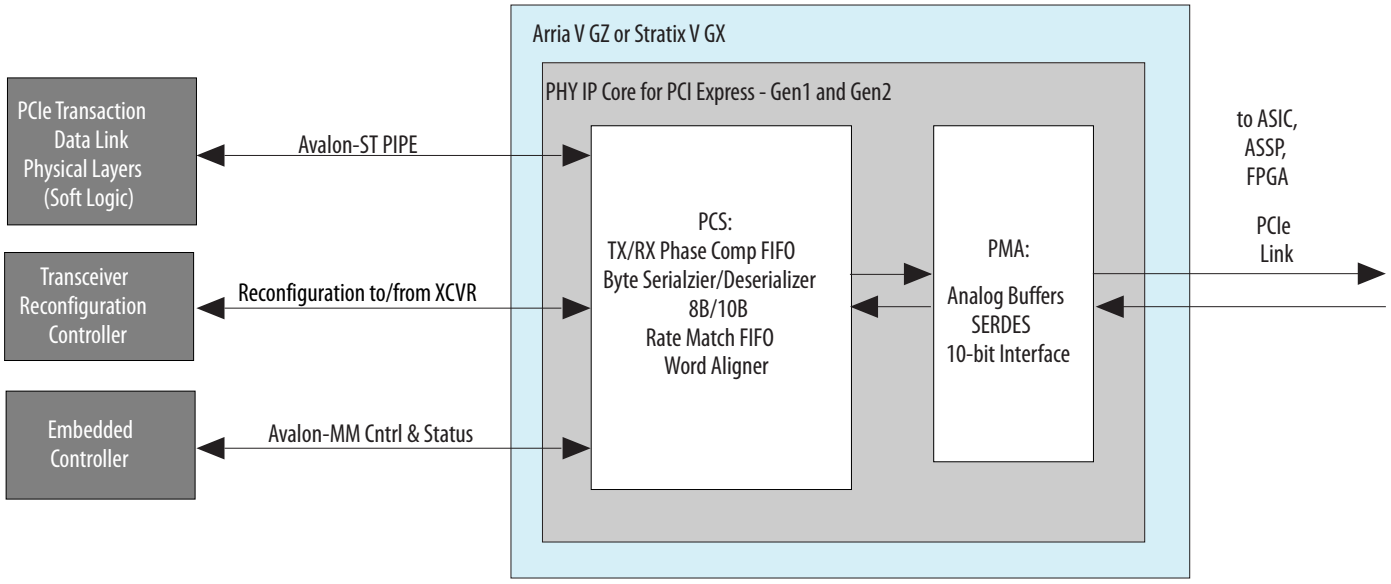


Figure 9-2: Gen1 and Gen2 PCI Express PHY (PIPE) with Hard IP PCS and PMA in Arria V GZ and Stratix V GX Devices



For more detailed information about the PCI Express PHY PIPE transceiver channel datapath, clocking, and channel placement, refer to the "PCI Express" section in the in the *Transceiver Configurations in Arria V GZ Devices or Transceiver Configurations in Stratix V Devices* as appropriate.

Related Information

- **Intel PHY I nterface for PCI Express (PIPE) Architecture PCI Express 2.0**
- **PHY Interface for the PCI Express Architecture PCI Express 3.0**

- **Stratix V Hard IP for PCI Express IP Core User Guide**
- **Transceiver Configurations in Arria V GZ Devices or Transceiver Configurations in Stratix V Devices**

PHY for PCIe (PIPE) Device Family Support

IP cores provide either final or preliminary support for target Altera device families. These terms have the following definitions:

- *Final support*—Verified with final timing models for this device.
- *Preliminary support*—Verified with preliminary timing models for this device.

Table 9-1: Device Family Support

Device Family	Support
Arria V GZ devices - Hard PCS + PMA	Final
Arria V GX, GT, SX, and ST devices - Hard PCS + PMA	Final
Stratix V devices - Hard PCS + PMA	Final
Cyclone V devices - Hard PCS + PMA (5)	Final
Other device families	No support

PHY for PCIe (PIPE) Resource Utilization

This section describes PIPE resource utilization.

Because the PHY IP Core for PCI Express is implemented in hard logic it uses less than 1% of the available adaptive logic modules (ALMs), memory, primary and secondary logic registers.

Parameterizing the PHY IP Core for PCI Express (PIPE)

Complete the following steps to configure the PHY IP Core for PCI Express in the MegaWizard Plug-In Manager:

1. Under **Tools** > **IP Catalog**, select the device family of your choice.
2. Under **Tools** > **IP Catalog** >**Interfaces** > **PCI Express**, select**PHY IP Core for PCI Express (PIPE)**.
3. Use the tabs on the MegaWizard Plug-In Manager to select the options required for the protocol.
4. Refer to the General Options Parameters to learn more about the parameters.
5. Click **Finish** to generate your customized PHY IP Core for PCI Express variant.

(5) Cyclone V devices only supports Gen1.

PHY for PCIe (PIPE) General Options Parameters

This section describes the PHY IP Core for PCI Express parameters, which you can set using the MegaWizard Plug-In Manager; the settings are available on the **General Options** tab.

Table 9-2: PHY IP Core for PCI Express General Options

Name	Value	Description
Device family	**Stratix V** **Arria V GZ** **Arria V GX** **Arria V GT** **Arria V SX** **Arria V ST**	Supports all Arria V and Stratix V devices.
Number of lanes	**1, 2, 4, 8**	The total number of duplex lanes.
Protocol version	**Gen1 (2.5 Gbps)** **Gen2 (5.0 Gbps)** **Gen3 (8.0 Gbps)**	The Gen1 and Gen2 implement the *Intel PHY Interface for PCI Express (PIPE) Architecture PCI Express 2.0* specification. The Gen3 implements the *PHY Interface for the PCI Express Architecture PCI Express 3.0* specification.
Gen1 and Gen2 base data rate	1 × **Lane rate** 2 × **Lane rate** 4 × **Lane rate** 8 ×**Lane rate**	The **base data rate** is the output clock frequency of the TX PLL. Select a **base data rate** that minimizes the number of PLLs required to generate all the clocks required for data transmission.
Data rate	**2500 Mbps** **5000 Mbps** **8000 Mbps**	Specifies the data rate. This parameter is based on the **Protocol version** you specify. You cannot change it.

Name	Value	Description
Gen1 and Gen2 PLL type	**CMU** **ATX**	You can select either the CMU or ATX PLL. The CMU PLL has a larger frequency range than the ATX PLL. The ATX PLL is designed to improve jitter performance and achieves lower channel-to-channel skew; however, it supports a narrower range of data rates and reference clock frequencies. Another advantage of the ATX PLL is that it does not use a transceiver channel, while the CMU PLL does. Gen3 variants require 2 PLLs for link training which begins in Gen1 and negotiates up to Gen3 if both sides of the link are Gen3 capable.
Gen3 PLL type	**ATX**	Gen3 uses the ATX PLL because its jitter characteristics are better than the CMU PLL for data rates above 6 Gbps.
PLL reference clock frequency	**100 MHz** **125 MHz**	You can use either the **100 MHz** or **125 MHz** input reference clock. (The PCI Express specifications, require an 100 MHz reference clock.)
FPGA transceiver width	**8, 16, 32**	Specifies the width of the interface between the PHY MAC and PHY (PIPE).The following options are available: • Gen1: 8 or 16 bits • Gen2: 16 bits • Gen3: 32 bits Using the Gen1 16-bit interface reduces the required clock frequency by half at the expense of extra FPGA resources.
Run length	**5–160**	Specifies the maximum number of consecutive 0s or 1s that can occur in the data stream. The `rx_rlv` signal is asserted if the maximum run length is violated.

Related Information

- **Intel PHY Interface for PCI Express (PIPE) Architecture PCI Express 2.0**
- **PHY Interface for the PCI Express Architecture PCI Express 3.0**

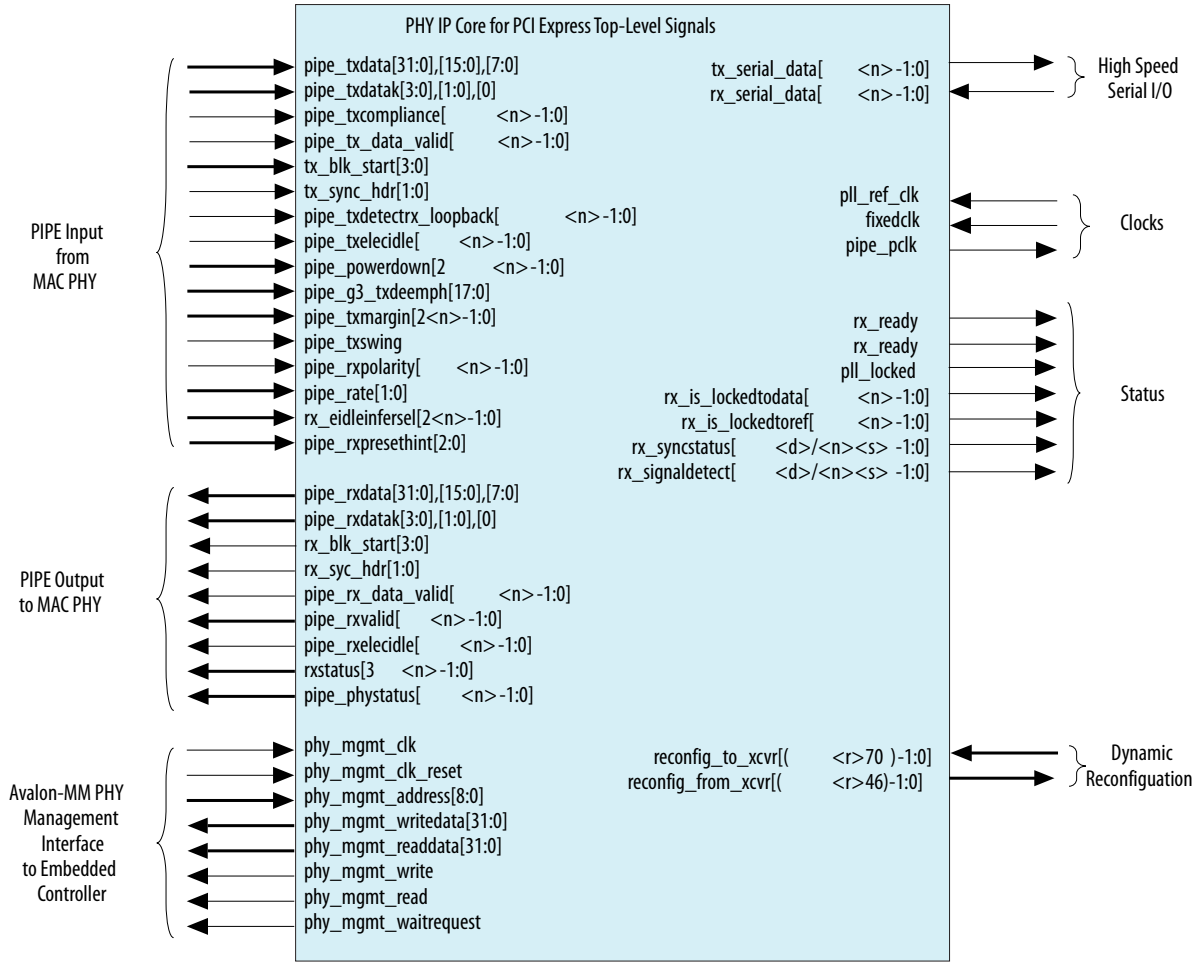
PHY for PCIe (PIPE) Interfaces

This section describes interfaces of the PHY IP Core for PCI Express (PIPE).

The following figure illustrates the top-level pinout of the PHY IP Core for PCI Express PHY. The port descriptions use the following variables to represent parameters:

- <*n*>—The number of lanes
- <*d*>—The total deserialization factor from the input pin to the PHY MAC interface.
- <*s*>—The symbols size.
- <*r*>—The width of the reconfiguration interface; <r> is automatically calculated based on the selected configuration.

Figure 9-3: Top-Level Signals of the PHY IP Core for PCI Express



Note: The **block diagram** shown in the GUI labels the external pins with the *interface type* and places the *interface name* inside the box. The interface type and name are used in the **_hw.tcl** file. If you turn on **Show signals**, the **block diagram** displays all top-level signal names.

For more information about **_hw.tcl** files, refer to refer to the *Component Interface Tcl Reference* chapter in volume 1 of the *Intel Quartus Prime Handbook*.

Related Information

Component Interface Tcl Reference

PHY for PCIe (PIPE) Input Data from the PHY MAC

Input data signals are driven from the PHY MAC to the PCS. This interface is compliant to the appropriate PIPE interface specification.

For more information about the Avalon-ST protocol, including timing diagrams, refer to the *Avalon Interface Specifications*.

Table 9-3: Avalon-ST TX Inputs

Signal Name	Direction	Description
Gen1 and Gen2		
`pipe_txdata[31:0]`, `[15:0]`, or `[7:0]`	Input	Parallel PCI Express data input bus. For the 16-bit interface, 16 bits represent 2 symbols of transmit data. Bits [7:0] is transmitted first; bits[15:8] are transmitted second. Bit 0 if the first to be transmitted. For the 32-bit interface, 32 bits represent the 4 symbols of TX data. Bits[23:16] are the third symbol to be transmitted and bits [31:24] are the fourth symbol.
`pipe_txdatak[(3:0]`, `[1:0]` or `[0]`	Input	For Gen1 and Gen2, data and control indicator for the received data. When 0, indicates that *pipe_txdata* is data, when 1, indicates that `pipe_txdata` is control. For Gen3, Bit[0] corresponds to `pipe_txdata[7:0]`, bit[1] corresponds to `pipe_txdata[15:8]`, and so on.
`pipe_txcompliance`	Input	Asserted for one cycle to set the running disparity to negative. Used when transmitting the compliance pattern. Refer to section 6.11 of the *Intel PHY Interface for PCI Express (PIPE) Architecture* for more information.

Signal Name	Direction	Description
pipe_tx_data_valid[<n>-1:0]	Input	For Gen3, pipe_tx_data_valid[<n>-1:0] is deasserted by the MAC to instruct the PHY to ignore pipe_txdata for one clock cycle. A value of 0 indicates the PHY should use the data. A value of 1 indicates the PHY should not use the data.
tx_blk_start	Input	For Gen3, specifies start block byte location for TX data in the 128-bit block data. Used when the interface between the PCS and PHY MAC is 32 bits. Not used for the Gen1 and Gen2 data rates.
tx_sync_hdr[1:0]	Input	For Gen3, indicates whether the 130-bit block being transmitted is a Data or Control Ordered Set Block. The following encodings are defined: • 2'b10: Data block • 2'b01: Control Ordered Set Block This value is read when tx_blk_start = 1b'1. Refer to "*Section 4.2.2.1. Lane Level Encoding*" in the *PCI Express Base Specification, Rev. 3.0* for a detailed explanation of data transmission and reception using 128b/130b encoding and decoding. Not used for the Gen1 and Gen2 data rates.
pipe_txdetectrx_loopback	Input	This signal instructs the PHY to start a receive detection operation. After power-up asserting this signal starts a loopback operation. Refer to section 6.4 of the *Intel PHY Interface for PCI Express (PIPE)* for a timing diagram.
pipe_txelecidle	Input	This signal forces the transmit output to electrical idle. Refer to section 7.3 of the *Intel PHY Interface for PCI Express (PIPE)* for timing diagrams.
pipe_powerdown<n>[1:0]	Input	This signal requests the PHY to change its power state to the specified state. The following encodings are defined: • 2'b00– P0, normal operation • 2'b01–P0s, low recovery time latency, power saving state • 2'b10–P1, longer recovery time (64 us maximum latency), lower power state • 2'b11–P2, lowest power state. (not supported)

Signal Name	Direction	Description
pipe_txdeemph	Input	Transmit de-emphasis selection. In PCI Express Gen2 (5 Gbps) mode it selects the transmitter de-emphasis: • 1'b0: -6 dB • 1'b1: -3.5 dB
pipe_g3_txdeemph[17:0]	Input	For Gen3, selects the transmitter de-emphasis. The 18 bits specify the following coefficients: • [5:0]: C_{-1} • [11:6]: C_0 • [17:12]: C_{+1} Refer to **Table 9-4** for presets to TX de-emphasis mappings. In Gen3 capable designs, the TX deemphasis for Gen2 data rates is always -6 dB. The TX deemphasis for Gen1 data rate is always -3.5 dB.
pipe_txmargin[3<n>-1:0]	Input	Transmit V_{OD} margin selection. The MAC PHY sets the value for this signal based on the value from the Link Control 2 Register. The following encodings are defined: • 3'b000: Normal operating range • 3'b001: Full swing: 800 - 1200 mV; Half swing: 400 - 700 mV • 3'b010:–3'b011: Reserved • 3'b100–3'b111: If last value, full swing: 200 - 400 mV, half swing: 100 - 200 mV else reserved
pipe_txswing	Input	Indicates whether the transceiver is using full- or low-swing voltages as defined by the tx_pipemargin. • 1'b0–Full swing. • 1'b1–Low swing.
pipe_rxpolarity	Input	When 1, instructs the PHY layer to invert the polarity on the received data. PCIe Gen 1 & 2 has its inversion blocks placed immediately prior to word alignment, whereas PCIe Gen 3 inverts the data coming from the PMA prior to block synchronization.

Signal Name	Direction	Description
pipe_rate[1:0]	Input	The 2-bit encodings have the following meanings: • 2'b00: Gen1 rate (2.5 Gbps) • 2'b01: Gen2 rate (5.0 Gbps) • 2'b1x: Gen3 (8.0 Gbps) The Rate Switch from Gen1 to Gen2 Timing Diagram illustrates the timing of a rate switch from Gen1 to Gen2 and back to Gen1.
rx_eidleinfersel[3<n>-1:0]	Input	When asserted high, the electrical idle state is inferred instead of being identified using analog circuitry to detect a device at the other end of the link. The following encodings are defined: • 3'b0xx: Electrical Idle Inference not required in current LTSSM state • 3'b100: Absence of COM/SKP OS in 128 ms window for Gen1 or Gen2 • 3'b101: Absence of TS1/TS2 OS in 1280 UI interval for Gen1 or Gen2 • 3'b110: Absence of Electrical Idle Exit in 2000 UI interval for Gen1 and 16000 UI interval for Gen2 • 3'b111: Absence of Electrical Idle exit in 128 ms window for Gen1
pipe_rxpresethint[2:0]	Input	Provides the RX preset hint for the receiver. Only used for the Gen3 data rate.

Table 9-4: Preset Mappings to TX De-Emphasis

Preset	C_{+1}	C_0	C_{-1}
1	001001	011010	000000
2	000110	011101	000000
3	000111	011100	000000
4	000101	011110	000000
5	000000	100011	000000
6	000000	011111	000100
7	000000	011110	000101
8	000111	011000	000100
9	000101	011010	000100

Preset	C_{+1}	C_0	C_{-1}
10	000000	011101	000110
11	001011	011000	000000

Related Information

- Avalon Interface Specifications
- Intel PHY Interface for PCI Express (PIPE) Architecture
- PCI Express Base Specification, Rev. 3.

PHY for PCIe (PIPE) Output Data to the PHY MAC

This section describes the PIPE output signals. These signals are driven from the PCS to the PHY MAC. This interface is compliant to the appropriate PIPE interface specification.

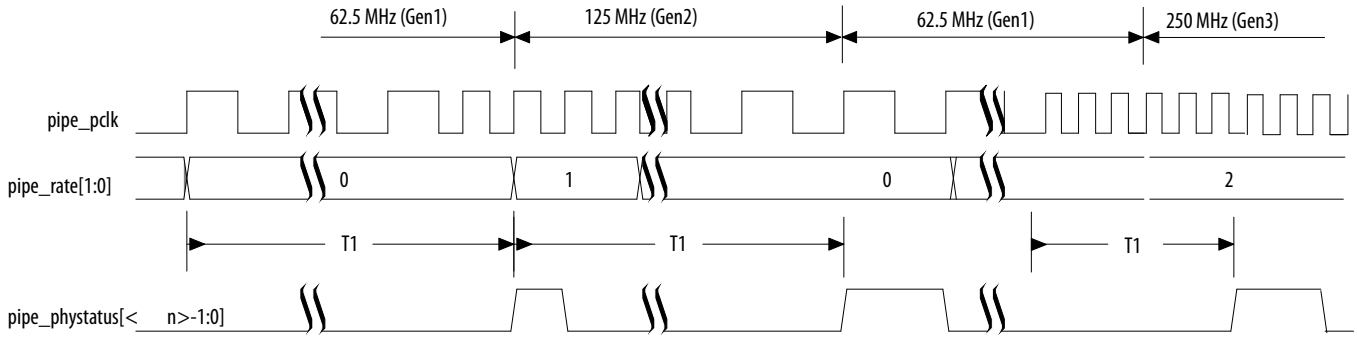
Table 9-5: Avalon-ST RX Inputs

Signal Name	Direction	Description
`pipe_rxdata[[(31,16or 8)-1:0]`	Output	This is RX parallel data driven from the PCS to the MAC PHY. The ready latency on this interface is 0, so that the MAC must be able to accept data as soon as the PHY comes out of reset. Width is 8 or 16 for Gen1 and Gen2. Width is 32 for Gen3. Transmission is little endian. For example, for Gen3, words are transmitted in the following order: • PIPE word 0: `pipe_rxdata[7:0]` • PIPE word 1: `pipe_rxdata[15:8]` • PIPE word 2: `pipe_rxdata[23:16]` • PIPE word 3: `pipe_rxdata[31:24]`
`pipe_rxdatak[(3,2 or 1)-1:0]`	Output	Data and control indicator for the source data. When 0, indicates that `pipe_rxdata` is data, when 1, indicates that `pipe_rxdata` is control. Bit[0] corresponds to byte 0. Bit[]1 corresponds to byte 1, and so on.
`rx_blk_start[3:0]`	Output	For Gen3 operation, indicates the block starting byte location in the received 32-bits data of the 130-bits block data. Data reception must start in bits [7:0] of the 32-bit data word, so that the only valid value is 4'b0001.

Signal Name	Direction	Description
rx_sync_hdr[1:0]	Output	For Gen3, indicates whether the 130-bit block being transmitted is a Data or Control Ordered Set Block. The following encodings are defined: • 2'b10: Data block • 2'b01: Control Ordered Set block This valued is read when rx_blk_start = 4'b0001. Refer to "*Section 4.2.2.1. Lane Level Encoding*" in the *PCI Express Base Specification, Rev. 3.0* for a detailed explanation of data transmission and reception using 128b/130b encoding and decoding.
pipe_rx_data_valid	Output	For Gen3, this signal is deasserted by the PHY to instruct the MAC to ignore pipe_rxdata for one clock cycle. A value of 1 indicates the MAC should use the data. A value of 0 indicates the MAC should not use the data.
pipe_rxvalid[<n>-1:0]	Output	Asserted when RX data and control are valid.
pipe_rxelecidle	Output	When asserted, indicates receiver detection of an electrical idle. For Gen2 and Gen3 data rates, the MAC uses logic to detect electrical idle entry instead of relying of this signal.
rxstatus<n>[2:0]	Output	This signal encodes receive status and error codes for the receive data stream and receiver detection.The following encodings are defined: • 3'b000–receive data OK • 3'b001–1 SKP added • 3'b010–1 SKP removed • 3'b011–Receiver detected • 3'b100–Both 8B/10B or 128b/130b decode error and (optionally) RX disparity error • 3'b101–Elastic buffer overflow • 3'b110–Elastic buffer underflow • 3'b111–Receive disparity error, not used if disparity error is reported using 3'b100.
pipe_phystatus	Output	This signal is used to communicate completion of several PHY requests.

Figure 9-4: Rate Switch from Gen1 to Gen2 Timing Diagram

In the figure, Time T1 is pending characterization and <*n*> is the number of lanes.



Related Information

PCI Express Base Specification, Rev. 3.0

PHY for PCIe (PIPE) Clocks

This section describes the clock ports.

Table 9-6: Clock Ports

Signal Name	Direction	Description
`pll_ref_clk`	Input	This is the 100 MHz input reference clock source for the PHY TX and RX PLL. You can optionally provide a 125 MHz input reference clock by setting the **PLL reference clock frequency** parameter to 125 MHz as described in PHY IP Core for PCI Express General Options.
`fixedclk`	Input	A 100 MHz or 125 MHz clock used for the receiver detect circuitry. This clock can be derived from `pll_ref_clk`.
`pipe_pclk`	Output	Generated in the PMA and driven to the MAC PHY interface. All data and status signals are synchronous to `pipe_pclk`. This clock has the following frequencies: • Gen1: 62.5 MHz • Gen2:125 MHz • Gen3: 250 MHz

The following table lists the `pipe_pclk` frequencies for all available PCS interface widths. Doubling the FPGA transceiver width haves the required frequency.

Table 9-7: pipe_pclk Frequencies

Capability	FPGA Transceiver Width	Gen1	Gen2	Gen3
Gen1 only	8 bits	250 MHz	—	—
	16 bits	125 MHz	—	—
Gen2 capable	16 bits	125 MHz	250 MHz	—
Gen3 capable	32 bits	62.5 MHz	125 MHz	250 MHz

PHY for PCIe (PIPE) Clock SDC Timing Constraints for Gen3 Designs

For Gen3 designs, you must add the following timing constraints to force Timequest to analyze the design at Gen1, Gen2 and Gen3 data rates. Include these constraints in your top-level SDC file for the project.

Add the following command to force Timequest analysis at 250 MHz.

```
create_generated_clock -name clk_g3 -source [get_ports {pll_refclk}] \
-divide_by 2 -multiply_by 5 -duty_cycle 50 -phase 0 -offset 0 [get_nets
 {*pipe_nr_inst|transceiver_core|inst_sv_xcvr_native|inst_sv_pcs|\
|ch[*].inst_sv_pcs_ch|inst_sv_hssi_tx_pld_pcs_interface|pld8gtxclkout}] -add
```

Add the following command to force Timequest analysis at 62.5 MHz.

```
create_generated_clock -name  clk_g1 -source [get_ports {pll_refclk}] \
-divide_by 8 -multiply_by 5 -duty_cycle 50 -phase 0 -offset 0 [get_nets  \
{*pipe_nr_inst|transceiver_core|inst_sv_xcvr_native|inst_sv_pcs| \
ch[*].inst_sv_pcs_ch|inst_sv_hssi_tx_pld_pcs_interface|pld8gtxclkout}] -add


#creating false paths between these clock groups
set_clock_groups -asynchronous -group [get_clocks clk_g3]
set_clock_groups -asynchronous -group [get_clocks clk_g1]
set_clock_groups -asynchronous -group [get_clocks *pipe_nr_inst| \
transceiver_core|inst_sv_xcvr_native|inst_sv_pcs|ch[*]. \
inst_sv_pcs_ch|inst_sv_hssi_8g_tx_pcs|wys|clkout]
```

PHY for PCIe (PIPE) Optional Status Interface

This section describes the signals the optional status signals.

Table 9-8: Status Signals

Signal Name [6]	Direction	Signal Name
`tx_ready`	Output	When asserted, indicates that the TX interface has exited the reset state and is ready to transmit.
`rx_ready`	Output	When asserted, indicates that the RX interface has exited the reset state and is ready to receive.

[6] <*n*> is the number of lanes. <*d*> is the deserialization factor. < *p*> is the number of PLLs.

Signal Name [6]	Direction	Signal Name
pll_locked[<p>-1:0]	Output	When asserted, indicates that the TX PLL is locked to the input reference clock. This signal is asynchronous.
rx_is_lockedtodata[<n>-1:0]	Output	When asserted, the receiver CDR is in to lock-to-data mode. When deasserted, the receiver CDR lock mode depends on the rx_locktorefclk signal level.
rx_is_lockedtoref[<n>-1:0]	Output	Asserted when the receiver CDR is locked to the input reference clock. This signal is asynchronous.
rx_syncstatus[<d><n>/8-1:0]	Output	Indicates presence or absence of synchronization on the RX interface. Asserted when word aligner identifies the word alignment pattern or synchronization code groups in the received data stream.
rx_signaldetect[<d><n>/8-1:0]	Output	When asserted indicates that the lane detects a sender at the other end of the link.

PHY for PCIe (PIPE) Serial Data Interface

This section describes the differential serial TX and RX connections to FPGA pins.

Table 9-9: Transceiver Differential Serial Interface

Signal Name	Direction	Description
rx_serial_data[<n>-1:0]	Input	Receiver differential serial input data, <n> is the number of lanes.
tx_serial_data[<n>-1:0]	Output	Transmitter differential serial output data <n> is the number of lanes.

For information about channel placement, refer to "Transceiver Clocking and Channel Placement Guidelines" in the *Transceiver Configurations in Arria V GZ Devices* or "Transceiver Clocking and Channel Placement Guidelines" in the *Transceiver Configurations in Stratix V Devices* as appropriate.

Note: For soft IP implementations of PCI Express, channel placement is determined by the Intel Quartus Prime fitter.

For information about channel placement of the Hard IP PCI Express IP Core, refer to the *Channel Placement Gen1 and Gen2 and Channel Placement Gen3* sections in the *Stratix V Hard IP for PCI Express User Guide*.

Related Information

- **Transceiver Configurations in Arria V GZ Devices**
- **Transceiver Configurations in Stratix V Devices**
- **Stratix V Hard IP for PCI Express User Guide**

[6] <*n*> is the number of lanes. <*d*> is the deserialization factor. < *p*> is the number of PLLs.

PHY for PCIe (PIPE) Register Interface and Register Descriptions

The Avalon-MM PHY management interface provides access to the PHY IP Core for PCI Express PCS and PMA features that are not part of the standard PIPE interface. You can use an embedded controller acting as an Avalon-MM master to send read and write commands to this Avalon-MM slave interface.

The following figure provides a high-level view of this hardware; modules shown in white are hard logic and modules shown in gray are soft logic.

Figure 9-5: PCI Express PIPE IP Core Top-Level Modules

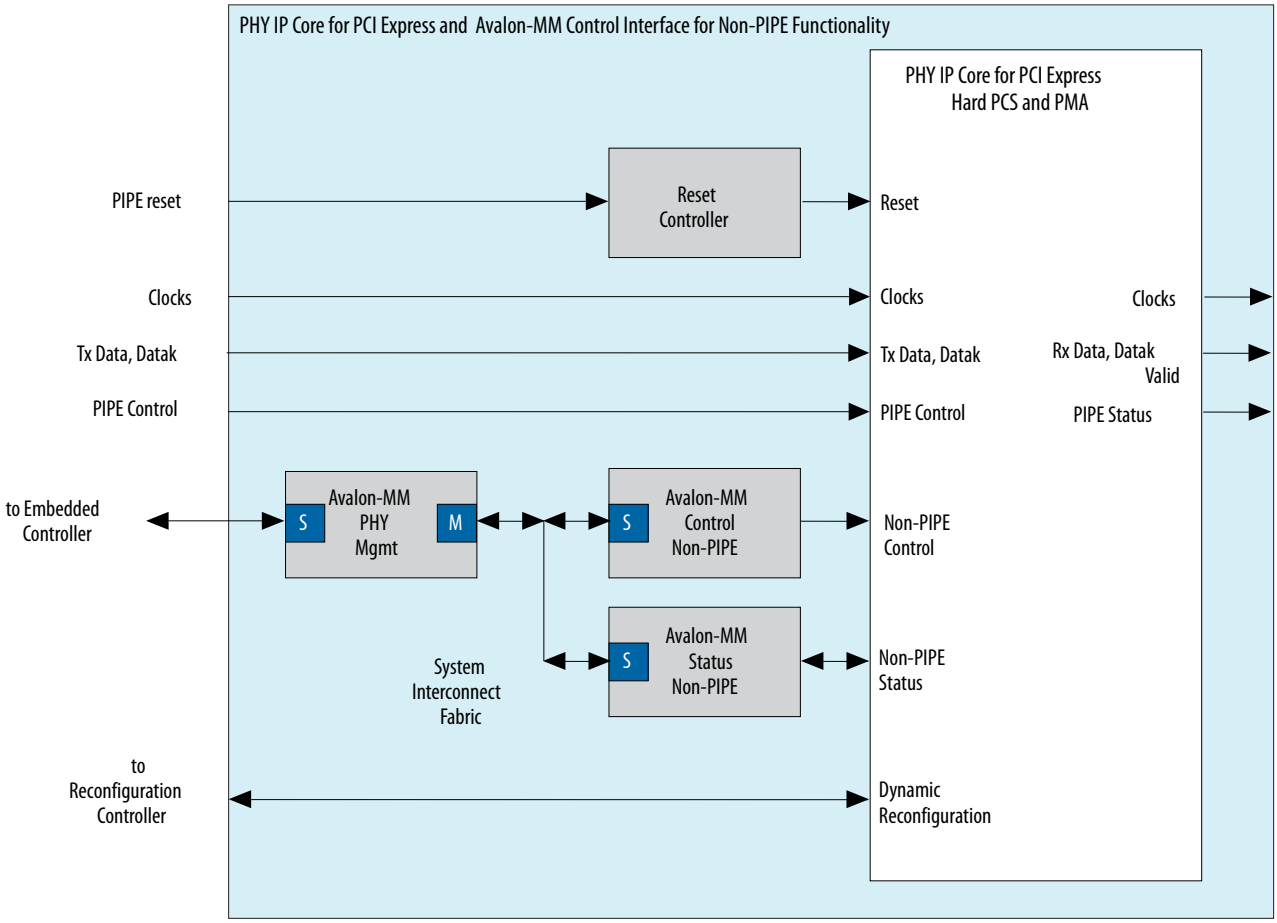


Table 9-10: Avalon-MM PHY Management Interface

Signal Name	Direction	Description
phy_mgmt_clk	Input	Avalon-MM clock input. There is no frequency restriction for Stratix V devices; however, if you plan to use the same clock for the PHY management interface and transceiver reconfiguration, you must restrict the frequency range of phy_mgmt_clk to 100-125 MHz to meet the specification for the transceiver reconfiguration clock.
phy_mgmt_clk_reset	Input	Global reset signal that resets the entire PHY IP core. This signal is active high and level sensitive.
phy_mgmt_address[8:0]	Input	9-bit Avalon-MM address.
phy_mgmt_writedata[31:0]	Input	Input data.
phy_mgmt_readdata[31:0]	Output	Output data.
phy_mgmt_write	Input	Write signal.
phy_mgmt_read	Input	Read signal.
phy_mgmt_waitrequest	Output	When asserted, indicates that the Avalon-MM slave interface is unable to respond to a read or write request. When asserted, control signals to the Avalon-MM slave interface must remain constant.

PHY for PCIe (PIPE) Register Interface and Register Descriptions on page 9-16 describes the registers that you can access over the Avalon-MM PHY management interface using word addresses and a 32-bit embedded processor. A single address space provides access to all registers.

Note: Writing to reserved or undefined register addresses may have undefined side effects.

Table 9-11: PCI Express PHY (PIPE) IP Core Registers

Word Addr	Bits	R/W	Register Name	Description
PMA Common Control and Status Registers				
0x022	[31:0]	R	pma_tx_pll_is_locked	Bit[P] indicates that the TX CMU PLL (P) is locked to the input reference clock. There is typically one pma_tx_pll_is_locked bit per system.
Reset Control Registers–Automatic Reset Controller				
0x041	[31:0]	RW	reset_ch_bitmask	Reset controller channel bitmask for digital resets. The default value is all 1s. Channel <*n*> can be reset when bit<*n*> = 1.

Word Addr	Bits	R/W	Register Name	Description
0x042	[1:0]	W	`reset_control` (write)	Writing a 1 to bit 0 initiates a TX digital reset using the reset controller module. The reset affects channels enabled in the `reset_ch_bitmask`. Writing a 1 to bit 1 initiates a RX digital reset of channels enabled in the `reset_ch_bitmask`. Refer to Timing Constraints for Reset Signals when Using Bonded PCS Channels for a SDC constraint you must include in your design.
		R	`reset_status` (read)	Reading bit 0 returns the status of the reset controller TX ready bit. Reading bit 1 returns the status of the reset controller RX ready bit.
Reset Controls –Manual Mode				

Word Addr	Bits	R/W	Register Name	Description
0x044	[31:0]	RW	reset_fine_control	You can use the reset_fine_control register to create your own reset sequence. The reset control module, illustrated in Transceiver PHY Top-Level Modules, performs a standard reset sequence at power on and whenever the phy_mgmt_clk_reset is asserted. Bits [31:4, 0] are reserved.
	[31:4]	RW	Reserved	It is safe to write 0s to reserved bits.
	[3]	RW	reset_rx_digital	Writing a 1 causes the RX digital reset signal to be asserted, resetting the RX digital channels enabled in reset_ch_bitmask. You must write a 0 to clear the reset condition.
	[2]	RW	reset_rx_analog	Writing a 1 causes the internal RX digital reset signal to be asserted, resetting the RX analog logic of all channels enabled in reset_ch_bitmask. You must write a 0 to clear the reset condition.
	[1]	RW	reset_tx_digital	Writing a 1 causes the internal TX digital reset signal to be asserted, resetting all channels enabled in reset_ch_bitmask. You must write a 0 to clear the reset condition. Refer to Timing Constraints for Reset Signals when Using Bonded PCS Channels for a SDC constraint you must include in your design.
	[0]	RW	pll_powerdown	Writing a 1 causes the internal TX PLL to powerdown. If you reset the transceiver, you must assert pll_powerdown by writing a 1 to this register and then writing a 0 after 1 ms.
PMA Control and Status Registers				
0x061	[31:0]	RW	phy_serial_loopback	Writing a 1 to channel <*n*> puts channel <*n*> in serial loopback mode.
0x063	[31:0]	R	pma_rx_signaldetect	When channel <*n*> =1, indicates that receive circuit for channel <*n*> senses the specified voltage exists at the RX input buffer. This option is only operational for the PCI Express PHY IP Core.

Word Addr	Bits	R/W	Register Name	Description
0x064	[31:0]	RW	pma_rx_set_locktodata	When set, programs the RX CDR PLL to lock to the incoming data. Bit <*n*> corresponds to channel <*n*>.
0x065	[31:0]	RW	pma_rx_set_locktoref	When set, programs the RX CDR PLL to lock to the reference clock. Bit <*n*> corresponds to channel <*n*>.
0x066	[31:0]	R	pma_rx_is_lockedtodata	When asserted, indicates that the RX CDR PLL is locked to the RX data, and that the RX CDR has changed from LTR to LTD mode. Bit <*n*> corresponds to channel <*n*>.
0x067	[31:0]	R	pma_rx_is_lockedtoref	When asserted, indicates that the RX CDR PLL is locked to the reference clock. Bit <*n*> corresponds to channel <*n*>.
PCS for PCI Express				
0x080	[31:0]	RW	Lane or group number	Specifies lane or group number for indirect addressing, which is used for all PCS control and status registers. For variants that stripe data across multiple lanes, this is the logical group number. For non-bonded applications, this is the logical lane number.
0x081	[31:6]	R	Reserved	—
	[5:1]	R	rx_bitslipboundaryselectout	Records the number of bits slipped by the RX Word Aligner to achieve word alignment. Used for very latency sensitive protocols. **From block**: Word aligner.
	[0]	R	rx_phase_comp_fifo_error	When set, indicates an RX phase compensation FIFO error. **From block**: RX phase compensation FIFO.
0x082	[31:1]	R	Reserved	—
	[0]	RW	tx_phase_comp_fifo_error	When set, indicates a TX phase compensation FIFO error. **From block**: TX phase compensation FIFO.

Word Addr	Bits	R/W	Register Name	Description
0x083	[31:6]	RW	Reserved	—
	[5:1]	RW	tx_bitslipboundary_select	Sets the number of bits the TX block needs to slip the output. Used for very latency sensitive protocols. **From block**: TX bit-slipper.
0x084	[31:1]	RW	Reserved	—
0x085	[31:4]	RW	Reserved	—
	[3]	RW	rx_bitslip	When set, the word alignment logic operates in bitslip mode. Every time this register transitions from 0 to 1, the RX data slips a single bit. **From block**: Word aligner.
	[2]	RW	rx_bytereversal_enable	When set, enables byte reversal on the RX interface. **From block**: Word aligner.
	[1]	RW	rx_bitreversal_enable	When set, enables bit reversal on the RX interface. **From block**k: Word aligner.
	[0]	RW	rx_enapatternalign	When set, the word alignment logic operates in pattern detect mode. **From block**: Word aligner.

Word Addr	Bits	R/W	Register Name	Description
0x086	[31:20]	R	Reserved	—
	[19:16]	R	rx_rlv	When set, indicates a run length violation. **From block**: Word aligner.
	[15:12]	R	rx_patterndetect	When set, indicates that RX word aligner has achieved synchronization. **From block**: Word aligner.
	[11:8]	R	rx_disperr	When set, indicates that the received 10-bit code or data group has a disparity error. When set, the corresponding errdetect bits are also set. **From block**: 8B/10B decoder.
	[7:4]	R	rx_syncstatus	When set, indicates that the RX interface is synchronized to the incoming data. **From block**: Word aligner.
	[3:0]	R	rx_errdetect	When set, indicates that a received 10-bit code group has an 8B/10B code violation or disparity error. It is used along with RX disparity to differentiate between a code violation error and a disparity error, or both. In PIPE mode, the PIPE specific output port called pipe_rxstatus encodes the errors. **From block**: 8B/10B decoder.

For more information about the individual PCS blocks, refer to *Transceiver Architecture in Stratix V Devices* in the *Stratix V Device Handbook*.

Related Information

Transceiver Architecture in Stratix V Devices

PHY for PCIe (PIPE) Link Equalization for Gen3 Data Rate

Gen3 requires both TX and RX link equalization because of the data rate, the channel characteristics, receiver design, and process variations. The link equalization process allows the Endpoint and Root Port to adjust the TX and RX setup of each lane to improve signal quality. This process results in Gen3 links with a receiver Bit Error Rate (BER) that is less than 10^{-12}.

"*Section 4.2.3 Link Equalization Procedure for 8.0 GT/s Data Rate*" in the *PCI Express Base Specification, Rev. 3.0* provides detailed information about the four-stage link equalization procedure. A new LTSSM state, Recovery.Equalization with Phases 0–3, reflects progress through Gen3 equalization. Phases 2 and 3 of link equalization are optional; however, the link must progress through all four phases, even if no adjustments occur. Skipping Phases 2 and 3 speeds up link training at the expense of link BER optimization.

Related Information

PHY Interface for the PCI Express Architecture PCI Express 3.0

Phase 0

Phase 0 includes the following steps:

1. Upstream component enters Phase 0 of equalization during Recovery.Rcvrconfig by sending EQ TS2 training sets with starting presets for the downstream component. EQ TS2 training sets may be sent at 2.5 GT/s or 5 GT/s.
2. The downstream component enters Phase 0 of equalization after exiting Recovery.Speed at 8 GT/s. It receives the starting presets from the training sequences and applies them to its transmitter. At this time, upstream component has entered Phase 1 and is operating at 8 GT/s.
3. To move to Phase 1, the receiver must have a BER < 10^{-4} and should be able to decode enough consecutive training sequences.
4. The downstream component must detect training sets with Equalization Control (EC) bits set to 2'b01 in order to move to EQ Phase 1.

Phase 1

During Phase 1 of equalization process, the link partners exchange FS (Full Swing) and LF (Low Frequency) information. These values represent the upper and lower bounds for the TX coefficients. The receiver uses this information to calculate and request the next set of transmitter coefficients.

1. Once training sets with EC bits set to 1'b0 are captured on all lanes, the upstream component moves to EQ Phase 2 sending EC=2'b10 along with starting pre-cursor, main cursor, and post-cursor coefficients.
2. The downstream component detects these new training sets, and moves to EQ Phase 2.

Phase 2 (Optional)

This section describes the (optional) Phase 2.

During Phase 2, the Endpoint tunes the TX coefficients of the Root Port. The TS1 Use Preset bit determines whether the Endpoint uses presets for coarse resolution or coefficients for fine resolution.

Note: If you are using the PHY IP Core for PCI Express (PIPE) PCI Express as an Endpoint, you cannot perform Phase 2 tuning. The PIPE interface does not provide any measurement metric to the Root Port to guide coefficient preset decision making. The Root Port should reflect the existing coefficients and move to the next phase. The default Full Swing (FS) value advertized by Altera device is 40 and Low Frequency (LF) is 13.

If you are using the PHY IP Core for PCI Express (PIPE) PCI Express as Root Port, the End Point can tune the Root Port TX coefficients.

The tuning sequence typically includes the following steps:

1. The Endpoint receives the starting presets from the Phase 2 training sets sent by the Root Port.
2. The circuitry in the Endpoint receiver determines the BER and calculates the next set of transmitter coefficients using FS and LF and embeds this information in the Training Sets for the Link Partner to apply to its transmitter.

   The Root Port decodes these coefficients and presets, performs legality checks for the three transmitter coefficient rules and applies the settings to its transmitter and also sends them in the Training Sets. Three rules for transmitter coefficients are:

   a. $|C_{-1}| <= \text{Floor } (FS/4)$
   b. $|C_{-1}|+C_0+|C_{+1}| = FS$
   c. $C_0-|C_{-1}|-|C_{+1}| >= LF$

   Where:

   C_0 is the main cursor (boost)

   C_{-1} is the pre-cursor (pre shoot)

   C_{+1} is the post-cursor (de emphasis)
3. This process is repeated until the downstream component's receiver achieves a BER of $< 10^{-12}$.

Phase 3 (Optional)

This section describes the (optional) Phase 3.

During this phase, the Root Port tunes the Endpoint's transmitter. This process is analogous to Phase 2 but operates in the opposite direction.

Note: If you are using the PHY IP Core for PCI Express (PIPE) PCI Express as a Root Port, you cannot perform Phase 3 tuning.

Once Phase 3 tuning is complete, the Root Port moves to Recovery.RcvrLock, sending EC=2'b00, along with the final coefficients or preset agreed upon in Phase 2. The Endpoint moves to Recovery.RcvrLock using the final coefficients or preset agreed upon in Phase 3.

Recommendations for Tuning Link Partner's Transmitter

This section describes tuning link partner's transmitter.

To improve the BER of the StratixV receiver, Altera recommends that you turn on Adaptive Equalization (AEQ) one-time mode during Phase 2 Equalization for Endpoints or Phase 3 Equalization for Root Ports. You enable AEQ through the Transceiver Reconfiguration Controller IP Core. For more information about this component, refer to Transceiver Reconfiguration Controller IP Core. For more information about running AEQ, refer to AEQ Registers.

Note: AEQ must be turned off while switching from Gen3 to Gen1 or from Gen3 to Gen2.

Enabling Dynamic PMA Tuning for PCIe Gen3

"Section 4.2.3 Link Equalization Procedure for 8.0 GT/s Data Rate" in the PCI Express Base Specification, Rev. 3.0 provides detailed information about the four-stage link equalization procedure. However, in some

instances you may want to override the specified four-stage link equalization procedure to dynamically tune PMA settings. Follow these steps to override Gen3 equalization:

1. Connect the Transceiver Reconfiguration Controller IP Core to your PHY IP Core for PCI Express as shown in PCI Express PIPE IP Core Top-Level Modules.
2. For each transmitter port, use the Intel Quartus Prime Assignment Editor to assign the **Transmitter VOD/Preemphasis Control Source** the value **RAM_CTL**.
3. Recompile your design.

You can now use the Transceiver Reconfiguration Controller to change VOD and pre-emphasis settings.

Related Information

PHY Interface for the PCI Express Architecture PCI Express 3.0

PHY for PCIe (PIPE) Dynamic Reconfiguration

Dynamic reconfiguration calibrates each channel to compensate for variations due to process, voltage, and temperature (PVT).

For Stratix V devices, each channel and each TX PLL have separate dynamic reconfiguration interfaces. The MegaWizard Plug-In Manager provides informational messages on the connectivity of these interfaces. The following example shows the messages for a 8-channel PHY IP Core for PCI Express (PIPE).

Although you must initially create a separate reconfiguration interface for each channel and TX PLL in your design, when the Intel Quartus Prime software compiles your design, it reduces the total number of reconfiguration interfaces by merging reconfiguration interfaces. The synthesized design typically includes a reconfiguration interface for at least three channels because the three channels within each transceiver triplet share a single physical Avalon-MM slave interface which connects to the Transceiver Reconfiguration Controller IP Core. Conversely, you cannot connect the three channels that share this single physical Avalon-MM interface to different Transceiver Reconfiguration Controllers. Doing so causes a Fitter error. For more information, refer to“Transceiver Reconfiguration Controller to PHY IP Connectivity”.

Example 9-1: Informational Messages for the Transceiver Reconfiguration Interface

```
PHY IP will require 9 reconfiguration interfaces for connection to the
external reconfiguration controller.

Reconfiguration interface offsets 0-7 are connected to the transceiver
channels.

Reconfiguration interface offset 8 is connected to the transmit PLL.
```

The reconfiguration interface uses the Avalon-MM PHY Management interface clock.

Table 9-12: Reconfiguration Interface Signals

Signal Name	Direction	Description
`reconfig_to_xcvr [<r>70-1:0]`	Input	Reconfiguration signals from the Transceiver Reconfiguration Controller. <*r*> grows linearly with the number of reconfiguration interfaces.
`reconfig_from_xcvr [<r>46-1:0]`	Output	Reconfiguration signals to the Transceiver Reconfiguration Controller. <*r*> grows linearly with the number of reconfiguration interfaces.

Logical Lane Assignment Restriction

If you are using ×6 or ×N bonding, transceiver dynamic reconfiguration requires that you assign the starting channel number.

Transceiver dynamic reconfiguration requires that you assign the starting channel number. For PCIe ×8 configurations, logical channel 0 must be assigned to either physical transceiver channel 1 or channel 4 of a transceiver bank. For PCIe x4 configurations, logical channel 1 must be assigned to either physical transceiver channel 1 or channel 4 of a transceiver bank. However, if you have already created a PCB with a different lane assignment for PCIe ×8 logical lane 0 or PCIe ×4 logical lane 1, you can use the workaound shown in the example below to remove this restriction; the example redefines the `pma_bonding_master` parameter using the Intel Quartus Prime Assignment Editor. In this example, the `pma_bonding_master` was originally assigned to physical channel 1. (The original assignment could also have been to physical channel 4.) The to parameter reassigns the `pma_bonding_master` to the PHY IP Core for PCI Express (PIPE) instance name. You must substitute the instance name from your design for the instance name shown in quotation marks

Example 9-2: Overriding Logical Channel 0 Channel Assignment Restrictions in Stratix V Devices for ×6 or ×N Bonding

```
set_parameter -name pma_bonding_master "\"1\"" -to "<PHY IP instance name>"
```

PHY for PCIe (PIPE) Simulation Files and Example Testbench

Refer to Running a Simulation Testbench for a description of the directories and files that the Intel Quartus Prime software creates automatically when you generate your PHY IP Core for PCI Express.

Refer to the Intel FPGA Wiki for an example testbench that you can use as a starting point in creating your own verification environment.

Related Information

Intel FPGA Wiki

Custom PHY IP Core 10

2020.06.02

UG-01080 Subscribe Send Feedback

The Altera Custom PHY IP Core is a generic PHY that you can customize for use in Arria V, Cyclone V, or Stratix V FPGAs. You can connect your application's MAC-layer logic to the Custom PHY to transmit and receive data at rates of 0.611–6.5536 Gbps for Arria V GX devices, 0.611–6.5536 Gbps in Arria V GT devices, 0.622–9.8304 Gbps in Arria V GZ devices, 0.611–3.125 Gbps for Cyclone V GX devices, 0.611–5.000 Gbps for Cyclone V GT devices, and 0.622–11.0 Gbps for Stratix V devices. You can parameterize the physical coding sublayer (PCS) to include the functions that your application requires.

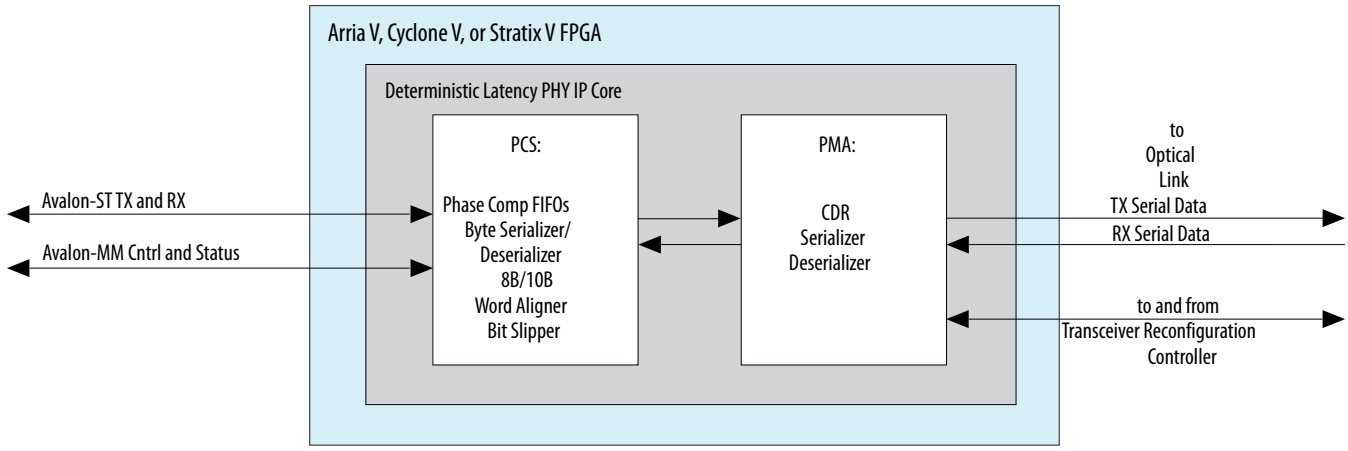
The following functions are available:

- 8B/10B encode and decode
- Three word alignment modes
- Rate matching
- Byte ordering

By setting the appropriate options using the MegaWizard Plug-In Manager, you can configure the Custom PHY IP Core to support many standard protocols, including all of the following protocols:

- Serial Data Converter (SDC(JESD204A))
- Serial digital interface (SDI)
- Ethernet (1.25 and 2.50 Gbps)
- Serial RapidIO ® (SRIO) 1.3
- Serial ATA (SATA) and serial attached SCSI (SAS) Gen1, Gen2, and Gen3
- Gigabit-capable passive optical network (GPON)

Figure 10-1: Custom PHY IP Core



Related Information

- **To access control and status registers in the Custom PHY, your design must include an embedded controller with an Avalon-MM master interface**
- **Transceiver Configurations in Stratix V Devices**

Device Family Support

IP cores provide either final or preliminary support for target Altera device families.

These terms have the following definitions:

- Final support—Verified with final timing models for this device.
- Preliminary support—Verified with preliminary timing models for this device.

Table 10-1: Device Family Support

Device Family	Support
Arria V devices-Hard PCS and PMA	Final
Cyclone V devices-Hard PCS and PMA	Final
Stratix V devices-Hard PCS and PMA	Final
Other device families	No support

Performance and Resource Utilization

Because the PCS and PMA are both implemented in hard logic, the Custom PHY IP Core requires less than 1% of FPGA resources.

Table 10-2: Custom PHY IP Core Performance and Resource Utilization—Stratix V GT Device

Channels	Combinational ALUTs	Logic Registers (Bits)
1	142	154
4	244	364

Parameterizing the Custom PHY

Complete the following steps to configure the Custom PHY IP Core:

1. Under **Tools** > **IP Catalog**, select the device family of your choice.
2. Under **Tools** > **IP Catalog** > **Interfaces** > **Transceiver PHY**, select **Custom PHY** .
3. Use the tabs on the MegaWizard Plug-In Manager to select the options required for the protocol.
4. Refer to the following topics to learn more about the parameters:
   a. **General Parameters**
   b. **Word Alignment Parameters** on page 10-7
   c. **Rate Match FIFO Parameters** on page 10-10
   d. **Rate Match FIFO Parameters** on page 10-10**8B/10B Encoder and Decoder Parameters** on page 10-11
   e. **Byte Order Parameters** on page 10-12
   f. **PLL Reconfiguration Parameters** on page 10-15
   g. **Analog Parameters** on page 10-16
5. Click Finish to generate your parameterized Custom PHY IP Core.

General Options Parameters

The General Options tab allows you to set the basic parameters of your transceiver PHY.

Table 10-3: Table 9-3. Custom PHY General Options

Name	Value	Description
Device family	**Arria V** **Cyclone V** **Stratix V**	Specifies the device family. Arria V, Cyclone V, and Stratix V are available.
Parameter validation rules	**Custom GIGE**	Allows you to specify the transceiver protocol. Select **Custom** if you are not implementing 1.25 or 2.50GIGE.
Mode of operation	**Duplex TX RX**	You can select to transmit data, receive data, or both.
Number of lanes	**1-32**	The total number of lanes in each direction.

Name	Value	Description
Enable lane bonding	**On/Off**	When enabled, a single clock drives multiple lanes, reducing clock skew. In Stratix V devices, up to 6 lanes can be bonded if you use an ATX PLL; 4 lanes can be bonded if you select the CMU PLL.
Bonding mode	**Non-bonded or x1** **Bonded or xN** **fb_compensation**	Select **Non-bonded or x1** to use separate clock sources for each channel. (This option is available for Cyclone V and Arria V devices.) If one PLL drives multiple channels, PLL merging is required. During compilation, the Intel Quartus Prime Fitter, merges all the PLLs that meet PLL merging requirements. Refer to **Merging TX PLLs In Multiple Transceiver PHY Instances** on page 17-58 to observe PLL merging rules. Select **Bonded or xN** to use the same clock source for up to 6 channels in a single transceiver bank, resulting in reduced clock skew. You must use contiguous channels when you select **×N** bonding. In addition, you must place logical channel 0 in either physical channel 1 or 4. Physical channels 1 and 4 are indirect drivers of the **×N** clock network. Select **fb_compensation** (feedback compensation) to use the same clock source for multiple channels across different transceiver banks to reduce clock skew. (This option is only available for Stratix V devices.) For more information about bonding, refer to "Transmitter Clock Network" in **Transceiver Clocking in Arria V Devices** in volume 2 of the Arria V Device Handbook. For more information about bonding, refer to "Transmitter Clock Network" in **Transceiver Clocking in Cyclone V Devices** in volume 2 of the Cyclone V Device Handbook. For more information about bonding, refer to "Bonded Channel Configurations Using the PLL Feedback Compensation Path" in **Transceiver Clocking in Stratix V Devices** in volume 2 of the Stratix V Device Handbook.
FPGA fabric transceiver interface width	**8,10,16,20, 32,40**	Specifies the total serialization factor, from an input or output pin to the MAC-layer logic.

<table>
<tr><th>Name</th><th>Value</th><th>Description</th></tr>
<tr><td>PCS-PMA interface width</td><td>8, 10, 16, 20</td><td>The PCS-PMA interface width depends on the FPGA fabric transceiver interface width and whether 8B/10B is enabled. The following combinations are available:
<table>
<tr><th>FPGA/XCVR</th><th>8B/10B</th><th>PMA Interface Width</th></tr>
<tr><td>8</td><td>No</td><td>8</td></tr>
<tr><td>8</td><td>Yes</td><td>10</td></tr>
<tr><td>10</td><td>No</td><td>10</td></tr>
<tr><td>16</td><td>No</td><td>8 or 16</td></tr>
<tr><td>16</td><td>Yes</td><td>10 or 20</td></tr>
<tr><td>20</td><td>No</td><td>10 or 20</td></tr>
<tr><td>32</td><td>No</td><td>16</td></tr>
<tr><td>32</td><td>Yes</td><td>20</td></tr>
<tr><td>40</td><td>No</td><td>20</td></tr>
</table></td></tr>
<tr><td>PLL type</td><td>CMU
ATX</td><td>The CMU PLL is available for Arria V and Cyclone V devices.
For Stratix V devices, you can select either the CMU or ATX PLL. The CMU PLL has a larger frequency range than the ATX PLL. The ATX PLL is designed to improve jitter performance and achieves lower channel-to-channel skew; however, it supports a narrower range of data rates and reference clock frequencies. Another advantage of the ATX PLL is that it does not use a transceiver channel, while the CMU PLL does.
Because the CMU PLL is more versatile, it is specified as the default setting. An informational message displays in the message pane telling you whether the chosen settings for Data rate and Input clock frequency are legal for the CMU PLL, or for both the CMU and ATX PLLs.</td></tr>
<tr><td>Data rate</td><td>622-11000 Mbps</td><td>Specifies the data rate. The possible data rates depend upon the device and configuration specified.</td></tr>
</table>

Name	Value	Description
Base data rate	**1 × Data rate** **2 × Data rate** **4 × Data rate**	The **base data rate** is the frequency of the clock input to the PLL. Select a **base data rate** that minimizes the number of PLLs required to generate all the clocks required for data transmission. By selecting an appropriate **base data rate**, you can change data rates by changing the divider used by the clock generation block. For higher frequency data rates 2 × and 4× base data rates are not available.
Input clock frequency	Variable	Specifies the frequency of the PLL input reference clock.
Additional Options		
Enable TX Bitslip	**On/Off**	When enabled, the TX bitslip word aligner is operational.
Create rx_coreclkin port	**On/Off**	This is an optional clock to drive the coreclk of the RX PCS
Create tx_coreclkin port	**On/Off**	This is an optional clock to drive the coreclk of the TX PCS
Create rx_recovered_clk port	**On/Off**	When enabled, the RX recovered clock is an output.
Create optional ports	**On/Off**	When you turn this option on, the following signals are added to the top level of your transceiver for each lane: • `tx_forceelecidle` • `rx_is_lockedtoref` • `rx_is_lockedtodata` • `rx_signaldetect`
Enable Avalon data interfaces and bit reversal	**On/Off**	When you turn this option **On**, the order of symbols is changed. This option is typically required if you are planning to import your Custom PHY IP Core into a Qsys system.

Name	Value	Description
Enable embedded reset control	**On/Off**	When **On**, the automatic reset controller initiates the reset sequence for the transceiver. When **Off** you can design your own reset logic using `tx_analogreset` , `rx_analogreset`, `tx_digitalreset`, `rx_digital-reset`, and `pll_powerdown` which are top-level ports of the Custom Transceiver PHY. You may also use the Transceiver PHY Reset Controller' to reset the transceivers. For more information, refer to the Transceiver Reconfiguration Controller IP Core . By default, the CDR circuitry is in automatic lock mode whether you use the embedded reset controller or design your own reset logic. You can switch the CDR to manual mode by writing the `pma_rx_setlockto-data` or `pma_rx_set_locktoref` registers to 1. If either the `pma_rx_set_locktodata` and `pma_rx_set_locktoref` is set, the CDR automatic lock mode is disabled.

Table 10-4: Reset Mode

The CDR can be put in either manual or automatic mode. The CDR mode is controlled with the `pma_rx_set_locktodata` and `pma_rx_set_locktoref` registers. This table shows the required settings to control the CDR mode.

rx_set_locktoref	rx_set_locktodata	CDR Lock Mode
1	0	Manual RX CDR locked to reference
X	1	Manual RX CDR locked to data
0	0	Automatic RX CDR

Related Information

Transceiver Reset Control in Stratix V Devices

Word Alignment Parameters

The word aligner restores word boundaries of received data based on a predefined alignment pattern. This pattern can be 7, 8, 10, 16, 20, or 32 bits long. The word alignment module searches for a programmed pattern to identify the correct boundary for the incoming stream.

Table 10-5: Word Aligner Options

Name	Value	Description
Word alignment mode	**Manual**	In this mode you enable the word alignment function by asserting `rx_enapatternalign` using the Avalon-MM interface. When the PCS exits reset, the word aligner automatically performs an initial alignment to the specified word alignment pattern when the interface between the PCS and PMA is 16 or 20 bits. For other cases, you must assert `rx_enapatternalign` to initiate another pattern alignment. `rx_enapatternalign` is edge sensitive in most cases; however, if the PMA-PCS interface width is 10 bits, it is level sensitive.
	Bit slipping	You can use bit slip mode to shift the word boundary using the Avalon-MM interface. For every rising edge of the rx_bitslip signal, the word boundary is shifted by 1 bit. Each bit slip removes the earliest received bit from the received data.

Name	Value	Description
Word alignment mode	**Automatic synchronization state machine**	In this mode, word alignment is controlled by a programmable state machine. This mode can only be used with 8B/10B encoding. The data width at the word aligner can be 10 or 20 bits. You can specify the following parameters: • **Number of consecutive valid words before sync state is reached:** Specifies the number of consecutive valid words needed to reduce the built up error count by 1. Valid values are 1-256. • **Number of bad data words before loss of sync state:** Specifies the number of bad data words required for alignment state machine to enter loss of sync state. Valid values are 1-256. • **Number of valid patterns before sync state is reached:** Specifies the number of consecutive patterns required to achieve synchronization. Valid values are 1-256. **Create optional word aligner status ports**: When enabled the `rx_syncstatus` and `rx_patterndetect` status ports are created. • **Word alignment pattern length:** Allows you to specify a 7-, 10-, or 20-bit pattern for use in the word alignment state machine. The 20-bit pattern is available when the PMA-PCS interface width is 20 bits. • **Word alignment pattern:** Allows you to specify a word alignment pattern.
Enable run length violation checking	**On/Off**	If you turn this option on, you can specify the run length which is the maximum legal number of contiguous 0s or 1s.
Run length	**40-640**	Specifies the threshold for a run-length violation.

Table 10-6: More Information about Word Aligner Functions

PMA-PCS Interface Width (bits)	Word Alignment Mode	Word Alignment Pattern Length (bits)	Word Alignment Behavior
8	**Manual alignment**	**8, 16**	User-controlled signal starts alignment process. Alignment occurs once unless signal is re-asserted.
10	**Manual alignment**	**7 , 10**	User-controlled signal starts alignment process. Alignment occurs once unless signal is re-asserted.
	Automatic synchronized state machine		Data must be 8B/10B encoded and aligns to selected word aligner pattern.
16	**Manual alignment**	**8 , 16, 32**	User-controlled signal starts alignment process. Alignment occurs once unless signal is re-asserted.
20	**Manual alignment**	**7, 10, 20, 40**	User-controlled signal starts alignment process. Alignment occurs once unless signal is re-asserted.
	Automatic Synchronized State Machine	**7, 10, 20**	Automatically selected word aligner pattern length and pattern.

Related Information

- **Transceiver Architecture in Stratix V Devices**
- **Transceiver Architecture in Arria V Devices**
- **Transceiver Architecture in Cyclone V Devices**

Rate Match FIFO Parameters

The rate match FIFO compensates for small clock frequency differences between the upstream transmitter and the local receiver clocks by inserting or removing skip (SKP) symbols or ordered-sets from the inter-packet gap (IPG) or idle streams. It deletes SKP symbols or ordered-sets when the upstream transmitter reference clock frequency is greater than the local receiver reference clock frequency. It inserts SKP symbols or ordered-sets when the local receiver reference clock frequency is greater than the upstream transmitter reference clock frequency.

If you enable the rate match FIFO, the MegaWizard Plug-In Manager provides options to enter the rate match insertion and deletion patterns. The lower 10 bits are the control pattern, and the upper 10 bits are the skip pattern.

Table 10-7: Rate Match FIFO Options

Name	Value	Description
Enable rate match FIFO	**On/Off**	Turn this option on, to enable the rate match functionality. Turning this option on adds the `rx_rmfifodatainserted`, and `rx_rmfifodatadeleted` status signals to your PHY.
Rate match insertion/deletion +ve disparity pattern	1101000011 1010000011	Enter a 10-bit skip pattern (bits 10-19) and a 10-bit control pattern (bits 0-9). The skip pattern must have neutral disparity.
Rate match insertion/deletion -ve disparity pattern	0010111100 0101111100	Enter a 10-bit skip pattern (bits 10-19) and a 10-bit control pattern (bits 0-9). The skip pattern must have neutral disparity.
Create optional rate match FIFO status ports	**On/Off**	When enabled, creates the `rx_rmfifoddatainserted` and `rx_rmfifodatadeleted` signals from the rate match FIFO become output ports.

Note: If you have the auto-negotiation state machine in your transceiver design, please note that the rate match FIFO is capable of inserting or deleting the first two bytes (K28.5//D2.2) of /C2/ ordered sets during auto-negotiation. However, the insertion or deletion of the first two bytes of /C2/ ordered sets can cause the auto-negotiation link to fail. For more information, visit **Altera Knowledge Base Support Solution**.

8B/10B Encoder and Decoder Parameters

The 8B/10B encoder generates 10-bit code groups (control or data word) with proper disparity from the 8-bit data and 1-bit control identifier. The 8B/10B decoder receives 10-bit data from the rate matcher and decodes it into an 8-bit data and 1-bit control identifier.

Table 10-8: 8B/10B Options

Name	Value	Description
Enable 8B/10B decoder/encoder	**On/Off**	Enable this option if your application requires 8B/10B encoding and decoding. This option on adds the `tx_datak <n>`, `rx_datak <n>`, and `rx_runningdisp <n>` signals to your transceiver.

Name	Value	Description
Enable manual disparity control	**On/Off**	When enabled, you can use the `tx_forcedisp` signal to control the disparity of the 8B/10B encoder. Turning this option on adds the `tx_forcedisp` and `tx_dispval` signals to your transceiver.
Create optional 8B/10B status port	**On/Off**	Enable this option to include the 8B/10B `rx_errdetect` and `rx_disperr` error signals at the top level of the Custom PHY IP Core.

Byte Order Parameters

The byte ordering block is available when the PCS width is doubled at the byte deserializer. Byte ordering identifies the first byte of a packet by determining whether the programmed start-of-packet (SOP) pattern is present; it inserts enough pad characters in the data stream to force the SOP to the lowest order byte lane.

Note: You cannot enable Rate Match FIFO when your application requires byte ordering. Because the rate match function inserts and deletes idle characters, it may shift the SOP to a different byte lane.

Table 10-9: Byte Order Options

Name	Value	Description

Enable byte ordering block	**On/Off**	Turn this option on if your application uses serialization to create a datapath that is larger than 1 symbol. This option is only available if you use the byte deserializer for the following configurations:
		• Configuration 1:
		  • 16-bit FPGA fabric-transceiver interface
		  • No 8B/10B decoder (8-bit PMA-PCS interface)
		  • Word aligner in manual alignment mode
		• Configuration 2:
		  • 16-bit FPGA fabric-transceiver interface
		  • 8B/10B decoder (10-bit PMA-PCS interface)
		  • Word aligner in automatic synchronization state machine mode
		• Configuration 3:
		  • 32-bit FPGA fabric-transceiver interface
		  • No 8B/10B decoder (16-bit PMA-PCS interface)
		  • Word aligner in manual alignment mode
		• Configuration 4:
		  • 32-bit FPGA fabric-transceiver interface
		  • 8B/10B decoder (20-bit PMA-PCS interface)
		  • Word aligner in manual alignment mode
		• Configuration 5:
		  • 40-bit FPGA fabric-transceiver interface
		  • No 8B/10B decoder (20-bit PMA-PCS interface)
		  • Word aligner in manual alignment mode
		This option creates the `rx_byteordflag` signal which is asserted when the received data is aligned to the byte order pattern that you specified.

Enable byte ordering block manual control	**On/Off**	Turn this option on to choose manual control of byte ordering. This option creates the `rx_enabyteord` signal. A byte ordering operation occurs whenever `rx_enabyteord` is asserted. To perform multiple byte ordering operations, deassert and reassert `rx_enabyteord`.

<table>
<tr><th>Name</th><th>Value</th><th>Description</th></tr>
<tr><td>Byte ordering pattern</td><td>Depends on configuration</td><td>Specifies the pattern that identifies the SOP. For 16-bit byte ordering pattern you must include a 2-bit pad so that the pattern entered is in the following format: 00 <<i>pattern</i>> 00 <<i>pattern</i>>. For example, if the required pattern is 10111100, enter the following pattern: 0010111100001011100

Enter the byte ordering pattern as follows based on the 5 configurations that support byte ordering as described in the Enable byte ordering block:

• Configuration 1: 8-bits
• Configuration 2: 10-bits

For example: If you select a /Kx.y/ control code group as the byte ordering pattern, the most significant 2 bits of the 10-bit byte ordering pattern must be 2'b01. If you select a /Dx.y/ data code group as the byte ordering pattern, the most significant 2-bits of the 10-bit byte ordering pattern must be 2'b00. The least significant 8-bits must be the 8B/10B decoded version of the code group used for byte ordering.
• Configuration 3:16-bits, 8-bits
• Configuration 4: 18-bits
• Configuration 5: 20-bits, 10-bits

For example: If you select a /Kx.y/Dx.y/ code group as the byte ordering pattern, the most significant 2-bits of the 20-bit byte ordering pattern must be 2'b01. Similarly bit[9:0] must be 2'b00. Bit[18:10] must be the 8B/10B decoded version of /Kx.y/. Bit[7:0] must be 8B/10B decoded version of /Dx.y/.</td></tr>
<tr><td>Byte ordering pad pattern</td><td>00000000</td><td>Specifies the pad pattern that is inserted to align the SOP. Enter the following size pad patterns:

<table>
<tr><th>Data Width</th><th>8B/10B Encoded?</th><th>Pad Pattern</th></tr>
<tr><td>8, 16, 32</td><td>No</td><td>8 bits</td></tr>
<tr><td>10, 20, 40</td><td>No</td><td>10 bits</td></tr>
<tr><td>8, 16, 3</td><td>No</td><td>9 bits</td></tr>
</table>
</td></tr>
</table>

PLL Reconfiguration Parameters

Table 10-10: PLL Reconfigurations

Name	Value	Description
Allow PLL Reconfiguration	**On/Off**	You must enable this option if you plan to reconfigure the PLLs in your design. This option is also required to simulate PLL reconfiguration.
Number of TX PLLs	**1-4**	Specifies the number of TX PLLs that can be used to dynamically reconfigure channels to run at multiple data rates. If your design does not require transceiver TX PLL dynamic reconfiguration, set this value to 1. The number of actual physical PLLs that are implemented depends on the selected clock network. Each channel can dynamically select between n PLLs, where n is the number of PLLs specified for this parameter. You must disable the embedded reset controller and design your own controlled reset controller or the use the highly configurable reset core described in "Transceiver Reconfiguration Controller IP Core" if you intend to use more than 1 TX PLL for a Custom PHY IP instance. **Note:** For more details, refer to the *Transceiver Clocking* chapter in the device handbook for the device family you are using.
Number of reference clocks	**1-5**	Specifies the number of input reference clocks. More than one reference clock may be required if your design reconfigures channels to run at multiple frequencies.
Main TX PLL logical index	**0-3**	Specifies the index for the TX PLL that should be instantiated at startup. Logical index 0 corresponds to TX PLL0, and so on.
CDR PLL input clock source	**0-3**	Specifies the index for the CDR PLL input clock that should be instantiated at startup. Logical index 0 corresponds to input clock 0 and so on.
TX PLL (0-3)		

Name	Value	Description
PLL Type	**CMU** **ATX**	Specifies the PLL type.
PLL base data rate	**1 × Lane rate** **2 × Lane rate** **4 × Lane rate**	Specifies **Base data rate**.
Reference clock frequency	Variable	Specifies the frequency of the PLL input reference clock. The PLL must generate an output frequency that equals the **Base data rate**/2. You can use any **Input clock frequency** that allows the PLLs to generate this output frequency.
Selected reference clock source	**0-4**	Specifies the index of the input clock for this TX PLL. Logical index 0 corresponds to input clock 0 and so on.
Channel Interface		
Enable channel interface	**On/Off**	Turn this option on to enable PLL and datapath dynamic reconfiguration. When you select this option, the width of `tx_parallel_data` and `rx_parallel_data` buses increases in the following way. • n The `tx_parallel_data` bus is 44 bits per lane; however, only the low-order number of bits specified by the **FPGA fabric transceiver interface width** contain valid data for each lane. • n The `rx_parallel_data` bus is 64 bits per lane; however, only the low-order number of bits specified by the **FPGA fabric transceiver interface width** contain valid data.

Related Information

General Options Parameters on page 10-3

Analog Parameters

Click the appropriate link to specify the analog options for your device:

Related Information

- **Analog Settings for Arria V Devices** on page 20-2

- **Analog Settings for Arria V GZ Devices** on page 20-11
- **Analog Settings for Cyclone V Devices** on page 20-26
- **Analog Settings for Stratix V Devices** on page 20-35

Presets for Ethernet

Presets allow you to specify a group of parameters to implement a particular protocol or application. If you apply the presets for GIGE-1.25 Gbps or GIGE–2.5 Gbps, parameters with specific required values for those protocols are set for you. Selecting a preset does not prevent you from changing any parameter to meet the requirements of your design.

Table 10-11: Presets for Ethernet Protocol

Parameter Name	GIGE-1.25 Gbps	GIGE-2.50 Gbps
General Options Tab		
Parameter validation rules	GIGE	GIGE
Enable bonding	Off	Off
FPGA fabric transceiver interface width	8	16
PCS-PMA Interface Width	10	10
Data rate	1250 Mbps	3125 Mbps
Input clock frequency	62.5 MHz	62.5 MHz
Enable TX Bitslip	Off	Off
Create rx_coreclkin port	Off	Off
Create tx_coreclkin port	Off	Off
Create rx_recovered_clk port	Off	Off
Create optional ports	Off	Off
Avalon data interfaces	Off	Off
Enabled embedded reset controller	On	On
Word Aligner Options		
Word alignment mode	Automatic synchronization state machine	Automatic synchronization state machine
Number of consecutive valid words before sync state is reached	3	3
Number of bad data words before loss of sync state	3	3
Number of valid patterns before sync state is reached	3	3

Parameter Name	GIGE-1.25 Gbps	GIGE-2.50 Gbps
Create optional word aligner status ports	Off	Off
Word aligner pattern length	10	10
Word alignment pattern	0101111100	0101111100
Enable run length violation checking	Off	Off
Run length	-	-
Rate Match Options		
Enable rate match FIFO	On	On
Rate match insertion/deletion +ve disparity pattern	10100010010101111100	10100010010101111100
Rate match insertion/deletion -ve disparity pattern	10101011011010000011	10101011011010000011
8B/10B Options		
Enable 8B/10B decoder/encoder	On	On
Enable manual disparity control	Off	Off
Create optional 8B/10B status port	Off	Off
Byte Order Options		
Enable byte ordering block	Off	Off
Enable byte ordering block manual control	Off	Off
Byte ordering pattern	-	-
Byte ordering pad pattern	-	-

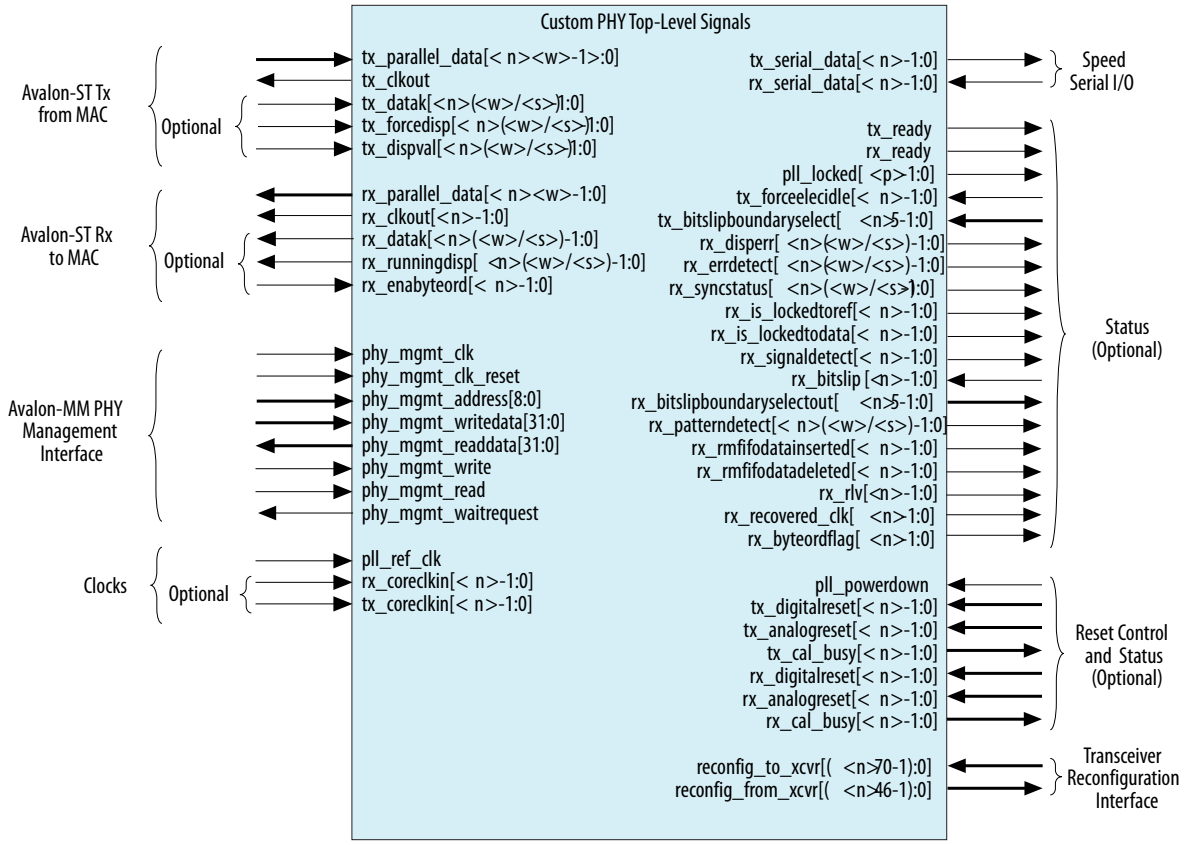
Interfaces

Figure 10-2: Custom PHY Top-Level Signals

The variables in Figure 9–2 represent the following parameters:

- <*n*>—The number of lanes
- <*w*>—The width of the FPGA fabric to transceiver interface per lane
- <*s*>— The symbol size
- <*p*>—The number of PLLs

Figure 10-3: Custom PHY Interfaces



Note: By default **block diagram** shown in the MegaWizard Plug-In Manager labels the external pins with the interface type and places the *interface name* inside the box. The interface type and name are used in the **_hw.tcl** file that describes the component. If you turn on **Show signals**, the **block diagram** displays all top-level signal names.

Related Information

Component Interface Tcl Reference

Data Interfaces

This topic describes the Avalon-ST TX and RX interface signals as well as the serial interface and status signals.

Table 10-12: Avalon-ST TX Interface Signals

Signal Name	Direction	Description

`tx_parallel_data[(<n> 43:0]`	Input	This is TX parallel data driven from the MAC. The ready latency on this interface is 0, so that the PHY must be able to accept data as soon as it comes out of reset.
		The bits of each 11-bit word have the following definitions when you enable 8B/10B encoding:
		• `tx_parallel_data[7:0]`: TX data bus.
		• `tx_parallel_data[8]`: TX data control character.
		• `tx_parallel_data[9]`: Force disparity. For the Gen1 and Gen2 PCIe PIPE interface, this signal forces running disparity to negative in compliance mode.
		• `tx_parallel_data[10]`: Disparity field.
		  • 1'b0: Transmit positive disparity.
		  • 1'b1: Transmit negative disparity.
		  • For Gen1 and Gen2 PCIe PIPE - Forces the TX ouptu to electrical idle.
		If 8B/10B encoding is disabled, the width of this interface is width you specified for **FPGA fabric transceiver interface width** If 8B/10B encoding is disabled, when you have enabled dynamic reconfiguration, the following mapping applies to each word:
		• `tx_parallel_data[7:0]`: Data input bus.
		• `tx_parallel_data[10:8]`: Unused.
		Refer to Table 10-13 for the location of valid data for a single- and double-word data buses, with and without the byte serializer.

`tx_clkout`	Output	This is the clock for TX parallel data, control, and status signals.
`tx_datak[< n >(<w>/<s>)-1:0]`	Input	Data and control indicator for the transmitted data. When 0, indicates that tx_data is data, when 1, indicates that tx_ data is control.
`tx_forcedisp[< n >(<w>/<s>)-1:0]`	Input	When asserted, this control signal enables disparity to be forced on the TX channel. This signal is created if you turn **On** the **Enable manual disparity control** option on the **8B/10B** tab.

Signal Name	Direction	Description
tx_dispval[< n >(<w>/<s>)-1:0]	Input	This control signal specifies the disparity of the data. When 0, indicates positive disparity, when 1, indicates negative disparity. This port is created if you turn **On** the **Enable disparity control** option on the **8B/10B** tab.

Table 10-13: Location of Valid Data Words for tx_parallel_data for Various FPGA Fabric to PCS Parameterizations

The following table shows the valid 11-bit data words with and without the byte deserializer for single- and double-word FPGA fabric to PCS interface widths. The byte serializer allows the PCS to operate at twice the data width of the PMA . This feature allows the PCS to run at a lower frequency and accommodates a wider range of FPGA interface widths.

Configuration	Bus Used Bits
Single word data bus, byte deserializer disabled	[10:0] (word 0)
Single word data bus, byte serializer enabled	[32:22], [10:0] (words 0 and 2)
Double word data bus, byte serializer disabled	[21:0] (words 0 and 1)
Double word data bus, byte serializer enabled	[43:0] (words 0-3)

Table 10-14: Avalon-ST RX Interface Signals

These signals are driven from the PCS to the MAC. This is an Avalon source interface.

Signal Name	Direction	Description

rx_parallel_data[<*n*>63:0]	Output	This is RX parallel data driven from the Custom PHY IP Core. The ready latency on this interface is 0, so that the MAC must be able to accept data as soon as the PHY comes out of reset. Data driven from this interface is always valid.
		
		The bits of each 16-bit word have the following definitions when you enable 8B/10B decoding:
		
		• rx_parallel_data[7:0]: RX data bus
		• rx_parallel_data[8]: RX data control character
		• rx_parallel_data[9]: Code violation
		• rx_parallel_data[10]: Word alignment status
		• rx_parallel_data[11]: Disparity error
		• rx_parallel_data[12]: Pattern detect
		• rx_parallel_data[14:13]
		  • 2'b00: Normal data
		  • 2'b01: Deletion
		  • 2'b10: Insertion (or Underflow with 9'h1FE or 9'h1F7
		  • 2'b11: Overflow
		• rx_parallel_data[14:13]: Running disparity value
		
		If 8B/10B decoding is disabled, the width of this interface is width you specified for **FPGA fabric transceiver interface width**. If 8B/10B encoding is disabled, when you have enabled dynamic reconfiguration, the following mapping applies to each word:
		
		• rx_parallel_data[9:0]: RX data bus
		• rx_parallel_data[10]: Sync status
		• rx_parallel_data[11]: Disparity error
		• rx_parallel_data[12]: Pattern detect
		• rx_parallel_data[14:13]
		  • 2'b00: Normal data
		  • 2'b01: Deletion
		  • 2'b10: Insertion (or Underflow with 9'h1FE or 9'h1F7
		  • 2'b11: Overflow
		• rx_parallel_data[15]: Running disparity value
		
		Refer to **Table 10-15** for the location of valid data for a single- and double-word data buses, with and without the byte serializer.

rx_clkout[< *n* >-1:0]	Output	This is the clock for the RX parallel data source interface.

Signal Name	Direction	Description
rx_datak[< n >(<w>/<s>)-1:0]	Output	Data and control indicator for the source data. When 0, indicates that rx_parallel_data is data, when 1, indicates that rx_parallel_data is control.
rx_runningdisp[< n >(<w>/<s>)-1:0]	Output	This status signal indicates the disparity of the incoming data.
rx_enabyteord[< n >-1:0]	Input	This signal is created if you turn **On** the **Enable byte ordering block control** option on the **Byte Order** tab. A byte ordering operation occurs whenever rx_enabyteord is asserted. To perform multiple byte ordering operations, deassert and reassert rx_enabyteord.

Table 10-15: Location of Valid Data Words for rx_parallel_data for Various FPGA Fabric to PCS Parameterizations

The following table shows the valid 11-bit data words with and without the byte deserializer for single- and double-word FPGA fabric to PCS interface widths. The byte deserializer allows the PCS to operate at twice the data width of the PMA . This feature allows the PCS to run at a lower frequency and accommodates a wider range of FPGA interface widths.

Configuration	Location of rx_parallel_data
Single word data bus, byte deserializer disabled	[15:0] (word 0)
Single word data bus, byte serializer enabled	[47:32], [15:0] (words 0 and 2)
Double word data bus, byte serializer disabled	[31:0] (words 0 and 1)
Double word data bus, byte serializer enabled	[63:0] (words 0-3)

Table 10-16: Serial Interface and Status Signals

Signal Name	Direction	Signal Name
rx_serial_data[< n >-1:0]	Input	Receiver differential serial input data.
tx_serial_data[< n >-1:0]	Output	Transmitter differential serial output data.

Clock Interface

The input reference clock, pll_ref_clk, drives a PLL inside the PHY-layer block, and a PLL output clock, rx_clkout is used for all data, command, and status inputs and outputs.

Table 10-17: Clock Signals

Signal Name	Direction	Description
pll_ref_clk	Input	Reference clock for the PHY PLLs. Frequency range is 50-700 MHz.

Signal Name	Direction	Description
rx_coreclkin[<n>-1:0]	Input	This is an optional clock to drive the coreclk of the RX PCS.
tx_coreclkin[<n>-1:0]	Input	This is an optional clock to drive the coreclk of the TX PCS

Related Information

Data Interfaces on page 10-20

Optional Status Interface

This topic describes the optional status signals for the TX and RX interface.

Table 10-18: Serial Interface and Status Signals

Signal Name	Direction	Signal Name
tx_ready	Output	When asserted, indicates that the TX interface has exited the reset state and is ready to transmit.
rx_ready	Output	When asserted, indicates that the RX interface has exited the reset state and is ready to receive.
pll_locked[<p>-1:0]	Output	When asserted, indicates that the PLL is locked to the input reference clock.
tx_forceelecidle[<n>-1:0]	Input	When asserted, enables a circuit to detect a downstream receiver. It is used for the PCI Express protocol. This signal must be driven low when not in use because it causes the TX PMA to enter electrical idle mode and tristate the TX serial data signals.
tx_bitslipboundaryselect [<n>5-1:0]	Input	This signal is used for bit slip word alignment mode. It selects the number of bits that the TX block must slip to achieve a deterministic latency.
rx_disperr[<n>(<w>/<s>)-1:0]	Output	When asserted, indicates that the received 10-bit code or data group has a disparity error.
rx_errdetect[<n>(<w>/<s>)-1:0]	Output	When asserted, indicates that a received 10-bit code group has an 8B/10B code violation or disparity error.

Signal Name	Direction	Signal Name
rx_syncstatus[<n> (<w>/<s>)-1:0]	Output	Indicates presence or absence of synchronization on the RX interface. Asserted when word aligner identifies the word alignment pattern or synchronization code groups in the received data stream. This signal is optional.
rx_is_lockedtoref[<n> -1:0]	Output	Asserted when the receiver CDR is locked to the input reference clock. This signal is asynchronous. This signal is optional.
rx_is_lockedtodata[<n> -1:0]	Output	When asserted, the receiver CDR is in to lock-to-data mode. When deasserted, the receiver CDR lock mode depends on the rx_locktorefclk signal level. This signal is optional.
rx_signaldetect[<n> -1:0]	Output	Signal threshold detect indicator required for the PCI Express protocol. When asserted, it indicates that the signal present at the receiver input buffer is above the programmed signal detection threshold value.
rx_bitslip[<n> -1:0]	Input	Used for manual control of bit slipping. The word aligner slips a bit of the current word for every rising edge of this signal. This is an asynchronous input signal and inside there is a synchronizer to synchronize it with rx_pma_clk/rx_clkout.
rx_bitslipboundaryselectout [<n> 5-1:0]	Output	This signal is used for bit slip word alignment mode. It reports the number of bits that the RX block slipped to achieve a deterministic latency.
rx_patterndetect[<n>(<w>/<s>)-1:0]	Output	When asserted, indicates that the programmed word alignment pattern has been detected in the current word boundary.
rx_rmfifodatainserted[<n>-1:0]	Output	When asserted, indicates that the RX rate match block inserted an ||R|| column.

Signal Name	Direction	Signal Name
rx_rmfifodatadeleted[<n>-1:0]	Output	When asserted, indicates that the RX rate match block deleted an ||R|| column.
rx_rlv[<n> -1:0]	Output	When asserted, indicates a run length violation. Asserted if the number of consecutive 1s or 0s exceeds the number specified in the MegaWizard Plug-In Manager.
rx_recovered_clk[<n>-1:0]	Output	This is the RX clock which is recovered from the received data stream.
rx_byteordflag[<n>-1:0]	Output	This status flag is asserted high the received data is aligned to the byte order pattern that you specify.

Optional Reset Control and Status Interface

This topic describes the signals in the optional reset control and status interface. These signals are available if you do not enable the embedded reset controller.

Table 10-19: Avalon-ST RX Interface

Signal Name	Direction	Description
pll_powerdown	Input	When asserted, resets the TX PLL.
tx_digitalreset[<n>-1:0]	Input	When asserted, reset all blocks in the TX PCS. If your design includes bonded TX PCS channels, refer to *Timing Constraints for Reset Signals when Using Bonded PCS Channels for a SDC* constraint you must include in your design.
tx_analogreset[<n>-1:0]	Input	When asserted, resets all blocks in the TX PMA. **Note:** For Arria V devices, while compiling a multi-channel transceiver design, you will see a compile warning (12020) in Intel Quartus Prime software related to the signal width of tx_analogreset. You can safely ignore this warning. Also, per-channel TX analog reset is not supported in Intel Quartus Prime software. Channel 0 TX analog resets all the transceiver channels.

Signal Name	Direction	Description
tx_cal_busy[<*n*>-1:0]	Output	When asserted, indicates that the initial TX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP. You must hold the channel in reset until calibration completes.
rx_digitalreset[<n>-1:0]	Input	When asserted, resets the RX PCS.
rx_analogreset[<n>-1:0]	Input	When asserted, resets the RX CDR.
rx_cal_busy[<*n*>-1:0]	Output	When asserted, indicates that the initial RX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP.

Related Information

- **Timing Constraints for Bonded PCS and PMA Channels** on page 18-11
- **Transceiver Reset Control in Stratix V Devices**
- **Transceiver Reset Control in Arria V Devices**
- **Transceiver Reset Control in Cyclone V Devices**

Register Interface and Register Descriptions

The Avalon-MM PHY management interface provides access to the Custom PHY PCS and PMA registers, resets, error handling, and serial loopback controls. You can use an embedded controller acting as an Avalon-MM master to send read and write commands to this Avalon-MM slave interface.

Figure 10-4: Custom PHY IP Core

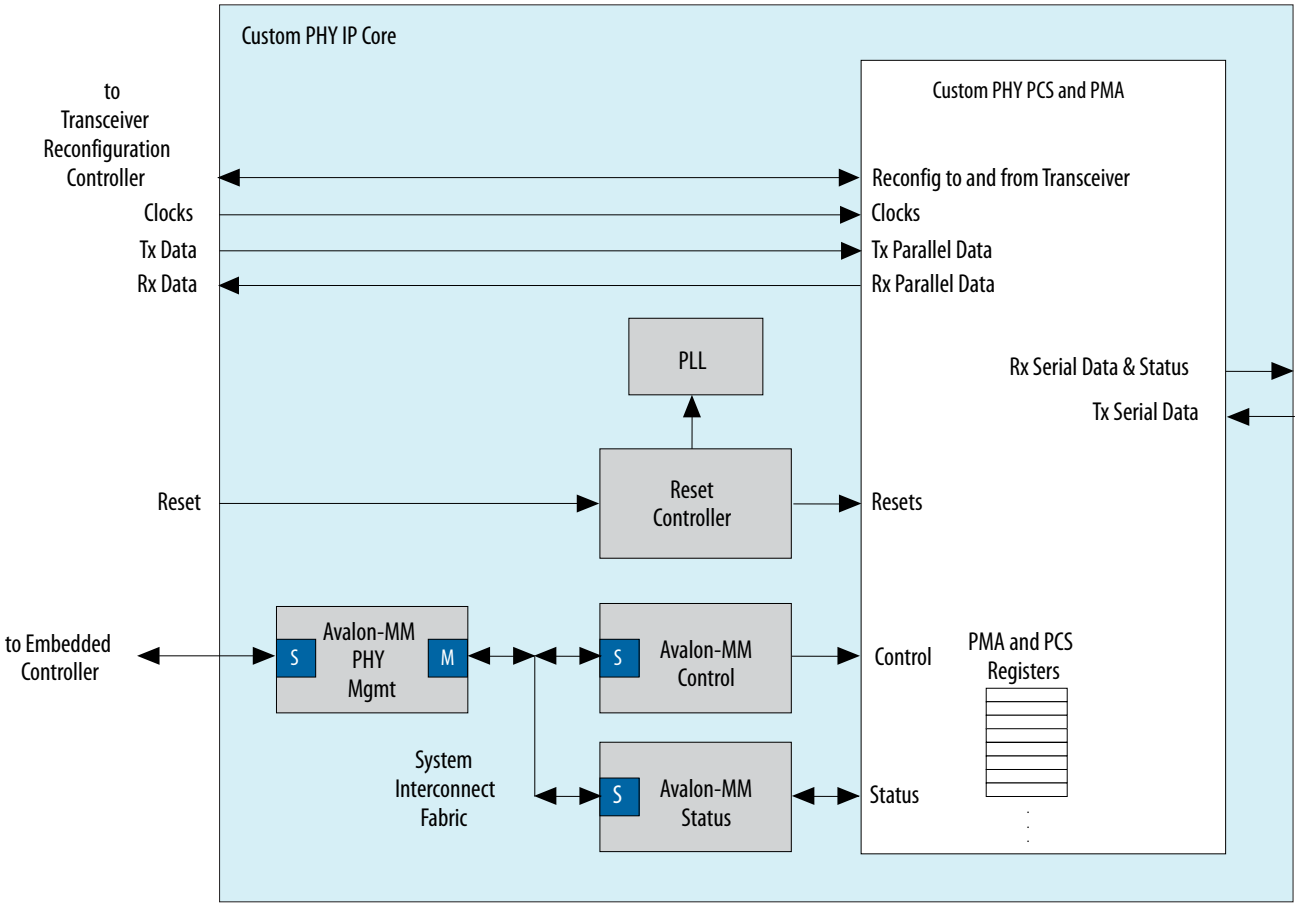


Table 10-20: Avalon-MM PHY Management Interface

Signal Name	Direction	Description
phy_mgmt_clk	Input	Avalon-MM clock input. There is no frequency restriction for the phy_mgmt_clk; however, if you plan to use the same clock for the PHY management interface and transceiver reconfiguration, you must restrict the frequency range of phy_mgmt_clk to 100-150 MHz to meet the specification for the transceiver reconfiguration clock.
phy_mgmt_clk_reset	Input	Global reset signal. This signal is active high and level sensitive.
phy_mgmt_address[8:0]	Input	9-bit Avalon-MM address.
phy_mgmt_writedata[31:0]	Input	Input data.

Signal Name	Direction	Description
phy_mgmt_readdata[31:0]	Output	Output data.
phy_mgmt_write	Input	Write signal.
phy_mgmt_read	Input	Read signal.
phy_mgmt_waitrequest	Output	When asserted, indicates that the Avalon-MM slave interface is unable to respond to a read or write request. When asserted, control signals to the Avalon-MM slave interface must remain constant.

Custom PHY IP Core Registers

This topic specifies the registers that you can access over the PHY management interface using word addresses and a 32-bit embedded processor. A single address space provides access to all registers.

Note: Writing to reserved or undefined register addresses may have undefined side effects.

PMA Common Control and Status Registers

Table 10-21: PMA Common Control and Status Registers

Word Addr	Bits	R/W	Register Name	Description
0x022	[31:0]	R	pma_tx_pll_is_locked	Bit[P] indicates that the TX/CMU PLL (P) is locked to the input reference clock. There is typically one pma_tx_pll_is_locked bit per system.

Reset Control Registers–Automatic Reset Controller

Table 10-22: Reset Control Registers–Automatic Reset Controller

Word Addr	Bits	R/W	Register Name	Description
0x041	[31:0]	RW	reset_ch_bitmask	Reset controller channel bit mask for reset registers at 0x042 and 0x044. The default value is all 1s. Channel <*n*> can be reset when bit <*n*> = 1.
0x042	[1:0]	R	reset_status (read)	Reading bit 0 returns the status of the reset controller TX ready bit. Reading bit 1 returns the status of the reset controller RX ready bit.

Reset Controls –Manual Mode

Table 10-23: Reset Controls –Manual Mode

Word Addr	Bits	R/W	Register Name	Description
0x044	[31:0] [31:4,0] are reserved	RW	reset_fine_control	You can use the reset_fine_control register to create your own reset sequence. If you disable **Enable embedded reset controller** on the **General Options** tab of the MegaWizard Plug-In Manager, you can design your own reset sequence using the tx_analogreset, rx_analogreset, tx_digitalreset, rx_digitalreset, and pll_powerdown which are top-level ports of the Custom Transceiver PHY. By default, the CDR circuitry is in automatic lock mode whether you use the embedded reset controller or design your own reset logic. You can switch the CDR to manual mode by writing the pma_rx_setlockto-data or pma_rx_set_locktoref registers to 1. It is safe to write 0s to reserved bits.
	[3]	RW	reset_rx_digital	Writing a 1 causes the internal RX digital reset signal to be asserted, resetting the RX digital channels enabled in reset_ch_bitmask. You must write a 0 to clear the reset condition.
	[2]	RW	reset_rx_analog	Writing a 1 causes the internal RX analog reset signal to be asserted, resetting the RX analog logic of all channels enabled in reset_ch_bitmask. You must write a 0 to clear the reset condition.
	[1]	RW	reset_tx_digital	Writing a 1 causes the internal TX digital reset signal to be asserted, resetting all channels enabled in reset_ch_bitmask. You must write a 0 to clear the reset condition.

PMA Control and Status Registers

Table 10-24: PMA Control and Status Registers

Word Addr	Bits	R/W	Register Name	Description
0x061	[31:0]	RW	`phy _ serial _ loopback`	Writing a 1 to channel <*n*> puts channel <*n*> in serial loopback mode.
0x063	[31:0]	R	`pma_rx_signaldetect`	When channel <*n*> =1, indicates that receive circuit for channel <*n*> senses the specified voltage exists at the RX input buffer.
0x064	[31:0]	RW	`pma_rx_set_locktodata`	When set, programs the RX CDR PLL to lock to the incoming data. Bit <*n*> corresponds to channel <*n*>.
0x065	[31:0]	RW	`pma_rx_set_locktoref`	When set, programs the RX CDR PLL to lock to the reference clock. Bit <*n*> corresponds to channel <*n*>.
0x066	[31:0]	RO	`pma_rx_is_lockedtodata`	When 1, indicates that the RX CDR PLL is locked to the RX data, and that the RX CDR has changed from LTR to LTD mode. Bit <*n*> corresponds to channel <*n*> .
0x067	[31:0]	RO	`pma_rx_is_lockedtoref`	When 1, indicates that the RX CDR PLL is locked to the reference clock. Bit <*n*> corresponds to channel <*n*>.

Custom PCS

Table 10-25: Custom PCS

Word Addr	Bits	R/W	Register Name	Description
0x080	[31:0]	RW	Lane or group number	Specifies lane or group number for indirect addressing, which is used for all PCS control and status registers. For variants that stripe data across multiple lanes, this is the logical group number. For non-bonded applications, this is the logical lane number.

Word Addr	Bits	R/W	Register Name	Description
0x081	[5:1]	R	rx_bitslipboundaryselectout	This is an output from the bit slip word aligner which shows the number of bits slipped. **From block:** Word aligner.
	[0]	R	rx_phase_comp_fifo_error	When set, indicates an RX phase compensation FIFO error. **From block:** RX phase Compensation FIFO
0x082	[0]	RW	tx_phase_comp_fifo_error	When set, indicates an TX phase compensation FIFO error. **From block:** TX phase Compensation FIFO
0x083	[5:1]	RW	tx_bitslipboundary_select	Sets the number of bits that the TX bit slipper needs to slip. **To block:** Word aligner.
	[0]	RW	tx_invpolarity	When set, the TX interface inverts the polarity of the TX data. **To block:** 8B/10B encoder.
0x084	0	RW	rx_invpolarity	When set, the RX channels inverts the polarity of the received data. **To block:** 8B/10B decoder.
0x085	[3]	RW	rx_bitslip	Every time this register transitions from 0 to 1, the RX data slips a single bit. **To block:** Word aligner.
	[2]	RW	rx_bytereversal_enable	When set, enables byte reversal on the RX interface. **To block:** Byte deserializer.
	[1]	RW	rx_bitreversal_enable	When set, enables bit reversal on the RX interface. **To block:** Word aligner.
	[0]	RW	rx_enapatternalign	When set in manual word alignment mode, the word alignment logic begins operation when this pattern is set. **To block:** Word aligner.

SDC Timing Constraints

The SDC timing constraints and approaches to identify false paths listed for Stratix V Native PHY IP apply to all other transceiver PHYs listed in this user guide. Refer to *SDC Timing Constraints of Stratix V Native PHY* for details.

Related Information

SDC Timing Constraints of Stratix V Native PHY on page 13-72
This section describes SDC examples and approaches to identify false timing paths.

Dynamic Reconfiguration

As silicon progresses towards smaller process nodes, circuit performance is affected more by variations due to process, voltage, and temperature (PVT). These process variations result in analog voltages that can be offset from required ranges.
The calibration performed by the dynamic reconfiguration interface compensates for variations due to PVT.

Each channel and each TX PLL have separate dynamic reconfiguration interfaces. The MegaWizard Plug-In Manager provides informational messages on the connectivity of these interfaces. The following example shows the messages for a single duplex channel parameterized for the 1.25 GIGE protocol.

Although you must initially create a separate reconfiguration interface for each channel and TX PLL in your design, when the Intel Quartus Prime software compiles your design, it reduces the number of reconfiguration interfaces by merging reconfiguration interfaces. The synthesized design typically includes a reconfiguration interface for at least three channels because three channels share an Avalon-MM slave interface which connects to the Transceiver Reconfiguration Controller IP Core. Conversely, you cannot connect the three channels that share an Avalon-MM interface to different Transceiver Reconfiguration Controller IP Cores. Doing so causes a Fitter error.

Example 10-1: Informational Messages for the Transceiver Reconfiguration Interface

```
PHY IP will require 2 reconfiguration interfaces for
connection to the external reconfiguration controller.
Reconfiguration interface offset 0 is connected to the transceiver channel.
Reconfiguration interface offset 1 is connected to the transmit PLL.
```

Table 10-26: Reconfiguration Interface

This interface uses the Avalon-MM PHY Management interface clock.

Signal Name	Direction	Description
`reconfig_to_xcvr [( <n> 70-1) :0]`	Input	Reconfiguration signals from the Transceiver Reconfiguration Controller. <*n*> grows linearly with the number of reconfiguration interfaces.

Signal Name	Direction	Description
reconfig_from_xcvr [(<*n*> 46-1) :0]	Output	Reconfiguration signals to the Transceiver Reconfiguration Controller. <*n*> grows linearly with the number of reconfiguration interfaces.

Transceiver dynamic reconfiguration requires that you assign the starting channel number if you are using ×6 or ×N bonding. Logical channel 0 should be assigned to either physical transceiver channel 1 or channel 4 of a transceiver bank. However, if you have already created a PCB with a different lane assignment for logical lane 0, you can use the workaound shown in the following example to remove this restriction. The example redefines the `pma_bonding_master` parameter using the Intel Quartus Prime Assignment Editor. In this example, the `pma_bonding_master` was originally assigned to physical channel 1. (The original assignment could also have been to physical channel 4.) The to parameter reassigns the `pma_bonding_master` to the Custom PHY instance name. You must substitute the instance name from your design for the instance name shown in quotation marks

Example 10-2: Overriding Logical Channel 0 Channel Assignment Restrictions in Stratix V Devices for ×6 or ×N Bonding

```
set_parameter -name pma_bonding_master "\"1\"" -to
"<custom phy instance>|altera_xcvr_custom:my_custom_phy_inst|
sv_xcvr_custom_nr:S5|sv_xcvr_custom_native:transceiver_core|
sv_xcvr_native:gen.sv_xcvr_native_insts[0].gen_bonded_group.sv_xcvr_native_in
st"
```

Related Information

Transceiver Reconfiguration Controller to PHY IP Connectivity on page 17-57

Low Latency PHY IP Core 11

2020.06.02

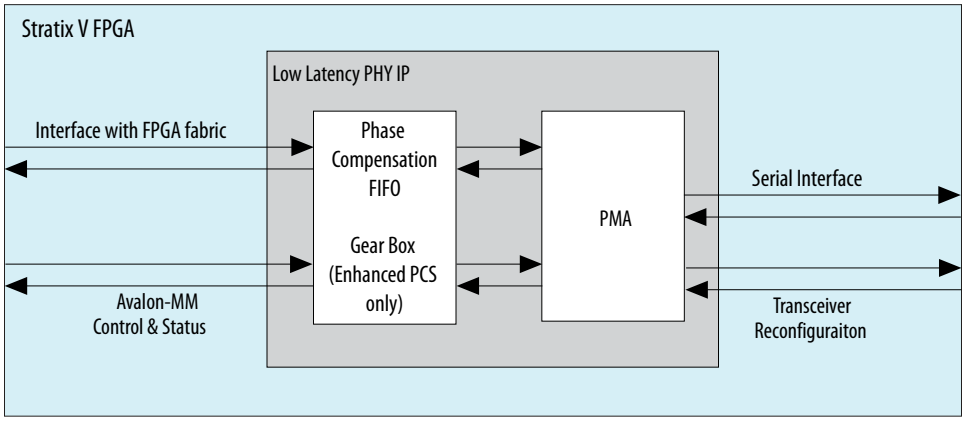
UG-01080

The Altera Low Latency PHY IP Core receives and transmits differential serial data, recovering the RX clock from the RX input stream. The PMA connects to a simplified PCS, which contains a phase compensation FIFO. Depending on the configuration you choose, the Low Latency PHY IP Core instantiates one of the following channels:

- GX channels using the Standard PCS
- GX channels using the 10G PCS
- GT channels in PMA Direct mode

An Avalon-MM interface provides access to control and status information. The following figure illustrates the top-level modules of the Low Latency PHY IP Core.

Figure 11-1: Low Latency PHY IP Core-Stratix V Devices



Because the Low Latency PHY IP Core bypasses much of the PCS, it minimizes the PCS latency.

For more detailed information about the Low Latency datapath and clocking, refer to the refer to the "Stratix V GX Device Configurations" section in the *Transceiver Configurations in Stratix V Devices* chapter of the *Stratix V Device Handbook*.

Related Information

Transceiver Configurations in Stratix V Devices

Device Family Support

IP cores provide either final or preliminary support for target Altera device families. These terms have the following definitions:

- *Final support*—Verified with final timing models for this device.
- *Preliminary support*—Verified with preliminary timing models for this device.

The following table shows the level of support offered by the Low Latency PHY IP Core for Altera device families.

Table 11-1: Device Family Support

Device Family	Support
Arria V GZ devices	Final
Stratix V devices	Final
Other device families	No support

Performance and Resource Utilization

The following table shows the typical expected device resource utilization for different configurations using the current version of the Intel Quartus Prime software targeting a Stratix V GX (5SGSMD612H35C2) device.

Table 11-2: Low Latency PHY Performance and Resource Utilization—Stratix V GX Device

Implementation	Number of Lanes	Serialization Factor	Worst-Case Frequency	Combinational ALUTs	Dedicated Registers	Memory Bits
11 Gbps	1	32 or 40	599.16	112	95	0
11 Gbps	4	32 or 40	584.8	141	117	0
11 Gbps	10	32 or 40	579.71	192	171	0
6 Gbps (10 Gbps datapath)	1	32 or 40	608.27	111	93	0
6 Gbps (10 Gbps datapath)	4	32 or 40	454.96	141	117	0

Implementation	Number of Lanes	Serialization Factor	Worst-Case Frequency	Combinational ALUTs	Dedicated Registers	Memory Bits
6 Gbps (10 Gbps datapath)	10	32 or 40	562.75	192	171	0
6 Gbps (8 Gbps datapath)	1	32 or 40	607.16	113	93	0
6 Gbps (8 Gbps datapath)	4	32 or 40	639.8	142	117	0
6 Gbps (8 Gbps datapath)	10	32 or 40	621.89	193	171	0
3 Gbps (8 Gbps datapath)	1	8, 10, 16, or 20	673.4	114	93	0
3 Gbps (8 Gbps datapath)	4	8, 10, 16, or 20	594.88	142	117	0
3 Gbps (8 Gbps datapath)	10	8, 10, 16, or 20	667.67	193	171	0

.

Parameterizing the Low Latency PHY

Complete the following steps to configure the Low Latency PHY IP Core in the MegaWizard Plug-In Manager:

1. Under **Tools** > **IP Catalog**, select **Stratix V** as the device family.
2. Under **Tools** > **IP Catalog** > **Interface Protocols** > **Transceiver PHY**, select **Low Latency PHY**.
3. Use the tabs on the MegaWizard Plug-In Manager to select the options required for the protocol.
4. Refer to the following topics to learn more about the parameters:

5. Click **Finish** to generate your parameterized Low Latency PHY IP Core.

General Options Parameters

The following table lists the settings available on **General Options** tab:

Table 11-3: Low Latency PHY General Options

Name	Value	Description
Device family	**Stratix V**	This IP core is only available for Stratix V devices.
Datapath type	**Standard** **10G** **GT**	The Low Latency PHY IP Core is part of a **Standard**, **10G**, or **GT** datapath. In most cases the **FPGA fabric transceiver interface width** determines the bandwidth of the datapath; however, when the **FPGA fabric transceiver interface width** is 32 or 40 bits, you have the option of using either the **Standard** datapath which is the default mode, or changing to the **10G** datapath by selecting this option. Refer to **Table 11-4** *Datapath Width Support* for a comprehensive list of datapath support.
Mode of operation	**Duplex** **RX** **TX**	Specifies the mode of operation as **Duplex**, **RX,** or **TX** mode.
Number of lanes	1-32	Specifies the total number of lanes in each direction. Stratix V devices include up to 32 GX channels (Standard or 10G) and up to 4 **GT** channels. You must instantiate each **GT** channel in a separate Low Latency PHY IP Core instance. You cannot specify both GX and GT channels within the same instance.
Enable lane bonding	**On/Off**	When enabled, the PMA uses the same clock source for up to 6 channels in a transceiver bank, reducing clock skew. Turn this option **Off** if you are using multiple TX PLLs in a single Low Latency PHY IP Core instance.

Name	Value	Description
Bonding mode	**×N** **fb_compensation**	Select **×N** to use the same clock source for up to 6 channels in a single transceiver bank, resulting in reduced clock skew. You must use contiguous channels when you select **×N** bonding. In addition, you must place logical channel 0 in either physical channel 1 or 4. Physical channels 1 and 4 are indirect drivers of the **×N** clock network. Select **fb_compensation** (feedback compensation) to use the same clock source for multiple channels across different transceiver banks to reduce clock skew. For more information about bonding, refer to "Bonded Channel Configurations Using the PLL Feedback Compensation Path" in *Transceiver Clocking in Stratix V Devices* in volume 2 of the Stratix V Device Handbook.
FPGA fabric transceiver interface width	**8, 10, 16, 20, 32, 40, 50, 64, 66, 128**	This option indicates the parallel data fabric transceiver interface width. **GT** datapath supports a single width of 128 bits. Refer to **Table 11-4** Datapath Width Support for the supported interface widths of the **Standard** and **10G** datapaths.
PCS PMA interface width	**8, 10, 16, 20, 30, 32, 64**	The **PCS-PMA interface width** depends on the F**PGA fabric transceiver interface width** and the Datapath type. Refer to Datapath Width Support for the supported interface widths of the **Standard** and **10G** datapaths.
PLL type	**CMU** **ATX**	The **CMU** PLL is available for the **Standard** and **10G** datapaths. The **ATX** PLL is available for the **Standard**, **10G**, and **GT** datapaths. The **CMU** PLL has a larger frequency range than the **ATX** PLL. The **ATX** PLL is designed to improve jitter performance and achieves lower channel-to-channel skew; however, it supports a narrower range of data rates and reference clock frequencies. Another advantage of the **ATX** PLL is that it does not use a transceiver channel, while the **CMU** PLL does. An informational message displays in the message panel if the **PLL type** that you select is not available at the frequency specified.

Name	Value	Description
Data rate	**Device dependent**	Specifies the data rate in Mbps. Refer to *Stratix V Device Datasheet* for the data rate ranges of datapath.
Base data rate	1 × **Data rate** 2 × **Data rate** 4 × **Data rate**	Select a **base data rate** that minimizes the number of PLLs required to generate all the clocks required for data transmission. By selecting an appropriate **base data rate**, you can change data rates by changing the divider used by the clock generation block. For higher frequency data rates 2 × and 4× base data rates are not available.
Input clock frequency	Variable	Specifies the frequency of the PLL input reference clock. The Input clock frequency drop down menu is populated with all valid frequencies derived as a function of the data rate and base data rate. However, if you select `fb_compensation` as the bonding mode, then the input reference clock frequency is limited to the (data rate) / (PCS-PMA interface width).

The following table lists **Standard** and **10G** datapath widths for the FPGA fabric-transceiver interface, the PCS-PMA interface, and the resulting frequencies for the `tx_clkout` and `rx_clkout` parallel clocks. In almost all cases, the parallel clock frequency is described by the following equation:

$$\text{frequency}_{\text{parallel clock}} = \text{data rate/FPGA fabrictransceiver interface width}$$

Note: The FPGA fabric transceiver interface width is always 128 bits for the **GT** datapath.

Table 11-4: Datapath Width Support

FPGA Fabric - Transceiver Interface Width	PCS-PMA Interface Width		tx_clkout and rx_clkout frequency
	Standard Datapath	10G Datapath	
8	8	—	data rate/8
10	10	—	data rate/10
16	8 or 16	—	data rate/16
20	10 or 20	—	data rate/20
32	16	32	data rate/32
40	20	40	data rate/40

FPGA Fabric - Transceiver Interface Width	PCS-PMA Interface Width		tx_clkout and rx_clkout frequency
	Standard Datapath	10G Datapath	
50	—	40	data rate/50 (7)
64	—	32	data rate/32 (8)
64	—	64	data rate/64
66	—	40	data rate/66

Related Information

- Stratix V Device Datasheet
- Transceiver Clocking in Stratix V Devices

Additional Options Parameters

The parameters on the **Additional Options** tab control clocking and datapath options. Both bonded (×N) and non-bonded modes are available. In bonded modes, a single PLL can drive all channels. In non-bonded modes, each channel may have its own PLL.

(7) For this datapath configuration, the `tx_clkout` frequency generated by the Low Latency PHY is the **data rate** /40. You must generate a /50 frequency clock from the /40 clock and feed this clock back into the `tx_coreclkin`. The `rx_clkout` frequency generated by the Low Latency PHY is /40 of the data rate. You must generate a /50 frequency from the recovered clock and feed this back into the `rx_coreclkin`.

(8) For this datapath configuration, the `tx_clkout` frequency generated by the Low Latency PHY is the **data rate**/32. You must generate a /64 frequency clock from the /32 clock and feed this clock back into the `tx_coreclkin`. The `rx_clkout` frequency generated by the Low Latency PHY is the data rate/32. You must generate a /64 frequency from the recovered clock and feed this back into the `rx_coreclkin`

The following table describes the options available on the **Additional Options** tab:

Table 11-5: Additional Options

Name	Value	Description
Enable tx_coreclkin	**On/Off**	When you turn this option on, `tx_coreclkin` connects to the write clock of the TX phase compensation FIFO and you can clock the parallel TX data generated in the FPGA fabric using this port. This port allows you to clock the write side of the TX phase compensation FIFO with a user-provided clock, either the FPGA fabric clock, the FPGA fabric-TX interface clock, or the input reference clock. You must turn this option **On** when the FPGA **fabric transceiver interface width:PCS-PMA interface width** is 50:40 or when you specify the **10G** datapath with a **fabric transceiver interface width:PCS-PMA interface width** of 64:32. For the GT datapath, if you are using different reference clock pins for the TX and RX channels, you must instantiate two separate Low Latency PHY IP Core instances for TX and RX channels. The reference clock pins for each channel must reside in the same transceiver bank. For more information refer to the "FPGA Fabric-Transceiver Interface Clocking" section in the *Stratix V Transceiver Clocking* chapter.
Enable rx_coreclkin	**On/Off**	When you turn this option on, `rx_coreclkin` connects to the read clock of the RX phase compensation FIFO and you can clock the parallel RX output data using `rx_coreclk`. This port allows you to clock the read side of the RX phase compensation FIFO with a userprovided clock, either the FPGA fabric clock, the FPGA fabric RX interface clock, or the input reference clock. **rx_coreclkin** is not available for the GT datapath. You must turn this option **On** when the FPGA **fabric transceiver interface width:PCS-PMA Interface width** is 50:40 or when you specify the **10G** datapath with a **fabric transceiver interface width:PCS-PMA Interface width** of 64:32. For more information refer to the "FPGA Fabric-Transceiver Interface Clocking" section in the *Stratix V Transceiver Clocking* chapter.

Name	Value	Description
Enable TX bitslip	**On/Off**	The bit slip feature allows you to slip the transmitter side bits before they are sent to the gearbox. The maximum number of bits slipped is equal to the ((FPGA fabric-to-transceiver interface width) – 1). For example, if the FPGA fabric-to-transceiver interface width is 64 bits, the bit slip logic can slip a maximum of 63 bits. Each channel has 5 bits to determine the number of bits to slip. The value specified on the TX bitslip bus indicates the number of bit slips. Effectively, each value shifts the word boundary by one bit. For example, a TX bitslip value of 1 on a 64bit FPGA interface width shifts the word boundary by 1 bit. That is, bit[63] from the first word and bit[62:0] are concatenated to form a 64 bit word (bit[62:0] from the second word, bit[63] from the first word LSB). This option is only available for the **Standard** and **10G** datapaths.
Enable RX bitslip	**On/Off**	When enabled, the wordaligner operates in bitslip mode. This option is available for Stratix V and Arria V GZ devices using the **10G** datapath.
Enable embedded reset control	**On/Off**	This option is turned on by default. When **On**, the embedded reset controller initiates the reset sequence when it receives a positive edge on the `phy_mgmt_clk_reset` input signal. Disable this option to implement your own reset sequence using the `tx_analogreset`, `rx_analogreset`, `tx_digitalreset`, `rx_digital-reset`, and `pll_powerdown` which are available as top-level ports of the Low Latency Transceiver PHY. When you design your own reset controller, the `tx_ready` and `rx_ready` are not top-level signals of the core. Another option is to use Altera's Transceiver PHY Reset Controller IP Core to reset the transceivers. For more information, refer to the *Transceiver PHY Reset Controller IP Core*chapter. For more information about designing a reset controller, refer to the *User-Controlled Reset Controller* section in the *Transceiver Reset Control in Stratix V Devices* in volume 2 of the *Stratix V Device Handbook*.

Name	Value	Description
Avalon data interfaces	**On/Off**	When you turn this option **On**, the order of symbols is changed. This option is typically required if you are planning to import your Low Latency Transceiver PHY IP Core into a Qsys system.

Related Information

- **Stratix V Transceiver Clocking**
- **Transceiver Reset Control in Stratix V Devices**

PLL Reconfiguration Parameters

The following table describes the options available on the **PLL Reconfiguration** tab.

Note: The PLL reconfiguration options are not available for the GT datapath.

Table 11-6: PLL Reconfigurations

Name	Value	Description
Allow PLL/CDR Reconfiguration	**On/Off**	You must enable this option if you plan to reconfigure the PLLs in your design. This option is also required to simulate PLL reconfiguration.
Number of TX PLLs	**1–4**	Specifies the number of TX PLLs that can be used to dynamically reconfigure channels to run at multiple data rates. If your design does not require transceiver TX PLL dynamic reconfiguration, set this value to 1. The number of actual physical PLLs that are implemented depends on the selected clock network. Each channel can dynamically select between n PLLs, where n is the number of PLLs specified for this parameter. You must disable the embedded reset controller and design your own controlled reset controller or the use the highly configurable reset core described in *Transceiver PHY Reset Controller IP Core* if you intend to use more than 1 TX PLL for a Low Latency PHY IP instance. **Note:** For more details, refer to the *Transceiver Clocking* chapter in the device handbook for the device family you are using.

Name	Value	Description
Number of reference clocks	**1–5**	Specifies the number of input reference clocks. More than one reference clock may be required if your design reconfigures channels to run at multiple frequencies.
Main TX PLL logical index	**0–3**	Specifies the index for the TX PLL that should be instantiated at startup. Logical index 0 corresponds to TX PLL0, and so on.
CDR PLL input clock source	**0–3**	Specifies the index for the TX PLL input clock that should be instantiated at startup. Logical index 0 corresponds to input clock 0 and so on.
TX PLL (0–3) (Refer to *Low Latency PHY General Options* for a detailed explanation of these parameters.)		
PLL Type	**CMU** **ATX**	Specifies the PLL type.
Base data rate	1 × **Data rate** 2 × **Data rate** 4 × **Data rate** 8 × **Data rate**	Specifies **Base data rate**.
Reference clock frequency	Variable	Specifies the frequency of the PLL input reference clock. The PLL must generate an output frequency that equals the **Base data rate**/2. You can use any **Input clock frequency** that allows the PLLs to generate this output frequency.
Selected reference clock source	**0–4**	Specifies the index of the input clock for this TX PLL. Logical index 0 corresponds to input clock 0 and so on.

Channel Interface		
Enable Channel Interface	**On/Off**	Turn this option on to enable PLL and datapath dynamic reconfiguration. When you select this option, the width of `tx_parallel_data` and `rx_parallel_data` buses increases in the following way. • **Standard** datapath: • The `tx_parallel_data` bus is 44 bits per lane; however, only the loworder number of bits specified by the **FPGA fabric transceiver interface width** contain valid data for each lane. • The `rx_parallel_data` bus is 64 bits per lane; however, only the loworder number of bits specified by the **FPGA fabric transceiver interface width** contain valid data. • **10G** datapath: • The both the `tx_parallel_data` and `rx_parallel_data` buses are 64 bits per lane; however, only the loworder number of bits specified by the **FPGA fabric transceiver interface width** contain valid data.

Related Information

- **PLL Reconfiguration** on page 17-34
- **General Options Parameters** on page 11-4

Low Latency PHY Analog Parameters

For analog parameters refer to Analog Settings for Stratix V Devices.

Related Information

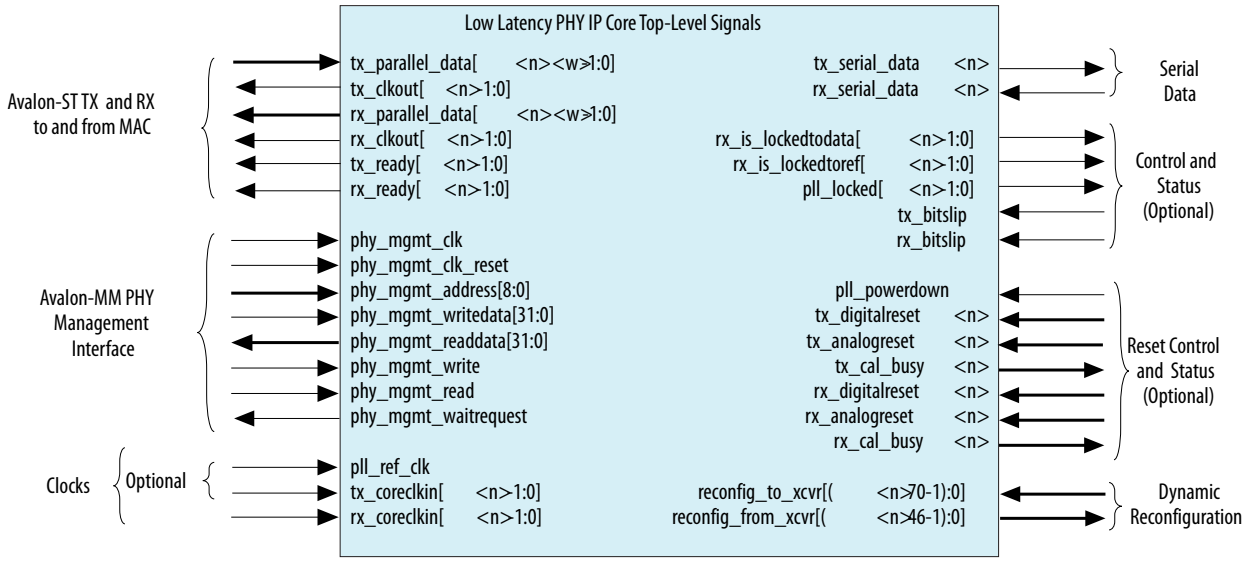
Analog Parameters Set Using QSF Assignments on page 20-1

Low Latency PHY Interfaces

The following figure illustrates the top-level signals of the Custom PHY IP Core. The variables in this figure represent the following parameters:

- <n>—The number of lanes
- <w>—The width of the FPGA fabric to transceiver interface per lane

Figure 11-2: Top-Level Low Latency Signals



Note: By default **block diagram** shown in the MegaWizard Plug-In Manager labels the external pins with the *interface type* and places the *interface name* inside the box. The interface type and name are used in the **_hw.tcl** file that describes the component. If you turn on **Show signals**, the **block diagram** displays all toplevel signal names.

For more information about **_hw.tcl** files refer to refer to the **Component Interface Tcl Reference** chapter in volume 1 of the Intel Quartus Prime Handbook.

Low Latency PHY Data Interfaces

The following table describes the signals in the Avalon-ST interface. This interface drives AvalonST TX and RX data to and from the FPGA fabric. These signals are named from the point of view of the MAC so that the TX interface is an Avalon-ST sink interface and the RX interface is an Avalon-ST source.

Table 11-7: Avalon-ST interface

Signal Name	Direction	Description
`tx_parallel_data[<n><w>-1:0]`	Input	This is TX parallel data driven from the MAC FPGA fabric. The ready latency on this interface is 0, so that the PCS in Low-Latency Bypass Mode or the MAC in PMA Direct mode must be able to accept data as soon as it comes out of reset.
`tx_clkout[<n>-1:0]`	Output	This is the clock for TX parallel data.

Signal Name	Direction	Description
tx_ready[<n>-1:0]	Output	When asserted, indicates that the Low Latency IP Core has exited the reset state is ready to receive data from the MAC. This signal is available if you select **Enable embedded reset control** on the **Additional Options** tab.
rx_parallel_data [<*n*><*w*>-1:0]	Output	This is RX parallel data driven by the Low Latency PHY IP Core. Data driven from this interface is always valid.
rx_clkout[<n>-1:0]	Output	Low speed clock recovered from the serial data.
rx_ready[<*n*>-1:0]	Output	This is the ready signal for the RX interface. The ready latency on this interface is 0, so that the MAC must be able to accept data as soon as the PMA comes out of reset. This signal is available if you select **Enable embedded reset control** on the **Additional Options** tab.

The following table describes the signals that comprise the serial data interface:

Table 11-8: Serial Data Interface

Signal Name	Direction	Description
rx_serial_data[<*n*>-1:0]	Input	Differential high speed input serial data.
tx_serial_data [<*n*>-1:0]	Output	Differential high speed output serial data.

Optional Status Interface

The following table describes the signals that comprise the optional status interface:

Table 11-9: Optional Status Interface

Signal Name	Direction	Description
rx_is_lockedtodata[<n>-1:0]	Output	When asserted, indicates that the RX CDR is locked to incoming data. This signal is optional. If latency is not critical, you can read the value of this signal from the Rx_is_lockedtodata register.
rx_is_lockedtoref[<n>-1:0]	Output	When asserted, indicates that the RX CDR is locked to the input reference clock. This signal is optional. When the RX CDR is locked to data, you can ignore transitions on this signal. If latency is not critical, you can read the value of this signal from the rx_is_lockedtoref register.
pll_locked[<n>-1:0]	Output	When asserted, indicates that the TX PLL is locked to the input reference clock. This signal is asynchronous.
tx_bitslip[<n>-1:0]	Input	When set, the data sent to the PMA is slipped. The maximum number of bits that can be slipped is equal to the value selected in the serialization factor field - 1 or <d> -1.
rx_bitslip[<n>-1:0]	Input	When set, the RX word aligner operates in bit slip mode.

Low Latency PHY Clock Interface

The following table describes reference clock for the Low Latency PHY. The input reference clock, pll_ref_clk, drives a PLL inside the PHY-layer block, and a PLL output clock, rx_clkout is used for all data, command, and status inputs and outputs.

Table 11-10: Clock Signals

Signal Name	Direction	Description
tx_coreclkin[<n>-1:0]	Input	This is an optional clock to drive the write side of the TX FIFO.
rx_coreclkin[<n>-1:0]	Input	This is an optional clock to drive the read side of the RX FIFO.

Signal Name	Direction	Description
pll_ref_clk	Input	Reference clock for the PHY PLLs. The frequency range is 60–700 MHz.

Optional Reset Control and Status Interface

The following table describes the signals in the optional reset control and status interface. These signals are available if you do not enable the embedded reset controller. For more information including timing diagrams, refer to **Transceiver Reset Control in Stratix V Devices** in volume 2 of the Stratix V Device Handbook.

Table 11-11: Avalon-ST RX Interface

Signal Name	Direction	Description
pll_powerdown	Input	When asserted, resets the TX PLL.
tx_digitalreset[<n>-1:0]	Input	When asserted, reset all blocks in the TX PCS. If your design includes bonded TX PCS channels, refer to *Timing Constraints for Reset Signals* when Using Bonded PCS Channels for a SDC constraint you must include in your design.
tx_analogreset[<n>-1:0]	Input	When asserted, resets all blocks in the TX PMA.
tx_cal_busy[<n>-1:0]	Output	When asserted, indicates that the initial TX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP. You must hold the channel in reset until calibration completes.
rx_digitalreset[<n>-1:0]	Input	When asserted, resets the RX PCS.
rx_analogreset[<n>-1:0]	Input	When asserted, resets the RX CDR.
rx_cal_busy[<n>-1:0]	Output	When asserted, indicates that the initial RX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP.

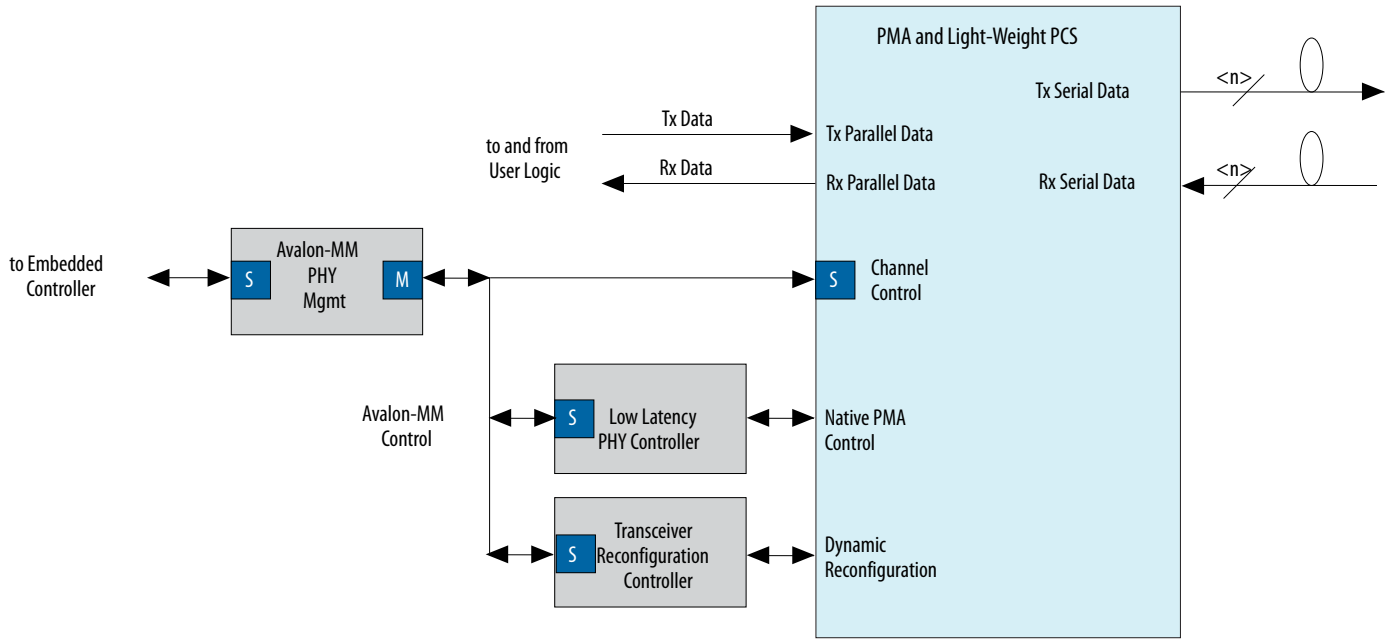
Related Information

Timing Constraints for Bonded PCS and PMA Channels on page 18-11

Register Interface and Register Descriptions

The Avalon-MM PHY management interface provides access to the Low Latency PHY PCS and PMA registers that control the TX and RX channels, the PMA powerdown, PLL registers, and loopback modes. The following figure provides a high level view of this hardware.

Figure 11-3: PMA Top-Level Modules



The following table describes the signals in the PHY Management interface:

Table 11-12: Avalon-MM PHY Management Interface

Signal Name	Direction	Description
`phy_mgmt_clk`	Input	Avalon-MM clock input. There is no frequency restriction for the `phy_mgmt_clk`; however, if you plan to use the same clock for the PHY management interface and transceiver reconfiguration, you must restrict the frequency range of `phy_mgmt_clk` to 100–150 MHz to meet the specification for the transceiver reconfiguration clock.
`phy_mgmt_clk_reset`	Input	Global reset signal. This signal is active high and level sensitive. This is an asynchronous signal.

Signal Name	Direction	Description
phy_mgmtaddress[8:0]	Input	9-bit Avalon-MM address.
phy_mgmt_writedata[31:0]	Input	Input data.
phy_mgmt_readdata[31:0]	Output	Output data.
phy_mgmt_write	Input	Write signal.
phy_mgmt_read	Input	Read signal.

For more information about the Avalon-MM and Avalon-ST protocols, including timing diagrams, refer to the **Avalon Interface Specifications** .

The following table describes the registers that you can access over the PHY Management Interface using word addresses and a 32-bit embedded processor. The automatic reset controller automatically performs the required reset sequence. After this reset sequence completes, you can manually initiate TX or RX resets using the `reset_control` control register. You can also specify the clock data recovery (CDR) circuit to lock to the incoming data or the reference clock using the `pma_rx_set_locktodata` and `pma_rx_set_locktoref` registers.

Note: Writing to reserved or undefined register addresses may have undefined side effects.

Table 11-13: Low Latency PHY IP Core Registers (Part 1 of 2)

Word Addr	Bits	R/W	Register Name	Description
Reset Control Registers–Automatic Reset Controller				
0x041	[31:0]	RW	reset_ch_bitmask	Reset controller channel bitmask for digital resets. The default value is all 1s. Channel <*n*> can be reset when bit <*n*> = 1.
0x042	[1:0]	W	reset_control (write)	Writing a 1 to bit 0 initiates a TX digital reset using the reset controller module. The reset affects channels enabled in the reset_ch_bitmask. Writing a 1 to bit 1 initiates a RX digital reset of channels enabled in the reset_ch_bitmask.
		R	reset_status (read)	Reading bit 0 returns the status of the reset controller TX ready bit. Reading bit 1 returns the status of the reset controller RX ready bit.

Word Addr	Bits	R/W	Register Name	Description
Reset Control Registers–Automatic Reset Controller				
0x061	[31:0]	RW	phy_serial_loopback_	Writing a 1 to channel < n > puts channel < n > in serial loopback mode. For information about pre or postCDR serial loopback modes, refer to Loopback Modes.
PMA Control and Status Registers				
0x063	[31:0]	R	pma_rx_signaldetect	When channel <*n*> =1, indicates that receive circuit for channel <*n*> senses the specified voltage exists at the RX input buffer.
0x064	[31:0]	RW	pma_rx_set_locktodata	When set, programs the RX CDR PLL to lock to the incoming data. Bit <*n*> corresponds to channel <*n*>.
0x065	[31:0]	RW	pma_rx_set_locktoref	When set, programs the RX CDR PLL to lock to the reference clock. Bit <*n*> corresponds to channel <*n*>.
0x066	[31:0]	RO	pma_rx_is_lockedto-data	When asserted, indicates that the RX CDR PLL is locked to the RX data, and that the RX CDR has changed from LTR to LTD mode. Bit <*n*> corresponds to channel <*n*>.
0x067	[31:0]	RO	pma_rx_is_lockedtoref	When asserted, indicates that the RX CDR PLL is locked to the reference clock. Bit <*n*> corresponds to channel <*n*>.

Dynamic Reconfiguration

As silicon progresses towards smaller process nodes, circuit performance is affected more by variations due to process, voltage, and temperature (PVT). These process variations result in analog voltages that can be offset from required ranges. The calibration performed by the dynamic reconfiguration interface compensates for variations due to PVT.

Each channel and each TX PLL have separate dynamic reconfiguration interfaces. The MegaWizard Plug-In Manager provides informational messages on the connectivity of these interfaces. The following example shows the messages for a single duplex channel.

Example 11-1: Informational Messages for the Transceiver Reconfiguration Interface

PHY IP will require 2 reconfiguration interfaces for connection to the external reconfiguration controller.

Reconfiguration interface offset 0 is connected to the transceiver channel.

Reconfiguration interface offset 1 is connected to the transmit PLL.

Although you must initially create a separate reconfiguration interface for each channel and TX PLL in your design, when the Intel Quartus Prime software compiles your design, it reduces the number of reconfiguration interfaces by merging reconfiguration interfaces. The synthesized design typically includes a reconfiguration interface for at least three channels because three channels share an Avalon-MM slave interface which connects to the Transceiver Reconfiguration Controller IP Core. Conversely, you cannot connect the three channels that share an Avalon-MM interface to different Transceiver Reconfiguration Controller IP Cores. Doing so causes a Fitter error. For more information, refer to Transceiver Reconfiguration Controller to PHY IP Connectivity.

The following table describes the signals in the reconfiguration interface. This interface uses a clock provided by the reconfiguration controller.

Table 11-14: Reconfiguration Interface

Signal Name	Direction	Description
`reconfig_to_xcvr [(<n>70)-1:0]`	Input	Reconfiguration signals from the Transceiver Reconfiguration Controller. <n> grows linearly with the number of reconfiguration interfaces.
`reconfig_from_xcvr [(<n>46)-1:0]`	Output	Reconfiguration signals to the Transceiver Reconfiguration Controller. <n> grows linearly with the number of reconfiguration interfaces.

If you are using ×6 or ×N bonding, transceiver dynamic reconfiguration requires that you assign the starting channel number. Logical channel 0 should be assigned to either physical transceiver channel 1 or channel 4 of a transceiver bank. However, if you have already created a PCB with a different lane assignment for logical lane 0, you can use the workaound shown in The following example to remove this restriction. This example redefines the `pma_bonding_master` parameter using the Intel Quartus Prime Assignment Editor. In this example, the `pma_bonding_master` was originally assigned to physical channel 1. (The original assignment could also have been to physical channel 4.) The `to` parameter reassigns the `pma_bonding_master` to the Low Latency PHY instance name. You must substitute the instance name from your design for the instance name shown in quotation marks.

Example 11-2: Overriding Logical Channel 0 Channel Assignment Restrictions in Stratix V Devices for ×6 or ×N Bonding

```
set_parameter -name pma_bonding_master "\"1\"" -to "<low latency phy instance>
|altera_xcvr_low_latency_phy:my_low_latency_phy_inst|sv_xcvr_low_latency_phy_nr:
sv_xcvr_low_latency_phy_nr_inst|sv_xcvr_10g_custom_native:sv_xcvr_10g_custom_native_inst
|sv_xcvr_native:sv_xcvr_native_insts[0].gen_bonded_group_native.sv_xcvr_native_inst"
```

SDC Timing Constraints

The SDC timing constraints and approaches to identify false paths listed for Stratix V Native PHY IP apply to all other transceiver PHYs listed in this user guide. Refer to *SDC Timing Constraints of Stratix V Native PHY* for details.

Related Information

SDC Timing Constraints of Stratix V Native PHY on page 13-72
This section describes SDC examples and approaches to identify false timing paths.

Simulation Files and Example Testbench

Refer to Running a Simulation Testbench for a description of the directories and files that the Intel Quartus Prime software creates automatically when you generate your Low Latency PHY IP Core.

Refer to the **Altera wiki** for an example testbench that you can use as a starting point in creating your own verification environment.

Deterministic Latency PHY IP Core 12

2020.06.02

UG-01080 Subscribe Send Feedback

Deterministic latency enables accurate delay measurements and known timing for the transmit (TX) and receive (RX) datapaths as required in applications such as wireless communication systems, emerging Ethernet standards, and test and measurement equipment. The Deterministic Latency PHY IP Core support 1-32 lanes with a continuous range of data rates from 611–6144 Mbps for Arria V devices, 0.6222–6.144 Gbps in Arria V GZ, 611–5000 Mbps in Cyclone V devices, and 611 Mbps–12200 Mbps for Stratix V devices. By setting the appropriate options using the MegaWizard Plug-In Manager, you can configure the Deterministic Latency PHY IP Core to support many industry-standard protocols that require deterministic latency, including the following protocols:

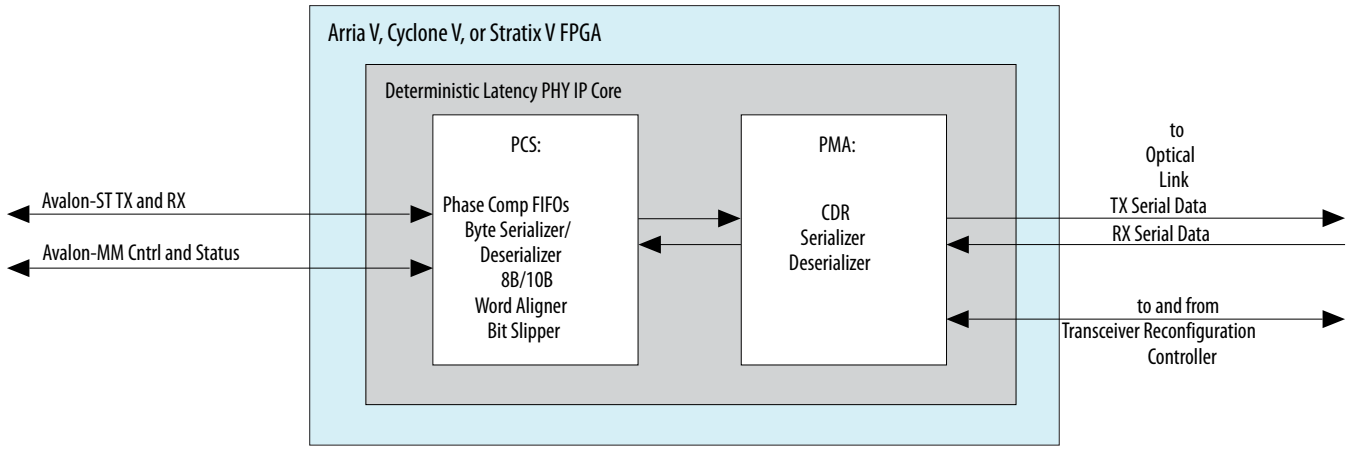
- Common Public Radio Interface (CPRI)
- Open Base Station Architecture Initiative (OBSAI)
- 1588 Ethernet

For more information about using the Deterministic Latency PHY IP Core to implement CPRI, refer to the application note, *Implementing the CPRI Protocol Using the Deterministic PHY IP Core*.

The following figure illustrates the top-level interfaces and modules of the Deterministic Latency PHY IP Core. As the figure shows, the physical coding sublayer (PCS) includes the following functions:

- TX and RX Phase Compensation FIFO
- Byte serializer and deserializer
- 8B/10B encoder and decoder
- Word aligner
- TX bit slipper

Figure 12-1: Deterministic Latency PHY IP Core



The data that the Deterministic Latency PHY receives data on its FPGA fabric interface employs the Avalon Streaming (Avalon-ST) protocol to transmit and receive data. The Avalon-ST protocol is a simple protocol designed for driving high bandwidth, low latency, unidirectional data. The Deterministic Latency PHY IP Core also includes an Avalon Memory-Mapped (Avalon-MM) interface to access control and status registers. This is a standard, memory-mapped protocol that is normally used to read and write registers and memory. The transceiver reconfiguration interface connects to the Altera Transceiver Reconfiguration Controller IP Core which can dynamically reconfigure transceiver settings. Finally, the PMA transmits and receives serial data.

Related Information

- **Implementing the CPRI Protocol Using the Deterministic PHY IP Core**
- **Avalon Interface Specifications**

Deterministic Latency Auto-Negotiation

The Deterministic Latency PHY IP Core supports auto-negotiation. When required, the channels initialize at the highest supported frequency and switch to successively lower data rates if frame synchronization is not achieved.

If your design requires auto-negotiation, choose a base data rate that minimizes the number of PLLs required to generate the clocks required for data transmission. By selecting an appropriate base data rate, you can change data rates by changing the divider used by the clock generation block. The following table shows an example where setting two base data rates, 4915.2 and 6144 Mbps, with the appropriate clock dividers generates almost the full range of data rates required by the CPRI protocol.

Note: You can use PMA Direct mode in the Transceiver Native PHYs for CPRI applications that require higher frequencies. For more information refer to the following documents:

Related Information

- **Arria V Transceiver Native PHY IP Core** on page 14-1
- **Stratix V Transceiver Native PHY IP Core** on page 13-1

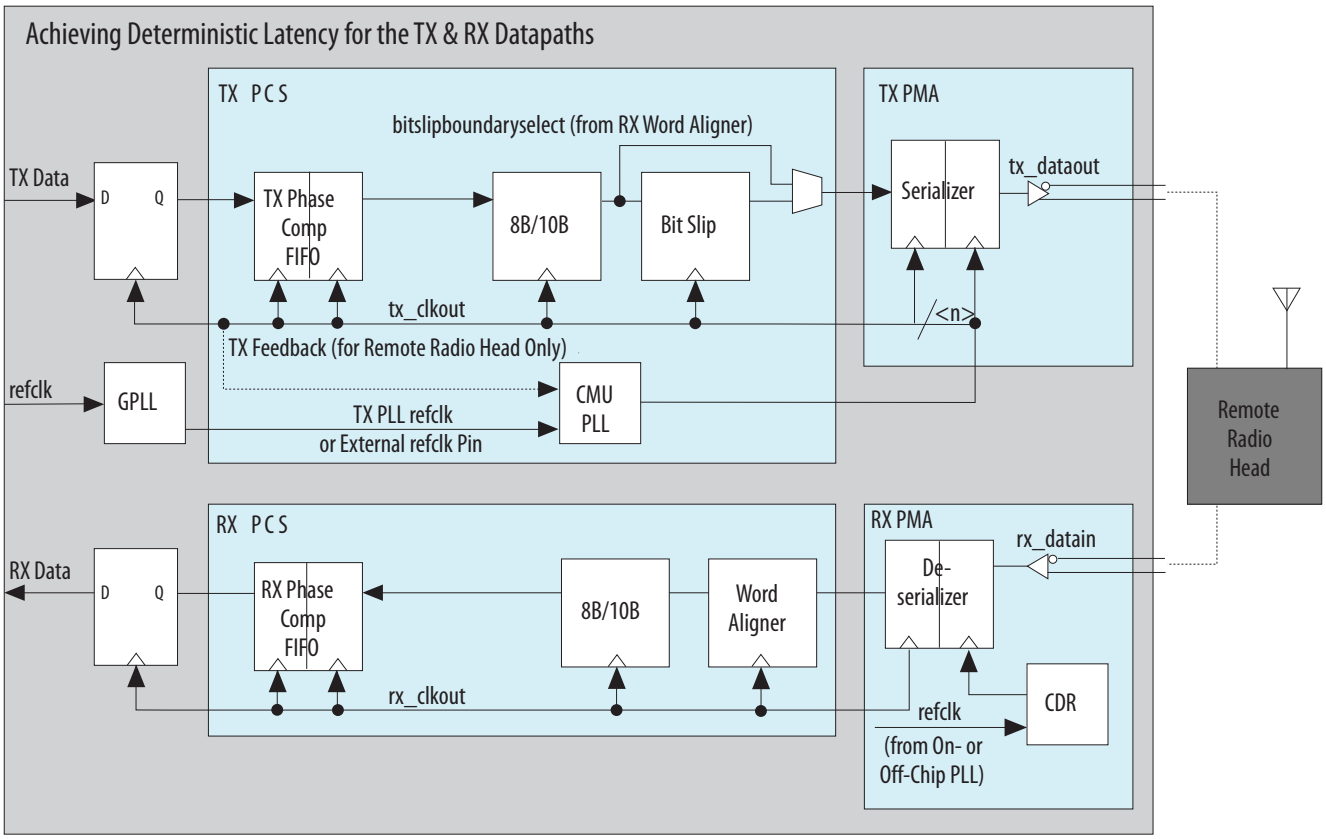
Achieving Deterministic Latency

This section provides an overview of the calculation that help you achieve deterministic delay in the Deterministic Latency PHY IP core.

This figure illustrates the TX and RX channels when configured as a wireless basestation communicating to a remote radio head (RRH) using a CPRI or OBSAI interface. The figure also provides an overview of the calculations that guarantee deterministic delay. As this figure illustrates, you can use a general-purpose PLL to generate the clock that drives the TX CMU PLL or an external reference clock input pin.

Figure 12-2: Achieving Deterministic Latency for the TX and RX Datapaths

The TX and RX Phase Compensation FIFOs always operate in register mode.



To control the total latency through the datapath, use sampling techniques in a delay estimate FIFO to measure the phase difference between the `tx_clkout` and `rx_clkout`, and the clock output of the PLL (as shown in above figure) and ensure the delay through the FIFO to a certain accuracy.

Note: Systems that require multiple frequencies in a single transceiver block must use a delay estimate FIFO to determine delay estimates and the required phase adjustments.

Deterministic Latency PHY Delay Estimation Logic

This section provides the equations to calculate delays when the Deterministic Latency PHY IP core implements CPRI protocol.

This section provides the equations to calculate delays when the Deterministic Latency PHY IP Core implements CPRI protocol. CPRI defines the radio base station interface between network radio equipment controllers (REC) and radio equipment (RE) components.

Example 12-1: For RE

RX _latency_ RE = <R X P CS latency in parallel clock cycles >
+ (<RX PMA latency in UI >
+ < rx_std_bitslipboundaryselect > delay)

TX_latency_RE = <TX PCS latency in parallel clock cycles>
+ <T X PMA latency in U I > + Tx bitslip latency >

Note:

In single width (PMA =10) mode, add one UI delay per value of `rx_std_bitslipboundaryselect`. For constant round-trip delay (RX+TX), set `tx_std_bitslipboundaryselect` <= (5'd9 - `rx_std_bitslipboundaryselect`).

In double width (PMA =20) mode, add one UI delay per value of (5'd9 - `rx_std_bitslipboundaryselect`). For constant round-trip delay (RX+TX), set `tx_std_bitslipboundaryselect` <= `rx_std_bitslipboundaryselect`.

Example 12-2: For REC

For REC

RX_latency_REC = <RX PCS latency in parallel clock cycles>
+ <RX PMA latency in UI> + <rx_clkout phase shift of tx_clkout>

TX_latency_REC = <TX PCS latency in parallel clock cycles>
+ <TX PMA latency in UI>

Example 12-3: For Round Trip Delay

$$\text{Launch\_time (from TX pins)} = \langle\text{clock arrival time}\rangle + \langle\text{data arrival time}\rangle$$
$$= \langle\text{clock arrival time}\rangle + \langle\text{TX latency in REC (tx bitslip=0)}\rangle$$
$$= \langle t_{PD\ GPLL\ to\ CMU\ PLL} - t_{feedback}\rangle + ((\langle\text{TX\_latency in REC}\rangle \times \langle\text{tx\_clkout\_period}\rangle) + t_{TX\_tc\ lock\_output})$$

$$\text{Arrival\_time (at RX pins)} = \langle\text{latency time in RE}\rangle - \langle\text{RX latency time in REC}\rangle$$
$$= (\langle\text{Round\_trip\_latency}\rangle \times \langle\text{tx\_clkout\_period}\rangle) - ((\langle\text{RX\_latency in REC}\rangle \times \langle\text{rx\_clkout\_period}\rangle) + \langle t_{PDIO->RX\_deser}\rangle + \langle\text{rx\_clkout\_phase\_WRT\_tx\_clkout/360} \times \text{rx\_clkout\_period}\rangle)$$

$$\text{Total Delay} = \langle\text{Arrival\_time}\rangle - \langle\text{Launch\_time}\rangle$$

Example 12-4: Total Delay Uncertainty

Round trip delay estimates are subject to process, voltage, and temperature (PVT) variation.

$$t_{RXCLK\_Phase\_detector\_uncertainty} = 2 \times \max(\langle t_{GLL\_phase\_step}\rangle, \langle t_{CDR\_to\_GPLL\_jitter}\rangle) + \mu t_{SU} + \mu t_H$$

$$t_{Round\_trip\_uncertainty} = \langle t_{RX\_CLK\_Phase\_detector\_uncertainty} + t_{GPLL->CMU\ PLL\_variation}\rangle + \langle t_{feedback\_variation}\rangle + \langle t_{TX\_tco\_variation}\rangle + \langle t_{IO->RX\ deser\_delay\_variation}\rangle + \langle t_{PLL\_multicycle\_jitter}\rangle + \langle t_{offset\_uncertainty}\rangle$$

Table 12-1: TX PCS Total Latency

This table shows the total latency through the TX PCS in parallel clock cycles with the byte serializer/deserializer turned off. The TX compensation FIFO is in register mode.

PCS Datapath Width	TX Phase Comp FIFO	Serializer	8B/10B	Bitslip (tx_std_bitslipboundaryselect)[9]	Total TX Parallel Clock Cycles
Byte Serializer/Deserializer Turned Off					
8 bits	1.0	1.0	1.0	0	3.0
16 bits	1.0	1.0	1.0	0	3.0
Byte Serializer/Deserializer Turned On					
16 bits	1.0	0.5	0.5	0	2.0
32 bits	1.0	0.5	0.5	0	2.0

(9) This latency is calculated assuming that the optional tx_std_bitslipboundaryselect is set to zero. Add one UI of latency per value of this port. For example, if tx_std_bitslipboundaryselect is set to one, add one UI of latency to the total.

Table 12-2: RX PCS Total Latency

The RX compensation FIFO is in register mode. When the byte serializer/deserializer in turned on, the latency through is function depends on the location of the alignment pattern. When the alignment pattern is in the upper symbol, the delay is 0.5 cycles. When the alignment pattern is in the lower symbol, the delay is 1.0 cycles.

PCS Datapath Width	RX Phase Comp FIFO	Byte Ordering	Deserial-izer	8B/10B	Word Aligner [11][10]	Total RX Parallel Clock Cycles [10][11]
Byte Serializer/Deserializer Turned Off						
8 bits	1.0	1.0	1.0	1.0	4.0	8.0
16 bits	1.0	1.0	1.0	1.0	5.0	9.0
Byte Serializer/Deserializer Turned On						
16 bits	1.0	1.0	0.5 or 1.0	0.5	2.0	5.0 or 5.5
32 bits	1.0	1.0	0.5 or 1.0	0.5	2.5	5.5 or 6.0

Table 12-3: PMA Datapath Total Latency

The latency numbers in this table are actual hardware delays .

Device	RX PMA Latency in UI		TX PMA Latency in UI	
	PCS to PMA Width 10 bits	PCS to PMA Width 20 bits	PCS to PMA Width 10 bits	PCS to PMA Width 20 bits
Cyclone V	26	31	42	62
Arria V	34	49	52	82
Arria V GZ	26	31	53	83
Stratix V	26	31	53	83

Deterministic Latency PHY Device Family Support

This section describes Deterministic Latency PHY IP core device support.

IP cores provide either final or preliminary support for target Altera device families. These terms have the following definitions:

- Final support—Verified with final timing models for this device.
- Preliminary support—Verified with preliminary timing models for this device.

(10) When the word aligner is in manual mode, and the byte deserializer is turned off, add x UI of latency to the total latency if `rx_std_bitslipboundaryselect` is outputting x. For constant RX + TX latency, set `tx_std_bitslipboundaryselect` = 5'd9 – `rx_std_bitslipboundaryselect`.

(11) When the word aligner is in manual mode, and the byte serializer is turned on, add (19-x) UI of latency to the total latency if `rx_std_bitslipboundaryselect` is outputting x. For constant RX + TX latency, set `tx_std_bitslipboundaryselect` = `rx_std_bitslipboundaryselect`.

Table 12-4: Device Family Support

Device Family	Support
Arria V devices	Final
Arria V GZ devices	Final
Cyclone V devices	Final
Stratix V devices	Final
Other device families	No support

Parameterizing the Deterministic Latency PHY

This section provides a list of steps on how to configure Deterministic Latency PHY

1. Under **Tools** > **IP Catalog**, select the device family of your choice.
2. Under **Tools** > **IP Catalog** > **Interface Protocols** > **Transceiver PHY**, select **Deterministic Latency PHY**.
3. Use the tabs on the MegaWizard Plug-In Manager to select the options required for the protocol.
   a. Set the Deterministic Latency PHY general options parameters.
   b. Set the Deterministic Latency PHY additional options parameters.
   c. Set the Deterministic Latency PHY PLL reconfiguration parameters as required.
   d. Set the Deterministic Latency PHY additional options parameters as required.
4. Click **Finish**.
   Generates your customized Deterministic Latency PHY IP Core.

General Options Parameters for Deterministic Latency PHY

This section describes how to set basic parameters of your transceiver PHY for the Deterministic Latency PHY IP core using the general options tab.

Use the **General Options** tab to set your basic device parameter settings.

Table 12-5: General Options

Name	Value	Description
Device family	**Arria V**, **Cyclone V**, **Stratix V**	Specifies the device family. Arria V, Cyclone V, and Stratix V are available.
Mode of operation	**Duplex**, **TX**, **RX**	You can select to transmit data, receive data, or both.
Number of lanes	**1-32**	The total number of lanes in each direction.
FPGA fabric transceiver interface width	**8, 10, 16, 20, 32, 40**	Specifies the word size between the FPGA fabric and PCS. Refer to **Sample Channel Width Options for Supported Serial Data Rates** for the data rates supported at each word size.

Name	Value	Description
PCS-PMA interface width	**10, 20**	Specifies the datapath width between the transceiver PCS and PMA. A deserializer in the PMA receives serial input data from the RX buffer using the high-speed recovered clock and deserializes it using the low-speed parallel recovered clock.
PLL type	**CMU, ATX**	Specifies the PLL type. The CMU PLL has a larger frequency range than the ATX PLL. The ATX PLL is designed to improve jitter performance and achieves lower channel-to-channel skew; however, it supports a narrower range of data rates and reference clock frequencies. Another advantage of the ATX PLL is that it does not use a transceiver channel, while the CMU PLL does. Because the CMU PLL is more versatile, it is specified as the default setting.
Data rate	Device Dependent	If you select a data rate that is not supported by the configuration you have specified, the MegaWizard displays a error message in the message pane. **Sample Channel Width Options for Supported Serial Data Rates** for sample the channel widths that support these data rates.
Base data rate	1 × **Data rate** 2 × **Data rate** 4 × **Data rate** 8 × **Data rate**	For systems that transmit and receive data at more than one data rate, select a base data rate that minimizes the number of PLLs required to generate the clocks for data transmission. The Recommended Base Data Rate and Clock Divisors for CPRI table lists the recommended **Base data rates** for various **Data rates**. The available options are dynamically computed based on the Data rate you specified as long as those **Base data rates** are within the frequency range of the PLL.
Input clock frequency	Data rate/**20** Data rate/**10** Data rate/**8** Data rate/**5** Data rate/**4** Data rate/**2.5** Data rate/**2** Data rate/**1.25** Data rate/**1**	This is the reference clock for the PHY PLL. The available options are based on the **Base data rate** specified.
Enable tx_clkout feedback path for TX PLL	**On/ Off**	When **On**, the core uses TX PLL feedback to align the TX core clock with the source to the TX PLL which is the RX recovered clock. This configuration is shown in Using TX PLL Feedback to Align the TX Core Clock with the RX Core Clock.

The following table lists the available channel widths available at selected frequencies. The channel width options are restricted by the following maximum FPGA-PCS fabric interface frequencies:

- Arria V devices—153.6 MHz
- Cyclone V devices—153.6 MHz
- Stratix V devices—221 MHz

Table 12-6: Sample Channel Width Options for Supported Serial Data Rates

Serial Data Rate (Mbps)	Channel Width (FPGA-PCS Fabric)			
	Single-Width		Double-Width	
	8-Bit	16-Bit	16-Bit	32-Bit
614.4	Yes	Yes	No	No
1228.8	Yes	Yes	Yes	Yes
2457.6	No	Yes	Yes	Yes
3072	No	Yes	Yes	Yes
4915.2	No	No	No	Yes
6144	No	No	No	Yes

Additional Options Parameters for Deterministic Latency PHY

This section describes the settings available on the Additional Options tab for the Deterministic Latency PHY IP core.

Name	Value	Description
Word alignment mode		The word aligner restores word boundaries of received data based on a predefined alignment pattern. The word aligner automatically performs an initial alignment to the specified word pattern after reset deassertion. You can select 1 of the following 2 modes: **Deterministic latency state machine** or **Manual**

Name	Value	Description
Word alignment mode	**Deterministic latency state machine**	**Deterministic latency state machine**–In this mode, the RX word aligner automatically searches for the word alignment pattern after reset completes. After the word aligner detects the specified word alignment pattern, it sends `RX_CLKSLIP` to the RX PMA deserializer indicating the number of bits to slip to compensate for the bits that were slipped to achieve word alignment. When `RX_CLKSLIP` has a non-zero value, the deserializer either skips one serial bit or pauses the serial clock for one cycle. As a result, the period of the parallel clock could be extended by 1 unit interval (UI) during the clock slip operation. This procedure avoids using the TX bit slipper to ensure constant round-trip delay.
		In this mode, the specified word alignment pattern, which is currently forced to K28.5 (0011111010) is always placed in the least significant byte (LSB) of a word with a fixed latency of 3 cycles. User logic can assume the LSB placement. Altera recommends the deterministic latency state machine mode for new designs.
		During the word alignment process, the parallel clock shifts the phase to align to the data. This phase shifting will be 2/10 cycles (20%) in 10 bit mode, 2/20 cycles (10%) in 20 bit mode, and 2/40 cycles (5%) in 40 bit mode.
		For double-width datapaths using deterministic latency state machine mode, after the initial alignment following the deassertion of reset, the Avalon-MM register big `rx_enapatternalign` (not available as a signal) must be reasserted to initiate another pattern alignment. Asserting `rx_enapatternalign`, may cause the extra shifting in the RX datapath if `rx_enablepatternalign` is asserted while bit slipping is in progress; consequently `rx_enapatternalign` should only be asserted under the following conditions: • `rx_syncstatus` is asserted • `rx_bitslipboundaryselectout` changes from a non-zero value to zero or 1
Word alignment mode	**Manual**	**Manual**–In this mode, the RX word aligner parses the incoming data stream for a specific alignment character. After it identifies this pattern, it shifts the input stream to align the data and also outputs the number of bits slipped on `bitslip-boundaryselectout[4:0]` for latency compensation on the TX datapath. This mode is provided for backwards compatibility with designs implemented in Stratix IV and Arria II devices.

Name	Value	Description
TX bitslip	**On/ Off**	TX bitslip is enabled whenever the word aligner is in Manual alignment mode. The TX bitslipper uses the value of `bitslipboundarselect[4:0]` to compensate for bits slipped on the RX datapath to achieve deterministic latency.
Enable run length violation checking	**On/ Off**	If you turn this option on, you can specify the run length which is the maximum legal number of contiguous 0s or 1s. This option also creates the rx_rlv output signal which is asserted when a run length violation is detected.
Run length	**5-160**	Specifies the threshold for a run-length violation. Must be a multiple of 5.
Create optional word aligner status ports	**On/ Off**	Enable this option to include the `rx_patterndetect` and `rx_syncstatus` ports.
Create optional 8B/ 10B control and status ports	**On/ Off**	Enable this option to include the 8B/10B `rx_runningdisp`, `rx_errdetect`, and `rx_disperr` signals at the top level of the Deterministic Latency PHY IP Core.
Create PMA optional status ports	**On/ Off**	Enable this option to include the 8B/10B `rx_is_lockedtoref`, `rx_is_lockedtodata`, and `rx_signaldetect` signals at the top level of the Deterministic Latency PHY IP Core.
Avalon data interfaces	**On/ Off**	This option is typically required if you are planning to import your Deterministic Latency PHY IP Core into a Qsys system.
Enable embedded reset controller	**On/ Off**	When you turn this option On, the embedded reset controller handles reset of the TX and RX channels at power up. If you turn this option Off, you must design a reset controller that manages the following reset signals: `tx_digitalreset`, `tx_analogreset`, `tx_cal_busy`, `rx_digitalreset`, `rx_analogreset`, and `rx_cal_busy`. You may also use the Transceiver PHY Reset Controller to reset the transceivers. For more information, refer to the Transceiver Reconfiguration Controller IP Core.

Related Information

- **Transceiver Reconfiguration Controller IP Core Overview** on page 17-1
- **Transceiver Architecture in Arria V Devices**
- **Transceiver Architecture in Cyclone V Devices**
- **Transceiver Architecture in Stratix V Devices**

PLL Reconfiguration Parameters for Deterministic Latency PHY

The section describes the PLL Reconfiguration options for the Deterministic Latency PHY IP core.

This table lists the PLL Reconfiguration options. For more information about transceiver reconfiguration registers, refer to PLL Reconfiguration.

Table 12-7: PLL Reconfiguration Options

Name	Value	Description
Allow PLL/CDR Reconfiguration	**On/Off**	You must enable this option if you plan to reconfigure the PLLs in your design. This option is also required to simulate PLL reconfiguration.
Number of TX PLLs	Device dependent	Specifies the number of TX PLLs that can be used to dynamically reconfigure channels to run at multiple data rates. If your design does not require transceiver TX PLL dynamic reconfiguration, set this value to 1. The number of actual physical PLLs that are implemented depends on the selected clock network. Each channel can dynamically select between n PLLs, where n is the number of PLLs specified for this parameter. **Note:** For more details, refer to the *Transceiver Clocking* chapter in the device handbook for the device family you are using.
Number of reference clocks	**1-5**	Specifies the number of input reference clocks. More than one reference clock may be required if your design reconfigures channels to run at multiple frequencies.
Main TX PLL logical index	**0-3**	Specifies the index for the TX PLL that should be instantiated at startup. Logical index 0 corresponds to TX PLL0, and so on.
Main TX PLL input clock source	**0-3**	Specifies the index for the TX PLL input clock that should be instantiated at startup. Logical index 0 corresponds to input clock 0 and so on.
CDR PLL input clock source	**0-4**	Specifies the index for the CDR PLL input clock that should be instantiated at startup. Logical index 0 corresponds to input clock 0 and so on.
TX PLL (0–3) (Refer to General Options for a detailed explanation of these parameters.)		
PLL Type	**CMU**	Specifies the PLL type.
Base data rate	1 × **Lane rate** 2 × **Lane rate** 4 × **Lane rate**	Specifies **Base data rate**.
Input clock frequency	Variable	Specifies the frequency of the PLL input reference clock. The PLL must generate an output frequency that equals the **Base data rate**/2. You can use any **Input clock frequency** that allows the PLLs to generate this output frequency.

Name	Value	Description
Selected input clock source	**0-4**	Specifies the index of the input clock for this TX PLL. Logical index 0 corresponds to input clock 0 and so on.
Channel Interface		
Enable channel interface	**On/Off**	Turn this option on to enable PLL and datapath dynamic reconfiguration. When you select this option, the width of `tx_parallel_data` and `rx_parallel_data` buses increases in the following way: • The `rx_parallel_data` bus is 64 bits per lane; however, only the low-order number of bits specified by the FPGA fabric transceiver interface width contain valid data. • The `tx_parallel_databus` is 44 bits per lane; however, only the low-order number of bits specified by the FPGA fabric transceiver interface width contain valid data for each lane.

Related Information

Transceiver Reconfiguration Controller PLL Reconfiguration on page 17-28

Deterministic Latency PHY Analog Parameters

This section provides links to describe analog parameters for the Deterministic Latency PHY IP core.

The following links provide information to specify the analog options for your device:

Related Information

- **Analog Settings for Arria V Devices** on page 20-2
- **Analog Settings for Arria V GZ Devices** on page 20-11
- **Analog Settings for Cyclone V Devices** on page 20-26
- **Analog Settings for Stratix V Devices** on page 20-35

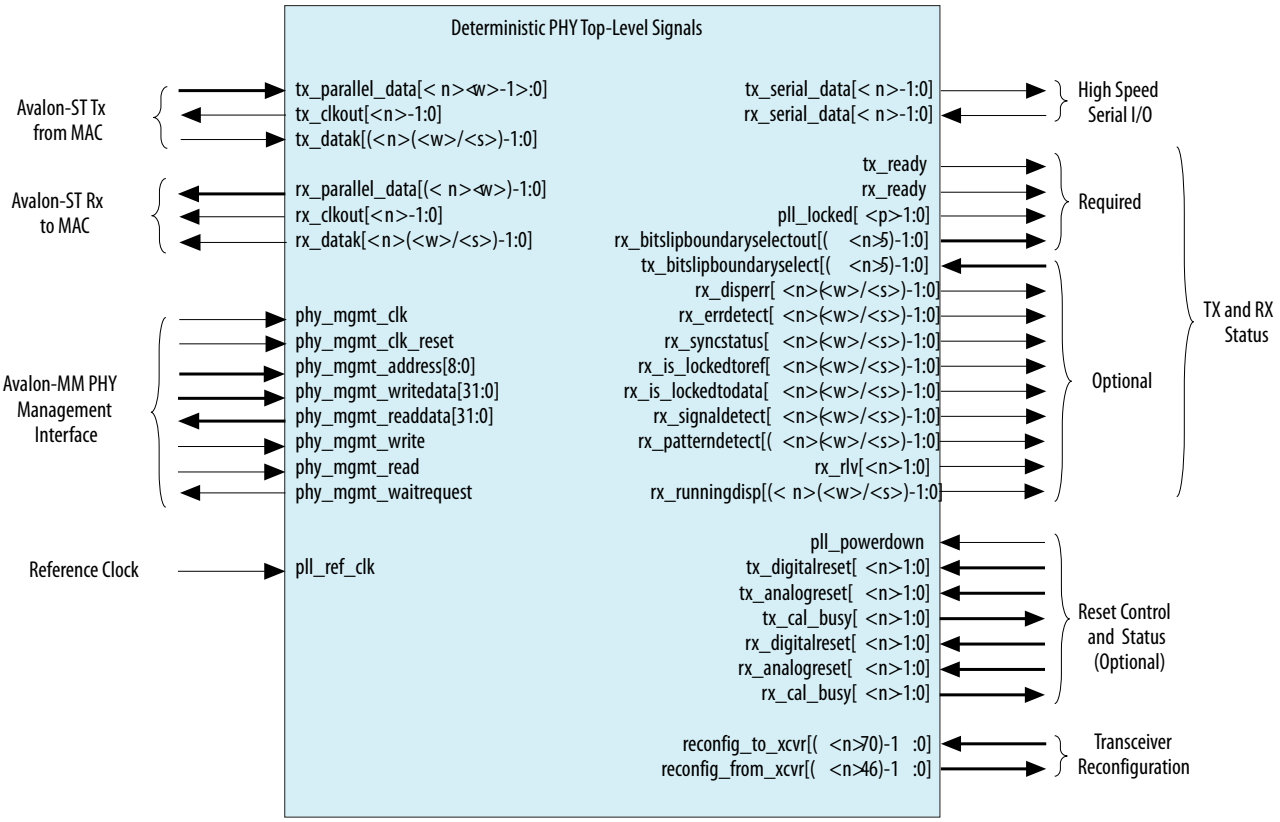
Interfaces for Deterministic Latency PHY

This section describes the top-level signals of the Deterministic Latency PHY IP Core.

The following figure illustrates the top-level signals of the Deterministic Latency PHY IP Core. The variables in the figure represent the following parameters:

- <n>—The number of lanes
- <w>—The width of the FPGA fabric to transceiver interface per lane
- <s>— The symbol size
- <p>—The number of PLLs

Figure 12-3: Deterministic Latency PHY Top-Level Signals



The **block diagram** shown in the MegaWizard Plug-In Manager labels the external pins with the interface type and places the interface name inside the box. The interface type and name are used in the **_hw.tcl** file that describes the component. If you turn on **Show signals**, the **block diagram** displays all top-level signal names.

Related Information

Component Interface Tcl Reference

Data Interfaces for Deterministic Latency PHY

This section describes the signals Avalon_ST protocol, output interface, and the differential serial data interface for the Deterministic Latency PHY IP core.

For more information about the Avalon-ST protocol, including timing diagrams, refer to the Avalon Interface Specifications.

Table 12-8: Avalon-ST TX Interface

The following table describes the signals in the Avalon-ST input interface. These signals are driven from the MAC to the PCS. This is an Avalon sink interface.

Signal Name	Direction	Description
tx_parallel_data[(<n><w>)-1:0]	Input	This is TX parallel data driven from the MAC. The ready latency on this interface is 0, so that the PHY must be able to accept data as soon as it comes out of reset. Refer to for definitions of the control and status signals with 8B/10B encoding enabled and disabled. Refer to Table 11-11 for the signals that correspond to data, control, and status signals.
tx_clkout[<n>-:0]	Output	This is the clock for TX parallel data, control, and status signals.
tx_datak[(<n><d>/<s>)-1:0]	Input	Data and control indicator for the transmitted data. When 0, indicates that tx_parallel_data is data, when 1, indicates that tx_parallel_data is control.

Table 12-9: Signal Definitions for tx_parallel_data with and without 8B/10B Encoding

The following table shows the signals within tx_parallel_data that correspond to data, control, and status signals.

TX Data Word	Description
Signal Definitions with 8B/10B Enabled	
tx_parallel_data[7:0]	TX data bus
tx_parallel_data[8]	TX data control character
tx_parallel_data[9]	Force disparity, validates disparity field.
tx_parallel_data[10]	Specifies the current disparity as follows: • 1'b0 = positive • 1'b1 = negative
Signal Definitions with 8B/10B Disabled	
tx_parallel_data[9:0]	TX data bus
tx_parallel_data[10]	Unused

Table 12-10: Avalon-ST RX Interface

The following table describes the signals in the Avalon-ST output interface. These signals are driven from the PCS to the MAC. This is an Avalon source interface.

Signal Name	Direction	Description
rx_parallel_data [(<n><d>)-1:0]	Output	This is RX parallel data driven from the Deterministic Latency PHY IP Core. The ready latency on this interface is 0, so that the MAC must be able to accept data as soon as the PHY comes out of reset. Data driven from this interface is always valid. Refer to the following "Signal Definitions for rx_parallel_data with and without 8B/10B Encoding" table for the signals that correspond to data, control, and status signals.
rx_clkout[<n>-1:0]	Output	This is the clock for the RX parallel data source interface.
rx_datak[(<n>(<d>/<s>)-:0]	Output	Data and control indicator for the source data. When 0, indicates that rx_parallel_data is data, when 1, indicates that rx_parallel_data is control.

Table 12-11: Signal Definitions for rx_parallel_data with and without 8B/10B Encoding

This table shows the signals within rx_parallel_data that correspond to data, control, and status signals.

RX Data Word	Description
Signal Definitions with 8B/10B Enabled	
rx_parallel_data[7:0]	RX data bus
rx_parallel_data[8]	RX data control character
rx_parallel_data[9]	Error Detect
rx_parallel_data[10]	Word Aligner / synchronization status
rx_parallel_data[11]	Disparity error
rx_parallel_data[12]	Pattern detect
rx_parallel_data[14:13]	The following encodings are defined: • 2'b00: Normal data • 2'b01: Deletion • 2'b10: Insertion • 2'b11: Underflow
rx_parallel_data[15]	Running disparity value
Signal Definitions with 8B/10B Disabled	
rx_parallel_data[9:0]	RX data bus
rx_parallel_data[10]	Word Aligner / synchronization status
rx_parallel_data[11]	Disparity error
rx_parallel_data[12]	Pattern detect

RX Data Word	Description
rx_parallel_data[14:13]	The following encodings are defined: • 2'b00: Normal data • 2'b01: Deletion • 2'b10: Insertion (or Underflow with 9'h1FE or 9'h1F7) • 2'b11: Overflow
rx_parallel_data[15]	Running disparity value

Table 12-12: Serial Interface and Status Signals

This table describes the differential serial data interface and the status signals for the transceiver serial data interface. <*n*> is the number of lanes.

Signal Name	Direction	Signal Name
rx_serial_data[<*n*>-:0]	Input	Receiver differential serial input data.
tx_serial_data[<*n*>-:0]	Output	Transmitter differential serial output data.

Related Information

Avalon Interface Specifications

Clock Interface for Deterministic Latency PHY

This section describes the clocks for the Deterministic Latency PHY IP core.

The following table describes clocks for the Deterministic Latency PHY. The input reference clock, pll_ref_clk, drives a PLL inside the PHY-layer block, and a PLL output clock, rx_clkout is used for all data, command, and status inputs and outputs.

Table 12-13: Clock Signals

Signal Name	Direction	Description
pll_ref_clk	Input	Reference clock for the PHY PLLs. Frequency range is 60-700 MHz.

Optional TX and RX Status Interface for Deterministic Latency PHY

This section describes the optional TX and RX status interface settings for the Deterministic Latency PHY IP core.

Table 12-14: Serial Interface and Status Signals

Signal Name	Direction	Signal Name
tx_ready	Output	When asserted, indicates that the TX interface has exited the reset state and is ready to transmit.
rx_ready	Output	When asserted, indicates that the RX interface has exited the reset state and is ready to receive.
pll_locked [<*p*>-1:0]	Output	When asserted, indicates that the PLL is locked to the input reference clock.
rx_bitslipboundaryselectout [(<*n*>5)-1:0]	Output	Specifies the number of bits slipped to achieve word alignment. In 3G (10-bit) mode, the output is the number of bits slipped. If no bits were slipped, the output is 0. In 6G (20-bit) mode, the output is (19 - the number of bits slipped). If no bits were slipped, the output is 19. The default value of rx_bitslipboundaryselectout[4:0] before alignment is achieved is 5'b01111 in 3G mode and 5'b11111 in 6G mode.
Optional Status Signals		
tx_bitslipboundaryselect [(<*n*>5)-1:0]	Input	This signal is used for bit slip word alignment mode. It selects the number of bits that the TX block must slip to achieve a deterministic latency.
rx_disperr [(<*n*><*d*>/<*s*>)-1:0]	Output	When asserted, indicates that the received 10-bit code or data group has a disparity error.
rx_errdetect [(<*n*><*d*>/<*s*>)-1:0]	Output	When asserted, indicates that a received 10-bit code group has an 8B/10B code violation or disparity error.
rx_syncstatus [(<*n*><*d*>/<*s*>)-1:0]	Output	Indicates presence or absence of synchronization on the RX interface. Asserted when word aligner identifies the word alignment pattern or synchronization code groups in the received data stream. This signal is optional.

Signal Name	Direction	Signal Name
rx_is_lockedtoref [(<n>(<d>/<s>) -1:0]	Output	Asserted when the receiver CDR is locked to the input reference clock. This signal is asynchronous. This signal is optional.
rx_is_lockedtodata [(<n><d>/<s>) -1:0]	Output	When asserted, the receiver CDR is in to lock-to-data mode. When deasserted, the receiver CDR lock mode depends on the rx_locktorefclk signal level. This signal is optional.
rx_patterndetect [(<n>(<d>/<s>)-1:0]	Output	When asserted, indicates that the programmed word alignment pattern has been detected in the current word boundary.
rx_rlv [<n> -1:0]	Output	When asserted, indicates a run length violation. Asserted if the number of consecutive 1s or 0s exceeds the number specified using the MegaWizard Plug-In Manager.
rx_runningdisp [(<n>(<d>/<s>)-1:0]	Output	This status signal indicates the disparity of the incoming data.

Optional Reset Control and Status Interfaces for Deterministic Latency PHY

The following table describes the signals in the optional reset control and status interface. These signals are available if you do not enable the embedded reset controller.

Table 12-15: Avalon-ST RX Interface

Signal Name	Direction	Description
pll_powerdown [<n>-1:0]	Input	When asserted, resets the TX PLL.
tx_digitalreset [<n>-1:0]	Input	When asserted, reset all blocks in the TX PCS.
tx_analogreset [<n>-1:0]	Input	When asserted, resets all blocks in the TX PMA.
tx_cal_busy [<n>-1:0]	Output	When asserted, indicates that the initial TX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP. You must hold the channel in reset until calibration completes.
rx_digitalreset [<n>-1:0]	Input	When asserted, resets the RX PCS.

Signal Name	Direction	Description
rx_analogreset [<*n*>-1:0]	Input	When asserted, resets the RX CDR.
rx_cal_busy [<*n*>-1:0]	Output	When asserted, indicates that the initial RX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP.

Related Information

- **Transceiver Reset Control in Arria V Devices**
- **Transceiver Reset Control in Cyclone V Devices**
- **Transceiver Reset Control in Stratix V Devices**

Register Interface and Descriptions for Deterministic Latency PHY

Describes the register interface and descriptions for the Deterministic Latency PHY IP core.

The Avalon-MM PHY management interface provides access to the Deterministic Latency PHY PCS and PMA registers that control the TX and RX channels, the PMA powerdown and PLL registers, and loopback modes.

The following figure illustrates the role of the PHY Management module in the Deterministic Latency PHY.

Figure 12-4: Deterministic Latency PHY IP Core

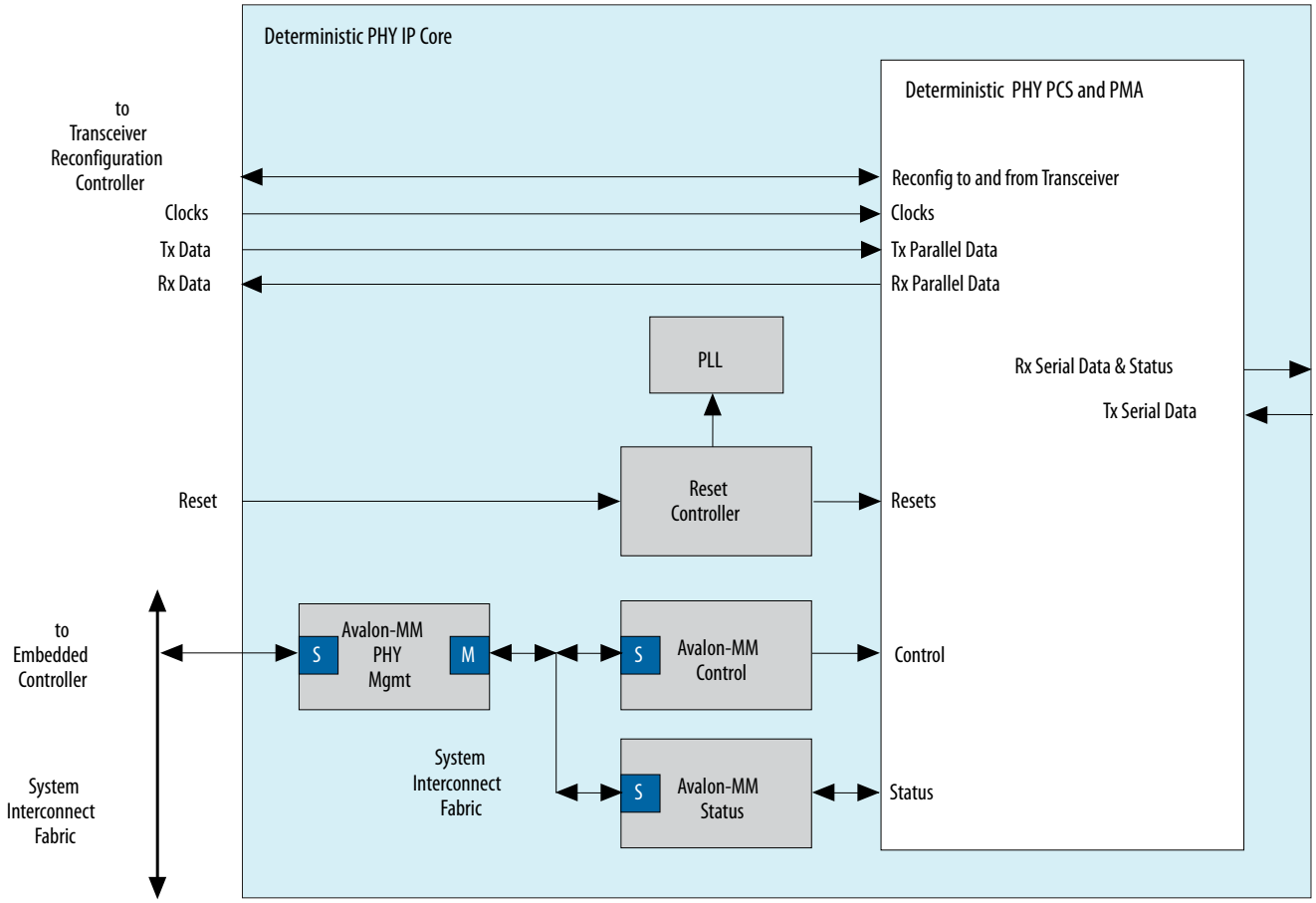


Table 12-16: Avalon-MM PHY Management Interface

Signal Name	Direction	Description
`phy_mgmt_clk`	Input	Avalon-MM clock input. There is no frequency restriction for Stratix V devices; however, if you plan to use the same clock for the PHY management interface and transceiver reconfiguration, you must restrict the frequency range of `phy_mgmt_clk` to 100-150 MHz to meet the specification for the transceiver reconfiguration clock.
`phy_mgmt_clk_reset`	Input	Global reset signal. This signal is active high and level sensitive.
`phy_mgmt_address[8:0]`	Input	9-bit Avalon-MM address.
`phy_mgmt_writedata[31:0]`	Input	Input data.
`phy_mgmt_readdata[31:0]`	Output	Output data.
`phy_mgmt_write`	Input	Write signal.
`phy_mgmt_read`	Input	Read signal.

Signal Name	Direction	Description
phy_mgmt_waitrequest	Output	When asserted, indicates that the Avalon-MM slave interface is unable to respond to a read or write request. When asserted, control signals to the Avalon-MM slave interface must remain constant.

Note: Writing to reserved or undefined register addresses may have undefined side effects.

This table specifies the registers that you can access over the PHY management interface using word addresses and a 32-bit embedded processor. A single address space provides access to all registers.

Table 12-17: Deterministic Latency PHY IP Core Registers

Word Addr	Bits	R/W	Register Name	Description
PMA Common Control and Status Registers				
0x021	[31:0]	RW	cal_blk_powerdown	Writing a 1 to channel < *n* > powers down the calibration block for channel < *n* > .
0x022	[31:0]	R	pma_tx_pll_is_locked	Bit[P] indicates that the TX CMU PLL (P) is locked to the input reference clock. There is typically one pma_tx_pll_is_locked bit per system.
Reset Control Registers–Automatic Reset Controller				
0x041	[31:0]	RW	reset_ch_bitmask	Reset controller channel bitmask for digital resets. The default value is all 1s. Channel < *n* > can be reset when bit< *n* > = 1.
0x42	[1:0]	W	reset_control (write)	Writing a 1 to bit 0 initiates a TX digital reset using the reset controller module. The reset affects channels enabled in the reset_ch_bitmask . Writing a 1 to bit 1 initiates a RX digital reset of channels enabled in the reset_ch_bitmask .
		R	reset_status (read)	Reading bit 0 returns the status of the reset controller TX ready bit. Reading bit 1 returns the status of the reset controller RX ready bit.
Reset Controls –Manual Mode				

Word Addr	Bits	R/W	Register Name	Description
0x044	[31:0]	RW	reset_fine_control	You can use the reset_fine_control register to create your own reset sequence. In manual mode, only the TX reset occurs automatically at power on and when the phy_mgmt_clk_reset is asserted. When pma_rx_setlocktodata or pma_rx_setlocktodata is set, the transceiver PHY is placed in manual mode.
	[31:4,0]	RW	Reserved	It is safe to write 0s to reserved bits.
	[3]	RW	reset_rx_digital	Writing a 1 causes the internal RX digital reset signal to be asserted, resetting the RX digital channels enabled in reset_ch_bitmask . You must write a 0 to clear the reset condition.
	[2]	RW	reset_rx_analog	Writing a 1 causes the internal RX analog reset signal to be asserted, resetting the RX analog logic of all channels enabled in reset_ch_bitmask . You must write a 0 to clear the reset condition.
	[1]	RW	reset_tx_digital	Writing a 1 causes the internal TX digital reset signal to be asserted, resetting all channels enabled in reset_ch_bitmask . You must write a 0 to clear the reset condition.
PMA Control and Status Registers				
0x061	[31:0]	RW	phy _ serial _ loopback	Writing a 1 to channel < *n* > puts channel < *n* > in serial loopback mode. For information about pre- or post-CDR serial loopback modes, refer to Loopback Modes.
0x064	[31:0]	RW	pma_rx_set_locktodata	When set, programs the RX CDR PLL to lock to the incoming data. Bit < *n*> corresponds to channel < *n*>.
0x065	[31:0]	RW	pma_rx_set_locktoref	When set, programs the RX CDR PLL to lock to the reference clock. Bit < *n*> corresponds to channel < *n*>.
0x066	[31:0]	RO	pma_rx_is_lockedtodata	When asserted, indicates that the RX CDR PLL is locked to the RX data, and that the RX CDR has changed from LTR to LTD mode. Bit <n> corresponds to channel <n>.

Word Addr	Bits	R/W	Register Name	Description
0x067	[31:0]	RO	pma_rx_is_lockedtoref	When asserted, indicates that the RX CDR PLL is locked to the reference clock. Bit < *n*> corresponds to channel < *n*>.
			PCS	
0x080	[31:0]	RW	Lane or group number	Specifies lane or group number for indirect addressing, which is used for all PCS control and status registers. For variants that stripe data across multiple lanes, this is the logical group number. For non-bonded applications, this is the logical lane number.
0x081	[31:6]	R	pcs8g_rx_status	Reserved.
	[5:1]	R	rx_bitslipboundaryselectout	This is an output from the bit slip word aligner which shows the number of bits slipped. **From block:** Word aligner.
	[0]	R	Reserved.	-
0x082	[31:1]	R	pcs8g_tx_status	Reserved.
	[0]	RW	Reserved	-
0x083	[31:6]	RW	pcs8g_tx_control	Reserved.
	[5:1]	RW	tx_bitslipboundary_select	Sets the number of bits that the TX bit slipper needs to slip. **To block:** Word aligner.
	[0]	RW	tx_invpolarity	When set, the TX interface inverts the polarity of the TX data. **To block**: 8B/10B encoder.
0x084	[31:1]	RW	Reserved.	-
	[0]	RW	rx_invpolarity	When set, the RX channels inverts the polarity of the received data. **To block:** 8B/10B decoder.

Word Addr	Bits	R/W	Register Name	Description
0x085	[31:4]	RW	pcs8g_rx_wa_control	Reserved.
	[3]	RW	rx_bitslip	Every time this register transitions from 0 to 1, the RX data slips a single bit. **To block:** Word aligner.
	[2]	RW	rx_bytereversal_enable	When set, enables byte reversal on the RX interface. **To block:** Byte deserializer RX Phase Comp FIFO.
	[1]	RW	rx_bitreversal_enable	When set, enables bit reversal on the RX interface. **To block:** Word aligner.
	[0]	RW	rx_enapatternalign	When set in manual word alignment mode, the word alignment logic begins operation when this bit is set. **To block:** Word aligner.

Related Information

Loopback Modes on page 17-59

Dynamic Reconfiguration for Deterministic Latency PHY

Dynamic reconfiguration compensates for circuit variations due to process, voltage, and temperature (PVT).

These process variations result in analog voltages that can be offset from required ranges. The calibration performed by the dynamic reconfiguration interface compensates for variations due to PVT.

Each channel and each TX PLL has a separate dynamic reconfiguration interfaces. The MegaWizard Plug-In Manager provides informational messages on the connectivity of these interfaces. The following example shows the messages for a single duplex channel.

Although you must initially create a separate reconfiguration interface for each channel and TX PLL in your design, when the Intel Quartus Prime software compiles your design, it reduces the number of reconfiguration interfaces by merging reconfiguration interfaces. The synthesized design typically includes a reconfiguration interface for at least three channels because three channels share an Avalon-MM slave interface which connects to the Transceiver Reconfiguration Controller IP Core. Conversely, you cannot connect the three channels that share an Avalon-MM interface to different Transceiver Reconfiguration Controller IP Cores. Doing so causes a Fitter error. For more information, refer to Transceiver Reconfiguration Controller to PHY IP Connectivity.

Example 12-5: Information Messages for the Transceiver Reconfiguration Interface

```
PHY IP will require 2 reconfiguration interfaces for
            connection to the external reconfiguration controller.
Reconfiguration interface offset 0 is connected to the
            transceiver channel.
Reconfiguration interface offset 1 is connected to the
            transmit PLL.
```

Table 12-18: Reconfiguration Interface

This table lists the signals in the reconfiguration interface. This interface uses the Avalon-MM PHY Management interface clock.

Signal Name	Direction	Description
`reconfig_to_xcvr [(<n>70)-1:0]`	Input	Reconfiguration signals from the Transceiver Reconfiguration Controller. <*n*> grows linearly with the number of reconfiguration interfaces.
`reconfig_from_xcvr [(<n>46)-1:0]`	Output	Reconfiguration signals to the Transceiver Reconfiguration Controller. <*n*> grows linearly with the number of reconfiguration interfaces.

Related Information

Transceiver Reconfiguration Controller to PHY IP Connectivity on page 17-57

Channel Placement and Utilization for Deterministic Latency PHY

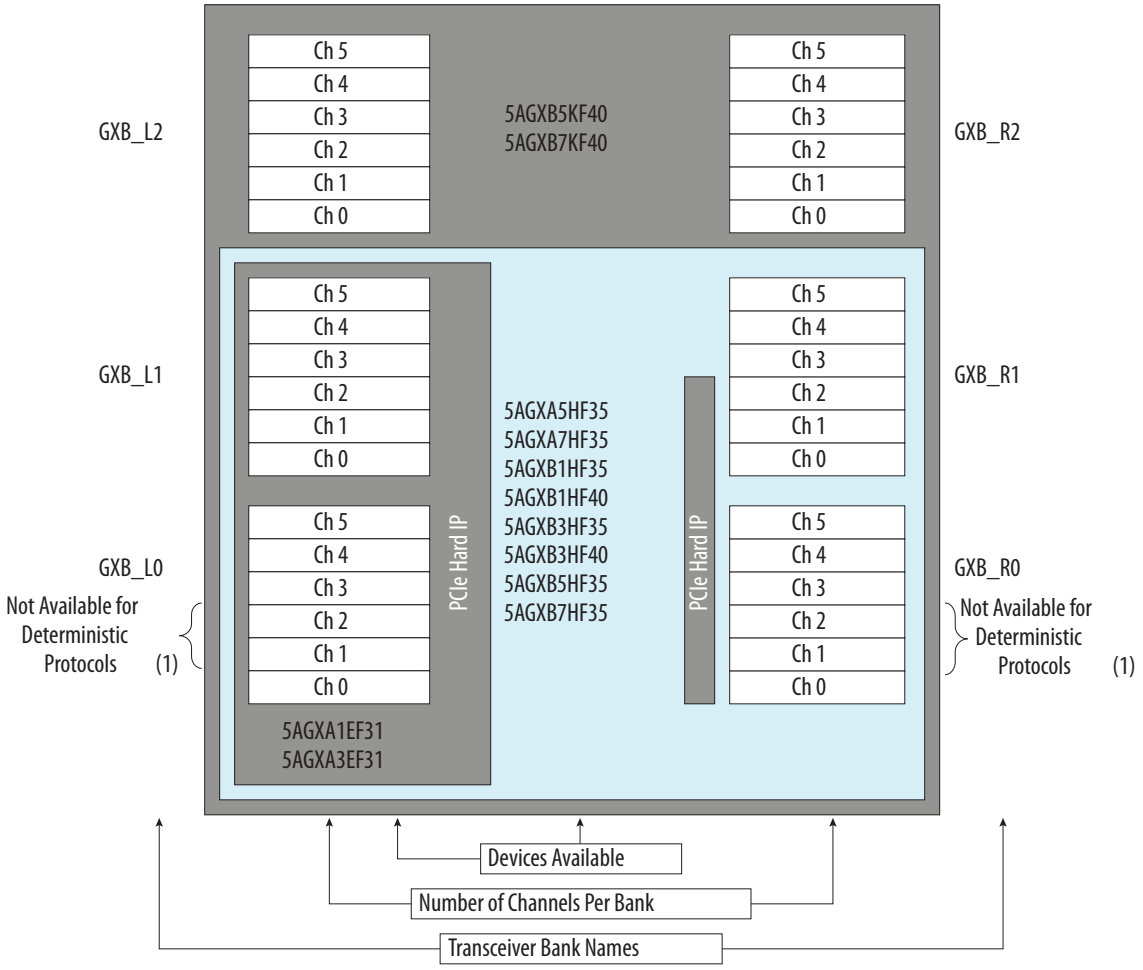
This section describes the channel placement utilization restrictions for the Deterministic Latency PHY IP core.

The Deterministic Latency PHY IP Core has the following restriction on channel placement:

- Channels 1 and 2 in transceiver banks GXB_L0 and GXB_R0 of Arria V devices are not available for deterministic latency protocols. However, in Arria V GZ devices, these channels are available for deterministic latency protocols.

The following figure shows the placement of transceiver banks in Arria V devices and indicates the channels that are not available.

Figure 12-5: Channel Placement and Available Channels in Arria V Devices



Note:

(1) In Arria V GZ devices, channel 1 and 2 are available for deterministic latency protocols.

SDC Timing Constraints

The SDC timing constraints and approaches to identify false paths listed for Stratix V Native PHY IP apply to all other transceiver PHYs listed in this user guide. Refer to *SDC Timing Constraints of Stratix V Native PHY* for details.

Related Information

SDC Timing Constraints of Stratix V Native PHY on page 13-72
This section describes SDC examples and approaches to identify false timing paths.

Simulation Files and Example Testbench for Deterministic Latency PHY

This section describes simulation file requirements for the Deterministic Latency PHY IP core.

Refer to *Running a Simulation Testbench* for a description of the directories and files that the Intel Quartus Prime software creates automatically when you generate your Deterministic Latency PHY IP Core.

Related Information

Running a Simulation Testbench

Stratix V Transceiver Native PHY IP Core 13

2020.06.02

UG-01080 Subscribe Send Feedback

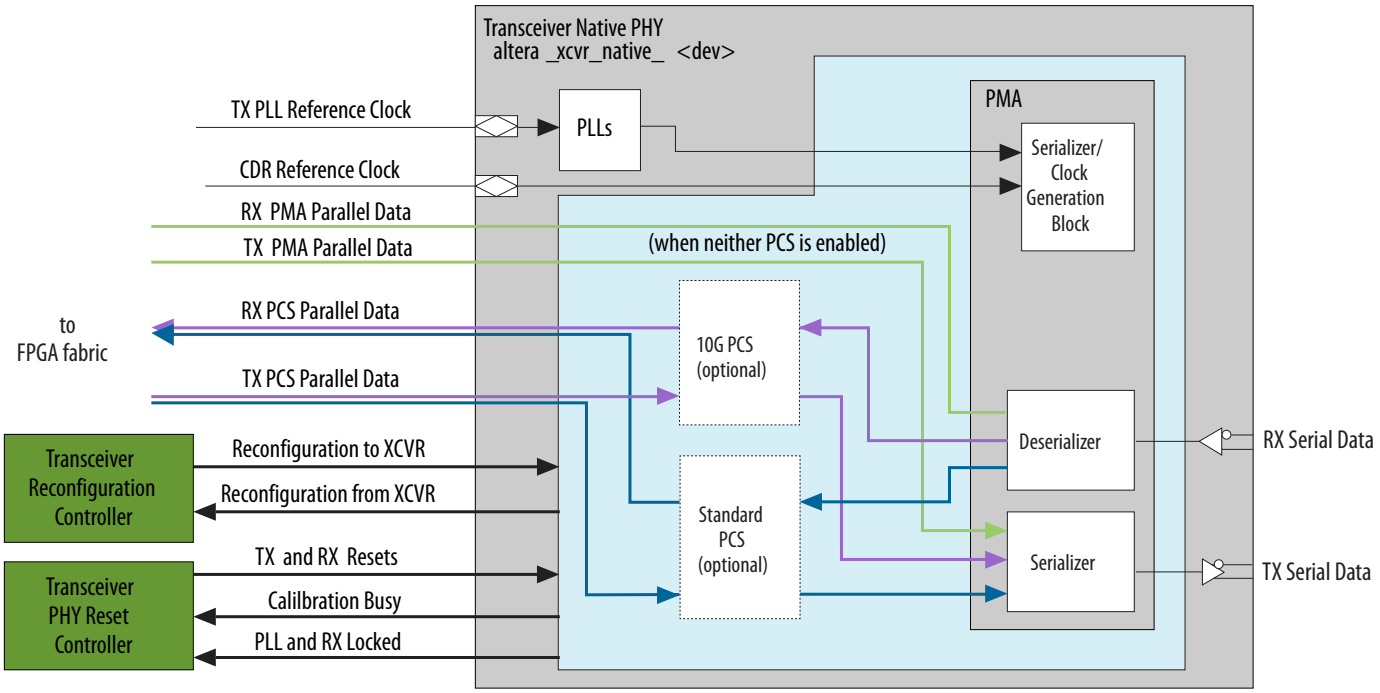
The Stratix V Transceiver Native PHY IP Core provides direct access to all control and status signals of the transceiver channels. Unlike protocol-specific PHY IP Cores, the Native PHY IP Core does not include an Avalon Memory-Mapped (Avalon-MM) interface. Instead, it exposes all signals directly as ports. The Stratix V Transceiver Native PHY IP Core provides the following three datapaths:

- Standard PCS
- 10G PCS
- PMA Direct

You can enable the Standard PCS, the 10G PCS, or both if your design uses the Transceiver Reconfiguration Controller to change dynamically between the two PCS datapaths. The transceiver PHY does not include an embedded reset controller. You can either design custom reset logic or incorporate Altera's "Transceiver PHY Reset Controller IP Core" to implement reset functionality. In PMA Direct mode, the Native PHY provides direct access to the PMA from the FPGA fabric; consequently, the latency for transmitted and received data is very low. However, you must implement any PCS function that your design requires in the FPGA fabric.

The following figure illustrates the use of the Stratix V Transceiver Native PHY IP Core. As this figure illustrates, TX PLL and clock data recovery (CDR) reference clocks from the pins of the device are input to the PLL module and CDR logic. When enabled, the 10G or Standard PCS drives TX parallel data and receives RX parallel data. When neither PCS is enabled the Native PHY operates in PMA Direct mode.

Figure 13-1: Stratix V Native Transceiver PHY IP Core



In a typical design, the separately instantiated Transceiver PHY Reset Controller drives reset signals to Native PHY and receives calibration and locked status signal from the Native PHY. The Native PHY reconfiguration buses connect the external Transceiver Reconfiguration Controller for calibration and dynamic reconfiguration of the PLLs.

You specify the initial configuration when you parameterize the IP core. The Transceiver Native PHY IP Core connects to the Transceiver Reconfiguration Controller IP Core to dynamically change reference clocks and PLL connectivity at runtime.

Device Family Support for Stratix V Native PHY

This section describes the device family support available in the Stratix V native PHY.

IP cores provide either final or preliminary support for target Altera device families. These terms have the following definitions:

- Final support—Verified with final timing models for this device.
- Preliminary support—Verified with preliminary timing models for this device.

Table 13-1: Device Family Support

This tables lists the level of support offered by the Stratix V Transceiver Native PHY IP Core for Altera device families.

Device Family	Support
Stratix V devices	Final

Device Family	Support
Other device families	No support

Performance and Resource Utilization for Stratix V Native PHY

This section describes the performance resource utilization for Stratix V native PHY.

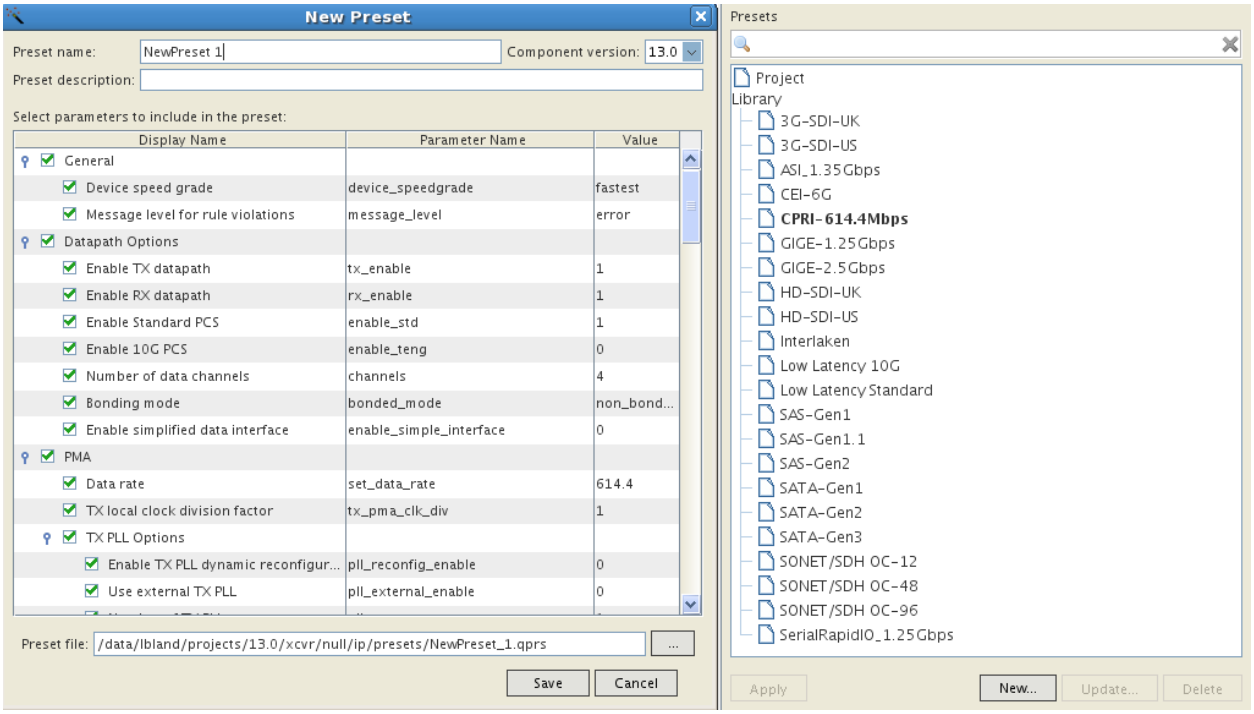
Because the 10G PCS, Standard PCS, and PMA are implemented in hard logic, the Stratix V Native PHY IP Core uses less than 1% of the available ALMs, memory, primary and secondary logic registers.

Parameter Presets

Presets allow you to specify a group of parameters to implement a particular protocol or application.

If you apply a preset, the parameters with specific required values are set for you. When applied, the preset is in boldface and remains as such unless you change some of the preset parameters. Selecting a preset does not prevent you from changing any parameter to meet the requirements of your design. The following figure illustrates the Preset panel and form to create custom presets.

Figure 13-2: Preset Panel and Form To Create Custom Presets



Parameterizing the Stratix V Native PHY

This section provides a list of instructions on how to configure the Stratix V Native PHY IP core

Complete the following steps to configure the Stratix V Native PHY IP Core

1. Under **Tools** > **IP Catalog**, select **Stratix V** as the device family.
2. Under **Tools** > **IP Catalog** > **Interface Protocols** > **Transceiver PHY**, select **Stratix V Native PHY**.
3. Use the tabs on the MegaWizard Plug-In Manager to select the options required for the protocol.
4. Click **Finish**.
   Generates your customized Stratix V Native PHY IP Core.

General Parameters for Stratix V Native PHY

This section describes the datapath parameters in the General Options tab for the Stratix V native PHY.

Table 13-2: General and Datapath Options

The following table lists the parameters available on the General Options tab. Note that you can enable the Standard PCS, the 10G PCS, or both if you intend to reconfigure between the two available PCS datapaths.

Name	Range	Description
Device speed grade	**fastest - 3_H3**	Specifies the speed grade.
Message level for rule violations	**error** **warning**	When you select the **error** message level, the Intel Quartus Prime rules checker reports an error if you specify incompatible parameters. If you select the **warning** message level, the Intel Quartus Prime rules checker reports a warning instead of an error.
Datapath Options		
Enable TX datapath	**On/Off**	When you turn this option **On**, the core includes the TX datapath.
Enable RX datapath	**On/Off**	When you turn this option **On**, the core includes the RX datapath.
Enable Standard PCS	**On/Off**	When you turn this option **On**, the core includes the Standard PCS . You can enable both the Standard and 10G PCS if you plan to dynamically reconfigure the Native PHY.
Enable 10G PCS	**On/Off**	When you turn this option **On**, the core includes the 10G PCS. You can enable both the **Standard** and **10G PCS** if you plan to dynamically reconfigure the Native PHY.
Initial PCS datapath selection	**Enable Standard PCS** **Enable 10G PCS**	Specifies the active datapath when you enable both the **Standard PCS** and **10G PCS**.
Number of data channels	Device Dependent	Specifies the total number of data channels in each direction. From 1-32 channels are supported.

Name	Range	Description
Bonding mode	**Non-bonded or x1** **×6/×N** **fb_compensation**	In **Non-bonded or x1** mode, each channel is paired with a PLL. If one PLL drives multiple channels, PLL merging is required. During compilation, the Intel Quartus Prime Fitter, merges all the PLLs that meet PLL merging requirements. Refer to **Merging TX PLLs In Multiple Transceiver PHY Instances** on page 17-58 to observe PLL merging rules. When you select **×6/×N Bonding Mode**, the Intel Quartus Prime software uses a single TX PLL to generate the clock for up to 6 channels in a single transceiver bank. If the channels used cross a transceiver bank boundary, the Intel Quartus Prime software uses the **×N** clock lines to route the same clock source to the channels. Bonded channels do not support dynamic reconfiguration of the transceiver. Select **fb_compensation** (feedback compensation) to use the same clock source for multiple channels across different transceiver banks to reduce clock skew. For more information about bonding, refer to "Bonded Channel Configurations Using the PLL Feedback Compensation Path" in volume 2 of the *Stratix V Device Handbook*.
Enable simplified data interface	**On/Off**	When you turn this option **On**, the Native PHY presents only the relevant data bits. When you turn this option **Off**, the Native PHY presents the full raw interface to the fabric. If you plan to dynamically reconfigure the Native PHY, you must turn this option **Off** and you need to understand the mapping of data to the FPGA fabric. Refer to **Table 13-10** for more information. When you turn this option **On** , the Native PHY presents an interface that includes only the data necessary for the single configuration specified.

Related Information

Transceiver Clocking in Stratix V Devices

PMA Parameters for Stratix V Native PHY

This section describes the PMA parameters for the Stratix V native PHY.

Table 13-3: PMA Options

The following table describes the options available for the PMA. For more information about the PMA, refer to the *PMA Architecture* section in the *Transceiver Architecture in Stratix V Devices*.

Some parameters have ranges where the value is specified as Device Dependent. For such parameters, the possible range of frequencies and bandwidths depends on the device, speed grade, and other design characteristics. Refer to the *Stratix V Device Datasheet* for specific data for Stratix V devices.

Parameter	Range	Description
Data rate	Device Dependent	Specifies the data rate.
TX local clock division factor	**1**, **2**, **4**, **8**	Specifies the value of the divider available in the transceiver channels to divide the input clock to generate the correct frequencies for the parallel and serial clocks.
TX PLL base data rate	Device Dependent	Specifies the base data rate for the clock input to the TX PLL. Select a **base data rate** that minimizes the number of PLLs required to generate all the clocks required for data transmission. By selecting an appropriate **base data rate**, you can change data rates by changing the divider used by the clock generation block.
PLL base data rate	Device Dependent	Shows the base data rate of the clock input to the TX PLL. The **PLL base data rate** is computed from the TX local clock division factor multiplied by the **data rate**. Select a **PLL base data rate** that minimizes the number of PLLs required to generate all the clocks for data transmission. By selecting an appropriate **PLL base data rate**, you can change data rates by changing the **TX local clock division factor** used by the clock generation block.

TX PMA Parameters

Table 13-4: TX PMA Parameters

The following table describes the TX PMA options you can specify.

For more information about the TX CMU, ATX, and fractional PLLs, refer to the *Stratix V PLLs* section in *Transceiver Architecture in Stratix V Devices*.

Parameter	Range	Description
Enable TX PLL dynamic reconfiguration	**On/Off**	When you turn this option **On**, you can dynamically reconfigure the PLL to use a different reference clock input. This option is also required to simulate TX PLL reconfiguration. If you turn this option **On**, the Intel Quartus Prime Fitter prevents PLL merging by default; however, you can specify merging using the `XCVR_TX_PLL_RECONFIG_GROUP` QSF assignment.

Parameter	Range	Description
Use external TX PLL	**On/Off**	When you turn this option **On**, the Native PHY does not automatically instantiate a TX PLL. Instead, you must instantiate an external PLL and connect it to the `ext_pll_clk[<p> -1 : 0]` port of the Stratix Native PHY. Use the Stratix V Transceiver PLL IP Core to instantiate a CMU or ATX PLL. Use Altera Phase-Locked Loop (ALTERA_ PLL) Megafunction to instantiate a fractional PLL.
Number of TX PLLs	**1-4**	Specifies the number of TX PLLs that can be used to dynamically reconfigure channels to run at multiple data rates. If your design does not require transceiver TX PLL dynamic reconfiguration, set this value to 1. The number of actual physical PLLs that are implemented depends on the selected clock network. Each channel can dynamically select between n PLLs, where n is the number of PLLs specified for this parameter. **Note:** Refer to *Transceiver Clocking in Stratix V Devices* chapter for more details.
Main TX PLL logical index	**0-3**	Specifies the index of the TX PLL used in the initial configuration.
Number of TX PLL reference clocks	**1-5**	Specifies the total number of reference clocks that are shared by all of the PLLs.

Table 13-5: TX PLL Parameters

The following table describes how you can define multiple TX PLLs for your Native PHY. The Native PHY GUI provides a separate tab for each TX PLL.

Parameter	Range	Description
PLL type	**CMU** **ATX**	You can select either the **CMU** or **ATX** PLL. The **CMU** PLL has a larger frequency range than the **ATX** PLL. The **ATX** PLL is designed to improve jitter performance and achieves lower channel-to-channel skew; however, it supports a narrower range of data rates and reference clock frequencies. Another advantage of the **ATX** PLL is that it does not use a transceiver channel, while the **CMU** PLL does. Because the **CMU** PLL is more versatile, it is specified as the default setting. An error message displays in the message pane if the settings chosen for **Data rate** and **Input clock frequency** are not supported for selected PLL.

Parameter	Range	Description
PLL base data rate	Device Dependent	Shows the base data rate of the clock input to the TX PLL.The **PLL base data rate** is computed from the **TX local clock division factor** multiplied by the **Data rate**. Select a **PLL base data rate** that minimizes the number of PLLs required to generate all the clocks for data transmission. By selecting an appropriate **PLL base data rate**, you can change data rates by changing the **TX local clock division factor** used by the clock generation block.
Reference clock frequency	Device Dependent	Specifies the frequency of the reference clock for the **Selected reference clock source** index you specify. You can define a single frequency for each PLL. You can use the Transceiver Reconfiguration Controller shown in Stratix V Native Transceiver PHY IP Core to dynamically change the reference clock input to the PLL. Note that the list of frequencies updates dynamically when you change the **Data rate**. The Input clock frequency drop down menu is populated with all valid frequencies derived as a function of the data rate and base data rate. However, if **fb_compensation** is selected as the bonding mode then the input reference clock frequency is limited to the data rate divided by the PCS-PMA interface width.
Selected reference clock source	**0-4**	You can define up to 5 frequencies for the PLLs in your core. The **Reference clock frequency** selected for index 0 , is assigned to TX PLL<0>. The **Reference clock frequency** selected for index 1 , is assigned to TX PLL<1>, and so on.

RX CDR Options

Table 13-6: RX PMA Parameters

The following table describes the RX CDR options you can specify. For more information about the CDR circuitry, refer to the Receiver Clock Data Recovery Unit section in Clock Networks and PLLs in Stratix V Devices.

Parameter	Range	Description
Enable CDR dynamic reconfiguration	**On/Off**	When you turn this option **On**, you can dynamically change the reference clock input the CDR circuit. This option is also required to simulate TX PLL reconfiguration.
Number of CDR reference clocks	**1-5**	Specifies the number of reference clocks for the CDRs.

Parameter	Range	Description
Selected CDR reference clock	**0-4**	Specifies the index of the selected CDR reference clock.
Selected CDR reference clock frequency	Device Dependent	Specifies the frequency of the clock input to the CDR.
PPM detector threshold	**+/- 1000 PPM**	Specifies the maximum PPM difference the CDR can tolerate between the input reference clock and the recovered clock.
Enable rx_is_lockedtodata port	**On/Off**	When you turn this option **On**, the `rx_is_lockedto-data` port is an output of the PMA.
Enable rx_is_lockedtoref port	**On/Off**	When you turn this option **On**, the `rx_is_lockedtoref` port is an output of the PMA.
Enable rx_set_locktodata and rx_set_locktoref ports	**On/Off**	When you turn this option **On**, the `rx_set_locktodata` and `rx_set_locktoref` ports are outputs of the PMA.
Enable rx_pma_bitslip_port	**On/Off**	When you turn this option **On**, the `rx_pma_bitslip` is an input to the core. The deserializer slips one clock edge each time this signal is asserted. You can use this feature to minimize uncertainty in the serialization process as required by protocols that require a datapath with deterministic latency such as CPRI.
Enable rx_seriallpbken port	**On/Off**	When you turn this option **On**, the `rx_seriallpbken` is an input to the core. When your drive a 1 on this input port, the PMA operates in loopback mode with TX data looped back to the RX channel.

PMA Optional Ports

Table 13-7: RX PMA Parameters

The following table describes the optional ports you can include in your IP Core. The QPI interface implements the Intel Quickpath Interconnect.

For more information about the CDR circuitry, refer to the *Receiver Clock Data Recovery Unit* section in *Clock Networks and PLLs in Stratix V Devices*.

Parameter	Range	Description
Enable tx_pma_qpipullup port (QPI)	**On/Off**	When you turn this option **On**, the core includes `tx_pma_qpipullup` control input port. This port is only used for QPI applications.
Enable tx_pma_qpipulldn port (QPI)	**On/Off**	When you turn this option **On**, the core includes `tx_pma_qpipulldn` control input port. This port is only used for QPI applications.

Parameter	Range	Description
Enable tx_pma_txdetectrx port (QPI)	**On/Off**	When you turn this option **On**, the core includes `tx_pma_txdetectrx` control input port. This port is only used for QPI applications. The RX detect block in the TX PMA detects the presence of a receiver at the other end of the channel. After receiving a `tx_pma_txdetectrx` request, the receiver detect block initiates the detection process.
Enable tx_pma_rxfound port (QPI)	**On /Off**	When you turn this option **On**, the core includes `tx_pma_rxfound` output status port. This port is only used for QPI applications. The RX detect block in the TX PMA detects the presence of a receiver at the other end of the channel. `tx_pma_rxfound` indicates the result of detection.
Enable rx_pma_qpipulldn port (QPI)	**On/Off**	When you turn this option **On**, the core includes the rx_pma_qpipulldn port. This port is only used for QPI applications.
Enable rx_pma_clkout port	**On/Off**	When you turn this option **On**, the RX parallel clock which is recovered from the serial received data is an output of the PMA.
Enable rx_is_lockedtodata port	**On/Off**	When you turn this option **On**, the `rx_is_lockedtodata` port is an output of the PMA.
Enable rx_is_lockedtoref port	**On/Off**	When you turn this option **On**, the `rx_is_lockedtoref` port is an output of the PMA.
Enable rx_set_lockedtodata and **rx_set_locktoref ports**	**On/Off**	When you turn this option **On**, the `rx_set_lockedtodata` and rx_set_lockedtoref ports are outputs of the PMA.
Enable rx_clkslip port	**On/Off**	When you turn this option **On**, the `rx_clkslip` control input port is enabled. The deserializer slips one clock edge each time this signal is asserted. You can use this feature to minimize uncertainty in the serialization process as required by protocols that require a datapath with deterministic latency such as CPRI.
Enable rx_seriallpbken port	**On/Off**	When you turn this option **On**, the `rx_seriallpbken` is an input to the core. When your drive a 1 on this input port, the PMA operates in loopback mode with TX data looped back to the RX channel.

The following table lists the best case latency for the most significant bit of a word for the RX deserializer for the PMA Direct datapath. For example, for an 8-bit interface width, the latencies in UI are 11 for bit 7, 12 for bit 6, 13 for bit 5, and so on.

Table 13-8: Latency for RX Deserialization in Stratix V Devices

FPGA Fabric Interface Width	Stratix V Latency in UI
8 bits	11
10 bits	13
16 bits	19
20 bits	23
32 bits	35
40 bits	43
64 bits	99
80 bits	123

Table 13-9: Latency for TX Serialization in Stratix V Devices

The following table lists the best- case latency for the LSB of the TX serializer for all supported interface widths for the PMA Direct datapath.

FPGA Fabric Interface Width	Stratix V Latency in UI
8 bits	44
10 bits	54
16 bits	68
20 bits	84
32 bits	100
40 bits	124
64 bits	132
80 bits	164

The following tables lists the bits used for all FPGA fabric to PMA interface widths. Regardless of the FPGA Fabric Interface Width selected, all 80 bits are exposed for the TX and RX parallel data ports. However, depending upon the interface width selected not all bits on the bus will be active. The following table lists which bits are active for each FPGA Fabric Interface Width selection. For example, if your interface is 16 bits, the active bits on the bus are [17:10] and [7:0] of the 80 bit bus. The non-active bits are tied to ground.

Table 13-10: Active Bits for Each Fabric Interface Width in PMA Direct Mode

FPGA Fabric Interface Width	Bus Bits Used
8 bits	[7:0]
10 bits	[9:0]
16 bits	{[17:10], [7:0]}
20 bits	[19:0]

FPGA Fabric Interface Width	Bus Bits Used
32 bits	{[37:30], [27:20], [17:10], [7:0]}
40 bits	[39:0]
64 bits	{[77:70], [67:60], [57:50], [47:40], [37:30], [27:20], [17:10], [7:0]}
80 bits	[79:0]

Related Information

- **Transceiver Architecture in Stratix V Devices**
- **Stratix V Device Datasheet**
- **Transceiver Clocking in Stratix V Devices**

Standard PCS Parameters for the Native PHY

This section shows the complete datapath and clocking for the Standard PCS and defines the parameters available in the GUI to enable or disable the individual blocks in the Standard PCS.

Figure 13-3: The Standard PCS Datapath

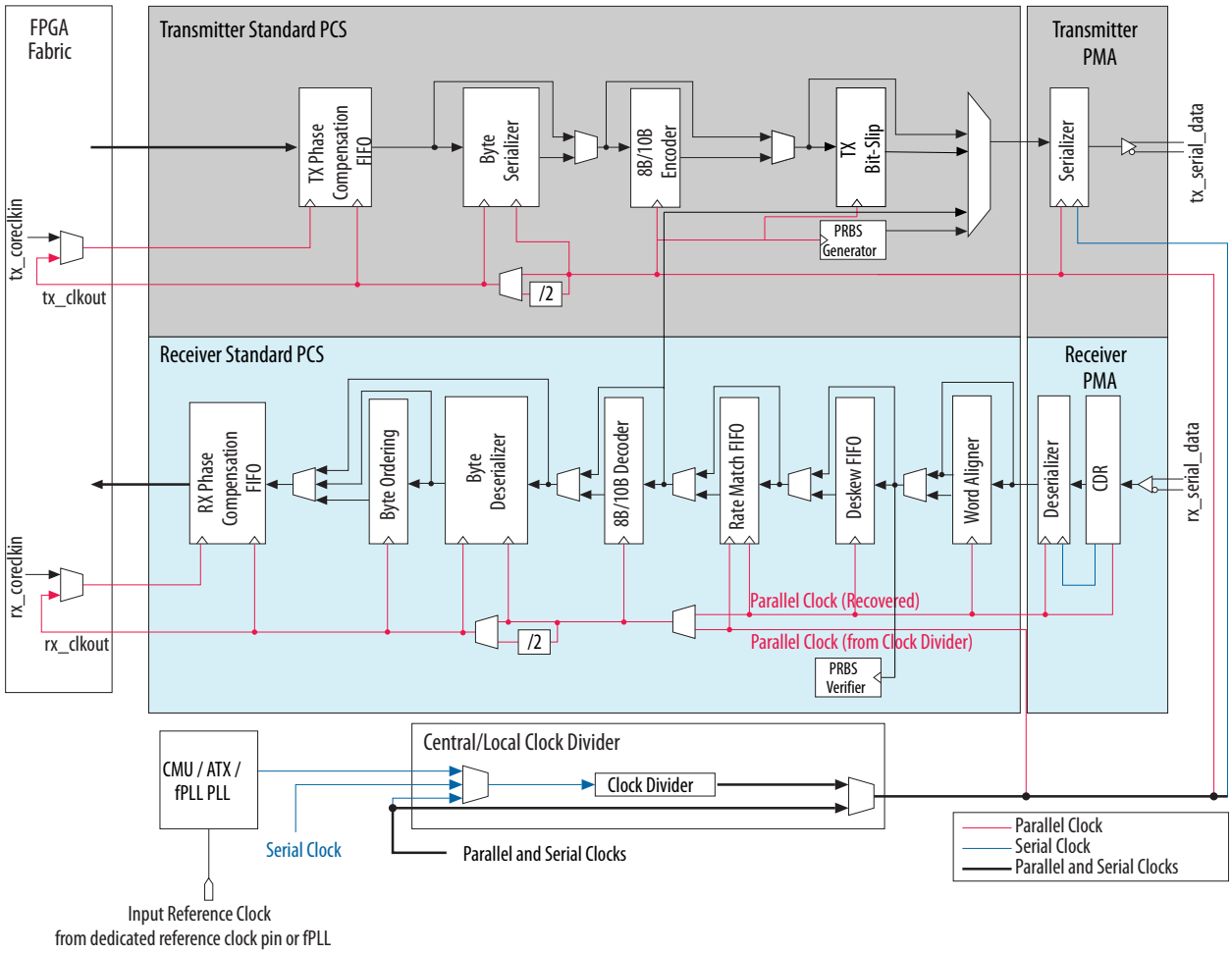


Table 13-11: General and Datapath Parameters

The following table describes the general and datapath options for the Standard PCS.

Parameter	Range	Description
Standard PCS protocol mode	**basic** **cpri** **gige** **srio_2p1**	Specifies the protocol that you intend to implement with the Native PHY. The protocol mode selected guides the MegaWizard in identifying legal settings for the Standard PCS datapath. Use the following guidelines to select a protocol mode: • **basic** -select this mode for when none of the other options are appropriate. You should also select this mode to enable diagnostics, such as loopback. • **cpri** select this mode if you intend to implement CPRI or another protocol that requires deterministic latency. Altera recommends that you select the appropriate CPRI preset for the CPRI protocol. • **gige** -select this mode if you intend to implement Gigabit Ethernet. Altera recommends that you select the appropriate GIGE preset for the Ethernet bandwidth you intend to implement. • **srio_2p1** -select this mode if you intend to implement the Serial RapidIO protocol.
Standard PCS/PMA interface width	**8, 10, 16, 20, 32, 40 64, 80**	Specifies the width of the datapath that connects the FPGA fabric to the PMA. The transceiver interface width depends upon whether you enable 8B/10B. To simplify connectivity between the FPGA fabric and PMA, the bus bits used are not contiguous for 16- and 32-bit buses. 16-, 32-, and 64-bit buses. Refer to **Table 13-10** for the bits used.
FPGA fabric/Standard TX PCS interface width	**8, 10, 16, 20**	Shows the FPGA fabric to TX PCS interface width which is calculated from the Standard PCS/PMA interface width.
FPGA fabric/Standard RX PCS interface width	**8, 10, 16, 20**	Shows the FPGA fabric to RX PCS interface width which is calculated from the Standard PCS/PMA interface width.
Enable Standard PCS low latency mode	**On/ Off**	When you turn this option **On**, all PCS functions are disabled. This option creates a the lowest latency Native PHY that allows dynamic reconfigure between multiple PCS datapaths.

Phase Compensation FIFO

The phase compensation FIFO assures clean data transfer to and from the FPGA fabric by compensating for the clock phase difference between the low-speed parallel clock and FPGA fabric interface clock. The following table describes the options for the phase compensation FIFO.

Table 13-12: Phase Compensation FIFO Parameters

Parameter	Range	Description
TX FIFO mode	**low_latency** **register_fifo**	The following 2 modes are possible: • **low_latency** : This mode adds 3-4 cycles of latency to the TX datapath. • **register_fifo** : In this mode the FIFO is replaced by registers to reduce the latency through the PCS. Use this mode for protocols that require deterministic latency, such as CPRI.
RX FIFO mode	**low_latency** **register_fifo**	The following 2 modes are possible: • **low_latency** : This mode adds 2-3 cycles of latency to the RX datapath. • **register_fifo** : In this mode the FIFO is replaced by registers to reduce the latency through the PCS. Use this mode for protocols that require deterministic latency, such as CPRI.
Enable tx_std_pcfifo_full port	**On/Off**	When you turn this option **On**, the TX Phase compensation FIFO outputs a FIFO full status flag.
Enable tx_std_pcfifo_empty port	**On/Off**	When you turn this option **On**, the TX Phase compensation FIFO outputs a FIFO empty status flag.
Enable rx_std_pcfifo_full port	**On/Off**	When you turn this option **On**, the RX Phase compensation FIFO outputs a FIFO full status flag.
Enable rx_std_pcfifo_empty port	**On/ Off**	When you turn this option **On**, the RX Phase compensation FIFO outputs a FIFO empty status flag.

Byte Ordering Block Parameters

The RX byte ordering block realigns the data coming from the byte deserializer. This block is necessary when the PCS to FPGA fabric interface width is greater than the PCS datapath. Because the timing of the RX PCS reset logic is indeterminate, the byte ordering at the output of the byte deserializer may or may not match the original byte ordering of the transmitted data. The following table describes the byte ordering block parameters.

Parameter	Range	Description
Enable RX byte ordering	**On/Off**	When you turn this option **On**, the PCS includes the byte ordering block.

Parameter	Range	Description
Byte ordering control mode	**Manual** **Auto**	Specifies the control mode for the byte ordering block. The following modes are available: • **Manual** : Allows you to control the byte ordering block • **Auto** : The word aligner automatically controls the byte ordering block once word alignment is achieved.
Byte ordering pattern width	**8-10**	Shows width of the pad that you must specify. This width depends upon the PCS width and whether nor not 8B/10B encoding is used as follows: **Width 8B/10B Pad Pattern** 8/16,32 No 8 bits 10,20,40 No 10 bits 8,16,32 Yes 9 bits
Byte ordering symbol count	**1-2**	Specifies the number of symbols the word aligner should search for. When the PMA is 16 or 20 bits wide, the byte ordering block can optionally search for 1 or 2 symbols.
Byte order pattern (hex)	User-specified 8-10 bit pattern	Specifies the search pattern for the byte ordering block.
Byte order pad value (hex)	User-specified 8-10 bit pattern	Specifies the pad pattern that is inserted by the byte ordering block. This value is inserted when the byte order pattern is recognized. The byte ordering pattern should occupy the least significant byte (LSB) of the parallel TX data. If the byte ordering block identifies the programmed byte ordering pattern in the most significant byte (MSB) of the byte-deserialized data, it inserts the appropriate number of user-specified pad bytes to push the byte ordering pattern to the LSB position, restoring proper byte ordering.
Enable rx_std_byteorder_ena port	**On/Off**	Enables the optional `rx_std_byte_order_ena` control input port. When this signal is asserted, the byte ordering block initiates a byte ordering operation if the **Byte ordering control mode** is set to **manual**. Once byte ordering has occurred, you must deassert and reassert this signal to perform another byte ordering operation. This signal is an synchronous input signal; however, it must be asserted for at least 1 cycle of `rx_std_clkout`.

Parameter	Range	Description
Enable rx_std_byteorder_flag port	**On/Off**	Enables the optional `rx_std_byteorder_flag` status output port. When asserted, indicates that the byte ordering block has performed a byte order operation. This signal is asserted on the clock cycle in which byte ordering occurred. This signal is synchronous to the `rx_std_clkout clock`.

Byte Serializer and Deserializer

The byte serializer and deserializer allow the PCS to operate at twice the data width of the PMA serializer. This feature allows the PCS to run at a lower frequency and accommodate a wider range of FPGA interface widths. The following table describes the byte serialization and deserialization options you can specify.

Table 13-13: Byte Serializer and Deserializer Parameters

Parameter	Range	Description
Enable TX byte serializer	**On/Off**	When you turn this option **On**, the PCS includes a TX byte serializer which allows the PCS to run at a lower clock frequency to accommodate a wider range of FPGA interface widths.
Enable RX byte deserializer	**On/Off**	When you turn this option **On**, the PCS includes an RX byte deserializer and deserializer which allows the PCS to run at a lower clock frequency to accommodate a wider range of FPGA interface widths.

8B/10B

The 8B/10B encoder generates 10-bit code groups from the 8-bit data and 1-bit control identifier. In 8-bit 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. The 8B/10B decoder decodes the data into an 8-bit data and 1-bit control identifier. The following table describes the 8B/10B encoder and decoder options.

Table 13-14: 8B/10B Encoder and Decoder Parameters

Parameter	Range	Description
Enable TX 8B/10B encoder	**On/Off**	When you turn this option **On**, the PCS includes the 8B/10B encoder.
Enable TX 8B/10B disparity control	**On/Off**	When you turn this option **On**, the PCS includes disparity control for the 8B/10B encoder. Your force the disparity of the 8B/10B encoder using the `tx_forcedisp` control signal.

Parameter	Range	Description
Enable RX 8B/10B decoder	**On/Off**	When you turn this option **On**, the PCS includes the 8B/10B decoder.

Rate Match FIFO

The rate match FIFO compensates for the very small frequency differences between the local system clock and the RX recovered clock. The following table describes the rate match FIFO parameters.

Table 13-15: Rate Match FIFO Parameters

Parameter	Range	Description
Enable RX rate match FIFO	**On/Off**	When you turn this option On , the PCS includes a FIFO to compensate for the very small frequency differences between the local system clock and the RX recovered clock.
RX rate match insert/delete +ve pattern (hex)	User-specified 20 bit pattern	Specifies the +ve (positive) disparity value for the RX rate match FIFO as a hexadecimal string.
RX rate match insert/delete -ve pattern (hex)	User-specified 20 bit pattern	Specifies the -ve (negative) disparity value for the RX rate match FIFO as a hexadecimal string.
Enable rx_std_rm_fifo_empty port	**On/Off**	When you turn this option **On**, the rate match FIFO outputs a FIFO empty status flag. The rate match FIFO compensates for small clock frequency differences between the upstream transmitter and the local receiver clocks by inserting or removing skip (SKP) symbols or ordered sets from the inter-packet gap (IPG) or idle stream. This port is only used for XAUI, GigE, and Serial RapidIO in double width mode. In double width mode, the FPGA data width is twice the PCS data width to allow the fabric to run at half the PCS frequency.
Enable rx_std_rm_fifo_full port	**On/Off**	When you turn this option **On**, the rate match FIFO outputs a FIFO full status flag. This port is only used for XAUI, GigE, and Serial RapidIO in double width mode.

When you enable the simplified data interface and enable the rate match FIFO status ports, the rate match FIFO bits map to the high-order bits of the data bus as listed in the following table. This table uses the following definitions:

- Basic double width: The **Standard PCS protocol mode** GUI option is set to **basic**. The FPGA data width is twice the PCS data width to allow the fabric to run at half the PCS frequency.
- Serial^{TM} RapidIO double width: You are implementing the Serial RapidIO protocol. The FPGA data width is twice the PCS data width to allow the fabric to run at half the PCS frequency.

Note: If you have the auto-negotiation state machine in your transceiver design, please note that the rate match FIFO is capable of inserting or deleting the first two bytes (K28.5//D2.2) of /C2/ ordered sets during auto-negotiation. However, the insertion or deletion of the first two bytes of /C2/ ordered sets can cause the auto-negotiation link to fail. For more information, visit **Altera Knowledge Base Support Solution**.

Table 13-16: Status Flag Mappings for Simplified Native PHY Interface

Status Condition	Protocol	Mapping of Status Flags to RX Data	Value
Full	PHY IP Core for PCI Express (PIPE) Basic double width	`RXD[62:62] = rx_rmfifostatus[1:0]`, or `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[30:29] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b11 = full`
	XAUI, GigE, Serial RapidIO double width	`rx_std_rm_fifo_full`	`1'b1 = full`
	All other protocols	Depending on the FPGA fabric to PCS interface width either: `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b11 = full`

Status Condition	Protocol	Mapping of Status Flags to RX Data	Value
Empty	PHY IP Core for PCI Express (PIPE) Basic double width	`RXD[62:62] = rx_rmfifostatus[1:0]`, or `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[30:29] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`(2'b10) AND (PAD or EDB)` PAD = K23.7 or 9'h1F7 EDB = K30.7 or 9'h1FE
	XAUI, GigE, Serial RapidIO double width	`rx_std_rm_fifo_empty`	`1'b1 = empty`
	All other protocols	Depending on the FPGA fabric to PCS interface width either: `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`(2'b10) AND (PAD or EDB)` (12) PAD = K23.7 or 9'h1F7 EDB = K30.7 or 9'h1FE
Insertion	Basic double width Serial RapidIO double width	`RXD[62:62] = rx_rmfifostatus[1:0]`, or `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[30:29] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b10`
	All other protocols	Depending on the FPGA fabric to PCS interface width either: `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b10`

(12) PAD and EBD are control characters. PAD character is typically used fo fill in the remaining lanes in a multi-lane link when one of the link goes to logical idle state. EDB indicates End Bad Packet.

Status Condition	Protocol	Mapping of Status Flags to RX Data	Value
Deletion	Basic double width Serial RapidIO double width	`RXD[62:62] = rx_rmfifostatus[1:0]`, or `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[30:29] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b01`
	All other protocols	Depending on the FPGA fabric to PCS interface width either: `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b01`

Word Aligner and Bit-Slip Parameters

The word aligner aligns the data coming from RX PMA deserializer to a given word boundary. When the word aligner operates in bit-slip mode, the word aligner slips a single bit for every rising edge of the bit slip control signal. The following table describes the word aligner and bit-slip parameters.

Table 13-17: Word Aligner and Bit-Slip Parameters

Parameter	Range	Description
Enable TX bit-slip	**On/Off**	When you turn this option **On**, the PCS includes the bit-slip function. The outgoing TX data can be slipped by the number of bits specified by the `tx_bitslipboundarysel` control signal.
Enable tx_std_bitslipboundarysel control input port	**On/Off**	When you turn this option On , the PCS includes the optional `tx_std_bitslipboundarysel` control input port.

Parameter	Range	Description
RX word aligner mode	**bit_slip** **sync_sm** **manual**	Specifies one of the following 3 modes for the word aligner: • **Bit_slip** : You can use bit slip mode to shift the word boundary. For every rising edge of the `rx_bitslip signal`, the word boundary is shifted by 1 bit. Each bit-slip removes the earliest received bit from the received data • **Sync_sm** : In synchronous state machine mode, a programmable state machine controls word alignment. You can only use this mode with 8B/10B encoding. The data width at the word aligner can be 10 or 20 bits. When you select this word aligner mode, the synchronous state machine has hysteresis that is compatible with XAUI. However, when you select **cpri** for the **Standard PCS Protocol Mode**, this option selects the deterministic latency word aligner mode. • **Manual** : This mode Enables word alignment by asserting the `rx_std_wa_patternalign`. This is an edge sensitive signal.
RX word aligner pattern length	**7**, **8**, **10** **16**, **20**, **32**	Specifies the length of the pattern the word aligner uses for alignment.
RX word aligner pattern (hex)	User-specified	Specifies the word aligner pattern in hex.
Number of word alignment patterns to achieve sync	`1-256`	Specifies the number of valid word alignment patterns that must be received before the word aligner achieves synchronization lock. The default is 3.
Number of invalid words to lose sync	`1-256`	Specifies the number of invalid data codes or disparity errors that must be received before the word aligner loses synchronization. The default is 3.
Number of valid data words to decrement error count	`1-256`	Specifies the number of valid data codes that must be received to decrement the error counter. If the word aligner receives enough valid data codes to decrement the error count to 0, the word aligner returns to synchronization lock.
Run length detector word count	`0-63`	Specifies the maximum number of contiguous 0s or 1s in the data stream before the word aligner reports a run length violation.

Parameter	Range	Description
Enable rx_std_wa_patternalign port	**On/Off**	Enables the optional `rx_std_wa_patternalign` control input port. A rising edge on this signal causes the word aligner to align the next incoming word alignment pattern when the word aligner is configured in **manual** mode.
Enable rx_std_wa_a1a2size port	**On/Off**	Enables the optional `rx_std_wa_a1a2size` control input port.
Enable rx_std_wa_bitslipboundarysel port	**On/Off**	Enables the optional `rx_std_wa_bitslipboundarysel` status output port.
Enable rx_std_wa_bitslip port	**On/Off**	Enables the optional `rx_std_wa_bitslip` control input port.
Enable rx_std_wa_runlength_err port	**On/Off**	Enables the optional `rx_std_wa_runlength_err` control input port.

Bit Reversal and Polarity Inversion

These functions allow you to reverse bit order, byte order, and polarity to correct errors and to accommodate different layouts of data. The following table describes these parameters.

Parameter	Range	Description
Enable TX bit reversal	**On/Off**	When you turn this option **On**, the word aligner reverses TX parallel data before transmitting it to the PMA for serialization. You can only change this static setting using the Transceiver Reconfiguration Controller.
Enable RX bit reversal	**On/Off**	When you turn this option **On**, the `rx_st_bitrev_ena` port controls bit reversal of the RX parallel data after it passes from the PMA to the PCS.
Enable RX byte reversal	**On/Off**	When you turn this option **On**, the word aligner reverses the byte order before transmitting data. This function allows you to reverse the order of bytes that were erroneously swapped. The PCS can swap the ordering of both 8 and10 bit words.
Enable TX polarity inversion	**On/Off**	When you turn this option **On**, the `tx_std_polinv` port controls polarity inversion of TX parallel data before transmitting the parallel data to the PMA.
Enable RX polarity inversion	**On/Off**	When you turn this option **On**, asserting `rx_std_polinv` controls polarity inversion of RX parallel data after PMA transmission.

Parameter	Range	Description
Enable rx_std_bitrev_ena port	**On/Off**	When you turn this option **On**, asserting `rx_std_bitrev_ena` control port causes the RX data order to be reversed from the normal order, LSB to MSB, to the opposite, MSB to LSB. This signal is an asynchronous input.
Enable rx_std_byterev_ena port	**On/Off**	When you turn this option **On**, asserting `rx_std_byterev_ena` input control port causes swaps the order of the individual 8- or 10-bit words received from the PMA.
Enable tx_std_polinv port	**On/Off**	When you turn this option **On**, the `tx_std_polinv` input is enabled. You can use this control port to swap the positive and negative signals of a serial differential link if they were erroneously swapped during board layout.
Enable rx_std_polinv port	**On/Off**	When you turn this option **On**, the `rx_std_polinv` input is enabled. You can use this control port to swap the positive and negative signals of a serial differential link if they were erroneously swapped during board layout.
Enable tx_std_elecidle port	**On/Off**	When you turn this option **On**, the `tx_std_elecidle` input port is enabled. When this signal is asserted, it forces the transmitter to electrical idle. This signal is required for the PCI Express protocol.
Enable rx_std_signaldetect port	**On/Off**	When you turn this option **On**, the optional `tx_std_signaldetect` output port is enabled. This signal is required for the PCI Express protocol. If enabled, the signal threshold detection circuitry senses whether the signal level present at the RX input buffer is above the signal detect threshold voltage that you specified. For SATA / SAS applications, enable this port and set the following QSF assignments to the transceiver receiver pin: • `set_instance_assignment -name XCVR_RX_SD_ENABLE ON` • `set_instance_assignment -name XCVR_RX_SD_THRESHOLD 7` • `set_instance_assignment -name XCVR_RX_COMMON_MODE_VOLTAGE VTT_OP55V` • `set_instance_assignment -name XCVR_RX_SD_OFF 1` • `set_instance_assignment -name XCVR_RX_SD_ON 2`

PRBS Verifier

You can use the PRBS pattern generators for verification or diagnostics. The pattern generator blocks support the following patterns:

- Pseudo-random binary sequence (PRBS)
- Square wave

Table 13-18: PRBS Parameters

Parameter	Range	Description
Enable rx_std_prbs ports	**On/Off**	When you turn this option **On**, the PCS includes the `rx_std_prbs_done` and `rx_std_prbs_err` signals to provide status on PRBS operation.

Related Information

Transceiver Architecture in Stratix V Devices

Standard PCS Pattern Generators

The Standard PCS includes a pattern generator that generates and verifies the PRBS patterns.

Table 13-19: Standard PCS PRBS Patterns

PATTERN	POLYNOMIAL
PRBS-7	$X^7 + X^6 + 1$
PRBS-8	$X^8 + X^7 + X^3 + X^2 + 1$
PRBS-10	$X^{10} + X^7 + 1$
PRBS-15	$X^{15} + X^{14} + 1$
PRBS-23	$X^{23} + X^{18} + 1$
PRBS-31	$X^{31} + X^{28} + 1$

The Standard PCS requires a specific word alignment for the PRBS pattern. You must specify a word alignment pattern in the verifier that matches the generator pattern specified. In the Standard PCS, PRBS patterns available depend upon the PCS-PMA width. The following table below illustrates the patterns are available based upon the PCS-PMA width.

Table 13-20: PRBS Patterns in the 8G PCS with PCS-PMA Widths

	PCS-PMA Width			
	8-Bit	10-Bit	16-Bit	20-Bit
PRBS-7	X		X	X
PRBS-8	X			
PRBS-10		X		
PRBS 15	X	X	X	X
PRBS 23	X		X	X
PRBS 31	X	X	X	X

Unlike the 10G PRBS verifier, the Standard PRBS verifier uses the Standard PCS word aligner. You must specify the word aligner size and pattern. The following table lists the encodings for the available choices.

Table 13-21: Word Aligner Size and Word Aligner Pattern

PCS-PMA Width	PRBS Patterns	PRBS Pattern Select	Word Aligner Size	Word Aligner Pattern
8-bit	PRBS 7	3'b010	3'b001	0x0000003040
	PRBS 8	3'b000	3'b001	0x000000FF5A
	PRBS 23	3'b100	3'b001	0x0000003040
	PRBS 15	3'b101	3'b001	0x0000007FFF
	PRBS 31	3'b110	3'b001	0x000000FFFF
10-bit	PRBS 10	3'b000	3'b010	0x00000003FF
	PRBS 15	3'b101	3'b000	0x0000000000
	PRBS 31	3'b110	3'b010	0x00000003FF
16-bit	PRBS 7	3'b000	3'b010	0x0000003040
	PRBS 23	3'b001	3'b101	0x00007FFFFF
	PRBS 15	3'b101	3'b011	0x0000007FFF
	PRBS 31	3'b110	3'b011	0x000000FFFF

PCS-PMA Width	PRBS Patterns	PRBS Pattern Select	Word Aligner Size	Word Aligner Pattern
20-bit	PRBS 7	3'b000	3'b100	0x0000043040
	PRBS 23	3'b001	3'b110	0x00007FFFFF
	PRBS 15	3'b101	3'b100	0x0000007FFF
	PRBS 31	3'b110	3'b110	0x007FFFFFFF

Registers and Values

The following table lists the offsets and registers for the Standard PCS pattern generator and verifier.

Note: All undefined register bits are reserved.

Table 13-22: Offsets for the Standard PCS Pattern Generator and Verifier

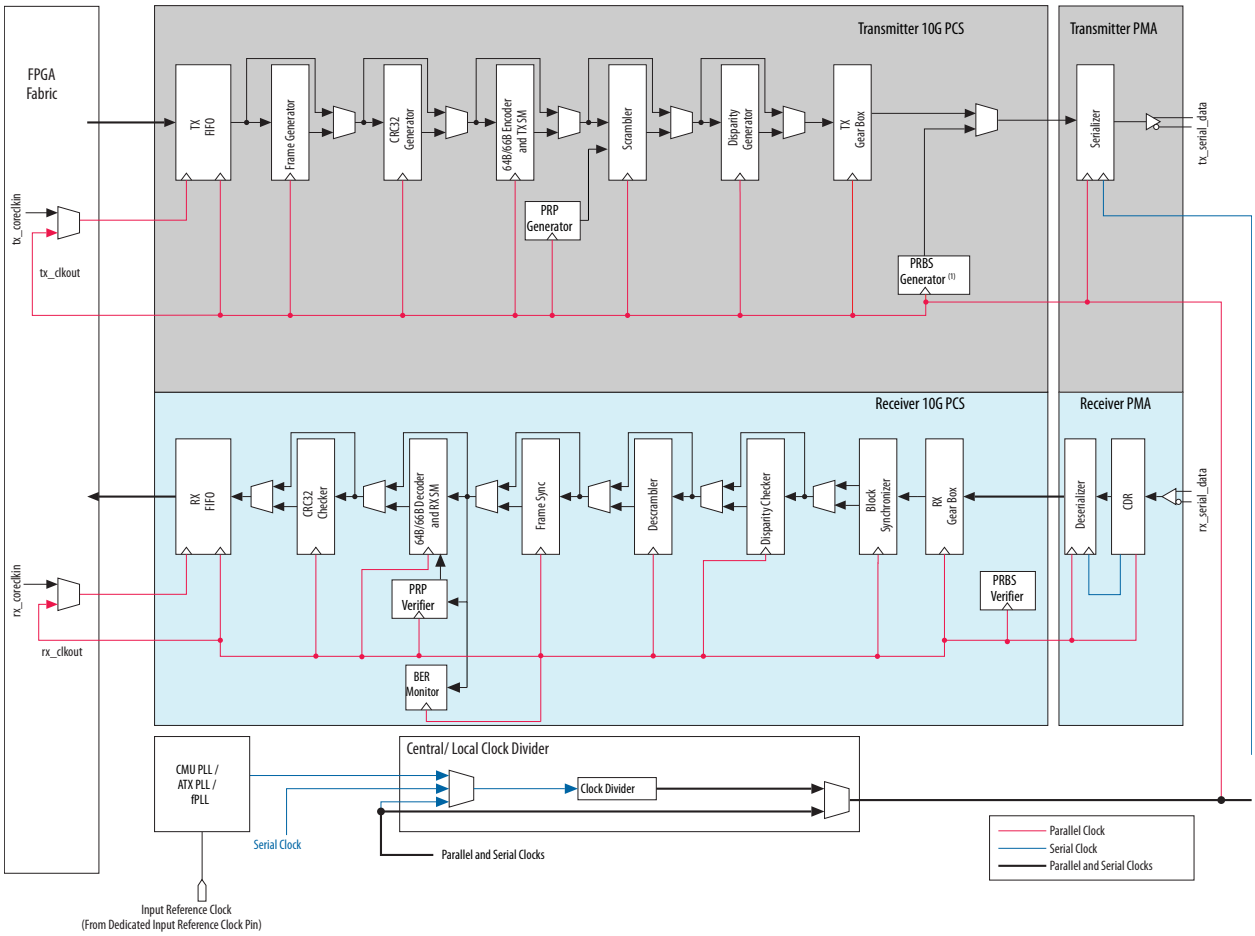
Offset	OffsetBits	R/W	Name	Description
0x97	[9]	R/W	`PRBS TX Enable`	When set to 1'b1, enables the PRBS generator.
	[8:6]	R/W	`PRBS Pattern Select`	Specifies the encoded PRBS pattern defined in the previous table.
0x99	[9]	R/W	`Clock Power Down TX`	When set to 1'b1, powers down the PRBS Clock in the transmitter. When set to 1'b0, enables the PRBS generator.
0x141	[0]	R/W	`PRBS TX Inversion`	Set to 1'b1 to invert the data leaving the PCS block.
0x16D	[2]	R/W	`PRBS RX Inversion`	Set to 1'b1 to invert the data entering the PCS block.
0xA0	[5]	R/W	`PRBS RX Enable`	When set to 1'b1, enables the PRBS verifier in the receiver.
	[4]	R/W	`PRBS Error Clear`	When set to 1'b1, deasserts `rx_prbs_` done and restarts the PRBS pattern.

Offset	OffsetBits	R/W	Name	Description
0xA1	[15:14]	R/W	Sync badcg	Must be set to 2'b00 to enable the PRBS verifier.
	[13]	R/W	Enable Comma Detect	Must be set to 1'b0 to enable the PRBS verifier.
	[11]	R/W	Enable Polarity	Must be set to 1'b0 to enable the PRBS verifier.
	[10:8]	R/W	Word Aligner Size	Specifies the word alignment size using the encodings defined in the previous table.
	[7:0]	R/W	Word Aligner Pattern [39:32]	Stores the high-order 8 bits of the word aligner pattern as specified in the previous table.
0xA2	[15:0]	R/W	Word Aligner Pattern [31:16]	Stores the middle 16 bits of the word aligner pattern as specified in the previous table.
0xA3	[15:0]	R/W	Word Aligner Pattern [15:0]	Stores the least significant 16 bits from the word aligner pattern as specified in the previous table.
0xA4	[15]	R/W	Sync State Machine Disable	Disables the synchronization state machine. When the PCS-PMA Width is 8 or 10, the value must be 1. When the PCS-PMA Width is 16 or 20, the value must be 0.
0xA6	[5]	R/W	Auto Byte Align Disable	Auto aligns the bytes. Must be set to 1'b0 to enable the PRBS verifier.
0xB8	[13]	R/W	DW Sync State Machine Enable	Enables the double width state machine. Must be set to 1'b0 to enable the PRBS verifier.
0xB9	[11]	R/W	Deterministic Latency State Machine Enable	Enables a deterministic latency state machine. Must be set to 1'b0 to enable the PRBS verifier.
0xBA	[11]	R/W	Clock Power Down RX	When set to 1'b0, powers down the PRBS clock in the receiver.

10G PCS Parameters for Stratix V Native PHY

This section shows the complete datapath and clocking for the 10G PCS and defines parameters available in the GUI to enable or disable the individual blocks in the 10G PCS.

Figure 13-4: The 10G PCS datapath



Note:
1. The PRBS pattern generator can dynamically invert the data pattern that leaves the PCS block.

Table 13-23: General and Datapath Parameters

Parameter	Range	Description
10G PCS protocol mode	**basic** **interlaken** **sfis** **teng_baser** **teng_1588** **teng_sdi**	Specifies the protocol that you intend to implement with the Native PHY. The protocol mode selected guides the MegaWizard in identifying legal settings for the 10G PCS datapath. Use the following guidelines to select a protocol mode: • **basic** : Select this mode for when none of the other options are appropriate. You should also select this mode to enable diagnostics, such as loopback. • **interlaken**: Select this mode if you intend to implement Interlaken. • **sfis** : Select this mode if you intend to implement the SERDES Framer Interface Level 5 protocol. • **teng_baser** : select this mode if you intend to implement the 10GBASE-R protocol. • **teng_1588**: select this mode if you intend to implement the 10GBASE-R protocol with the 1588 precision time stamping feature. • **teng_sdi** : 10G SDI
10G PCS/PMA interface width	**32, 40, 64**	Specifies the width of the datapath that connects the FPGA fabric to the PMA.
FPGA fabric/10G PCS interface width	**32, 40, 50** **64, 66, 67**	Specifies the FPGA fabric to TX PCS interface width . The 66-bit FPGA fabric/PCS interface width is achieved using 64-bits from the TX and RX parallel data and the lower 2-bits from the control bus. The 67-bit FPGA fabric/PCS interface width is achieved using the 64-bits from the TX and RX parallel data and the lower 3-bits from the control bus.

10G TX FIFO

The TX FIFO is the interface between TX data from the FPGA fabric and the PCS. This FIFO is an asynchronous 73-bit wide, 32-deep memory buffer It also provides full, empty, partially full, and empty flags based on programmable thresholds. The following table describes the 10G TX FIFO parameters.

Table 13-24: 10G TX FIFO Parameters

Parameter	Range	Description
TX FIFO Mode	**Interlaken** **phase_comp** **register**	Specifies one of the following 3 modes: • **interlaken** : The TX FIFO acts as an elastic buffer. The FIFO write clock frequency (`coreclk`) can exceed that of the effective read clock, `tx_clkout`. You can control writes to the FIFO with `tx_data_valid`. By monitoring the FIFO flags, you can avoid the FIFO full and empty conditions. The Interlaken frame generator controls reads. • **phase_comp** : The TX FIFO compensates for the clock phase difference between the `coreclkin` and `tx_clkout` which is an internal PCS clock. • **register** : The TX FIFO is bypassed. `tx_data` and `tx_data_valid` are registered at the FIFO output. You must control `tx_data_valid` precisely based on gearbox ratio to avoid gearbox underflow or overflow conditions.
TX FIFO full threshold	`0-31`	Specifies the full threshold for the 10G PCS TX FIFO. The active high TX FIFO full flag is synchronous to `coreclk`. The default value is 31.
TX FIFO empty threshold	`0-31`	Specifies the empty threshold for the 10G PCS TX FIFO. The active high TX FIFO empty flag is synchronous to `coreclk`. The default value is 0.
TX FIFO partially full threshold	`0-31`	Specifies the partially full threshold for the 10G PCS TX FIFO. The active high TX FIFO partially full flag is synchronous to `coreclk`. The default value is 23.
TX FIFO partially empty threshold	`0-31`	Specifies the partially empty threshold for the 10G PCS TX FIFO. The active high TX FIFO partially empty flag is synchronous to `coreclk`.
Enable tx_10g_fifo_full port	**On/Off**	When you turn this option On , the 10G PCS includes the active high `tx_10g_fifo_full` port. `tx_10g_fifo_full` is synchronous to `coreclk`.

Parameter	Range	Description
Enable tx_10g_fifo_pfull port	**On/Off**	When you turn this option On , the 10G PCS includes the active high `tx_10g_fifo_pfull port`. `tx_10g_fifo_pfull` is synchronous to `coreclk`.
Enable tx_10g_fifo_empty port	**On/Off**	When you turn this option **On**, the 10G PCS includes the active high `tx_10g_fifo_empty` port. `tx_10g_fifo_empty` is pulse-stretched. It is asynchronous to `coreclk` and synchronous to `tx_clkout` which is the read clock.
Enable tx_10g_fifo_pempty port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `tx_10g_fifo_pempty` port.
Enable tx_10g_fifo_del port (10GBASE-R)	**On/Off**	When you turn this option **On**, the 10G PCS includes the active high `tx_10g_fifo_del` port. This signal is asserted when a word is deleted from the TX FIFO. This signal is only used for the 10GBASE-R protocol.
Enable tx_10g_fifo_insert port (10GBASE-R)	**On/Off**	When you turn this option **On**, the 10G PCS includes the active high `tx_10g_fifo_insert` port. This signal is asserted when a word is inserted into the TX FIFO. This signal is only used for the 10GBASE-R protocol.

10G RX FIFO

The RX FIFO is the interface between RX data from the FPGA fabric and the PCS. This FIFO is an asynchronous 73-bit wide, 32-deep memory buffer It also provides full, empty, partially full, and empty flags based on programmable thresholds. The following table describes the 10G RX FIFO parameters.

Table 13-25: 10G RX FIFO Parameters

Parameter	Range	Description
RX FIFO Mode	**Interlaken** **clk_comp** **phase_comp** **register**	Specifies one of the following 3 modes: • **interlaken** : Select this mode for the Interlaken protocol. To implement the deskew process. In this mode the FIFO acts as an elastic buffer. The FIFO write clock can exceed the read clock. Your implementation must control the FIFO write (`tx_datavalid`) by monitoring the FIFO flags. The read enable is controlled by the Interlaken Frame Generator. • **clk_comp** : This mode compensates for the clock difference between the PLD clock (`coreclkin`) and `rxclkout`. After block lock is achieved, idle ordered set insertions and deletions compensate for the clock difference between RX PMA clock and PLD clock up to ± 100 ppm. Use this mode for 10GBASE-R. • **phase_comp** : This mode compensates for the clock phase difference between the PLD clock (`coreclkin`) and `rxclkout`. • **register** : The TX FIFO is bypassed. `rx_data` and `rx_data_valid` are registered at the FIFO output.
RX FIFO full threshold	`0-31`	Specifies the full threshold for the 10G PCS RX FIFO. The default value is 31.
RX FIFO empty threshold	`0-31`	Specifies the empty threshold for the 10G PCS RX FIFO. The default value is 0.
RX FIFO partially full threshold	`0-31`	Specifies the partially full threshold for the 10G PCS RX FIFO. The default value is 23.
RX FIFO partially empty threshold	`0-31`	Specifies the partially empty threshold for the 10G PCS RX FIFO.
Enable RX FIFO alignment word deletion (Interlaken)	**On/Off**	When you turn this option **On**, all alignment words (sync words), including the first sync word, are removed after frame synchronization is achieved. If you enable this option, you must also enable control word deletion.

Parameter	Range	Description
Enable RX FIFO control word deletion (Interlaken)	**On/Off**	When you turn this option On , the `rx_control_del` parameter enables or disables writing the Interlaken control word to RX FIFO. When disabled, a value of 0 for `rx_control_del` writes all control words to RX FIFO. When enabled, a value of 1 deletes all control words and only writes the data to the RX FIFO.
Enable rx_10g_fifo_data_valid port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_data_valid` signal which Indicates when `rx_data` is valid. This option is available when you select the following parameters: • 10G PCS protocol mode is Interlaken • 10G PCS protocol mode is Basic and RX FIFO mode is phase_comp • 10G PCS protocol mode is Basic and RX FIFO mode is register
Enable rx_10g_fifo_full port	**On/Off**	When you turn this option **On**, the 10G PCS includes the active high `rx_10g_fifo_full` port. `rx_10g_fifo_full` is synchronous to `rx_clkout`.
Enable rx_10g_fifo_pfull port	**On/Off**	When you turn this option **On**, the 10G PCS includes the active high `rx_10g_fifo_pfull` port. `rx_10g_fifo_pfull` is synchronous to `rx_clkout`.
Enable rx_10g_fifo_empty port	**On/Off**	When you turn this option **On**, the 10G PCS includes the active high `rx_10g_fifo_empty` port.
Enable rx_10g_fifo_pempty port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_fifo_pempty` port.
Enable rx_10g_fifo_del port (10GBASE-R)	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_fifo_del` port. This signal is asserted when a word is deleted from the RX FIFO. This signal is only used for the 10GBASE-R protocol.
Enable rx_10g_fifo_insert port (10GBASE-R)	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_fifo_insert` port. This signal is asserted when a word is inserted into the RX FIFO. This signal is only used for the 10GBASE-R protocol.

Parameter	Range	Description
Enable rx_10g_fifo_rd_en port (Interlaken)	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_fifo_rd_en` input port. Asserting this signal reads a word from the RX FIFO. This signal is only available for the Interlaken protocol.
Enable rx_10g_fifo_align_val port (Interlaken)	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_fifo_align_val` output port. This signal is asserted when the word alignment pattern is found. This signal is only available for the Interlaken protocol.
enable rx10g_clk33out port	**On/Off**	When you turn this option **On**, the 10G PCS includes a divide by 33 clock output port. You typically need this option when the fabric to PCS interface width is 66.
Enable rx_10g_fifo_align_clr port (Interlaken)	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_fifo_align_clr` input port. When this signal is asserted, the FIFO resets and begins searching for a new alignment pattern. This signal is only available for the Interlaken protocol.
Enable rx_10g_fifo_align_en port (Interlaken)	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_fifo_align_en` input port. This signal is used for FIFO deskew for Interlaken. When asserted, the corresponding channel is enabled for alignment. This signal is only available for the Interlaken protocol.

Interlaken Frame Generator

TX Frame generator generates the metaframe. It encapsulates the payload from MAC with the framing layer control words, including sync, scrambler, skip and diagnostic words. The following table describes the Interlaken frame generator parameters.

Table 13-26: Interlaken Frame Generator Parameters

Parameter	Range	Description
teng_tx_framgen_enable	**On/Off**	When you turn this option **On**, the frame generator block of the 10G PCS is enabled.
teng_tx_framgen_user_length	0-8192	Specifies the metaframe length.

Parameter	Range	Description
teng_tx_framgen_burst_enable	**On/Off**	When you turn this option **On**, the frame generator burst functionality is enabled.
Enable tx_10g_frame port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `tx_10g_frame` output port. When asserted, `tx_10g_frame` indicates the beginning of a new metaframe inside the frame generator.
Enable tx_10g_frame_diag_status port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `tx_10g_frame_diag_status` 2-bit input port. This port contains the lane Status Message from the framing layer Diagnostic Word, bits[33:32]. This message is inserted into the next Diagnostic Word generated by the frame generation block. The message must be held static for 5 cycles before and 5 cycles after the `tx_frame` pulse.
Enable tx_10g_frame_burst_en port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `tx_10g_frame_burst_en` input port. This port controls frame generator data reads from the TX FIFO. The value of this signal is latched once at the beginning of each Metaframe. It controls whether data is read from the TX FIFO or SKIP Words are inserted for the current Metaframe. It must be held static for 5 cycles before and 5 cycles after the `tx_frame` pulse. When `tx_10g_frame_burst_en` is 0, the frame generator does not read data from the TX FIFO for current Metaframe. It insert SKIPs. When `tx_10g_frame_burst_en` is 1, the frame generator reads data from the TX FIFO for current Metaframe.

Interlaken Frame Synchronizer

The Interlaken frame synchronizer block achieves lock by looking for four synchronization words in consecutive metaframes. After synchronization, the frame synchronizer monitors the scrambler word in the metaframe and deasserts the lock signal after three consecutive mismatches and starts the synchronization process again. Lock status is available to the FPGA fabric. The following table describes the Interlaken frame synchronizer parameters.

Table 13-27: Interlaken Frame Synchronizer Parameters

Parameter	Range	Description
teng_tx_framsync_enable	**On/Off**	When you turn this option **On**, the 10G PCS frame generator is enabled.
Enable rx_10g_frame port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame` output port. This signal is asserted to indicate the beginning of a new metaframe inside.
Enable rx_10g_frame_lock_port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame_lock` output port. This signal is asserted to indicate that the frame synchronization state machine has achieved frame lock.
Enable rx_10g_frame_mfrm_err port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame_mfrm_err` output port. This signal is asserted to indicate an metaframe error.
Enable rx_10g_frame_sync_err port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame_sync_err` output port. This signal is asserted to indicate synchronization control word errors. This signal remains asserted during the loss of block_lock and does not update until block_lock is recovered.
Enable rx_10g_frame_skip_ins port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame_skip_ins` output port. This signal is asserted to indicate a SKIP word was received by the frame sync in a non-SKIP word location within the metaframe.
Enable rx_10g_frame_pyld_ins port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame_pyld_ins` output port. This signal is asserted to indicate a SKIP word was not received by the frame sync in a SKIP word location within the metaframe.
Enable rx_10g_frame_skip_err port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame_skip_err` output port. This signal is asserted to indicate the frame synchronization has received an erroneous word in a Skip control word location within the Metaframe. This signal remains asserted during the loss of block_lock and does update until block_lock is recovered.

Parameter	Range	Description
Enable rx_10g_frame_diag_err port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame_diag_err` output port. This signal is asserted to indicate a diagnostic control word error. This signal remains asserted during the loss of block_lock and does update until block_lock is recovered.
Enable rx_10g_frame_diag_status port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame_diag_status` 2-bit output port per channel. This port contains the lane Status Message from the framing layer Diagnostic Word, bits[33:32]. This message is inserted into the next Diagnostic Word generated by the frame generation block.

Interlaken CRC32 Generator and Checker

CRC-32 provides a diagnostic tool on a per-lane basis. You can use CRC-32 to trace interface errors back to an individual lane. The CRC-32 calculation covers the whole metaframe including the Diagnostic Word itself. This CRC code value is stored in the CRC32 field of the Diagnostic Word. The following table describes the CRC-32 parameters.

Table 13-28: Interlaken CRC32 Generator and Checker Parameters

Parameter	Range	Description
Enable Interlaken TX CRC32 Generator	**On/Off**	When you turn this option **On**, the TX 10G PCS datapath includes the CRC32 function.
Enable Interlaken RX CRC32 Generator	**On/Off**	When you turn this option **On**, the RX 10G PCS datapath includes the CRC32 function.
Enable rx_10g_crc32_err port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_crc32_err` port. This signal is asserted to indicate that the CRC checker has found an error in the current metaframe.

10GBASE-R BER Checker

The BER monitor block conforms to the 10GBASE-R protocol specification as described in IEEE 802.3-2008 Clause-49. After block lock is achieved, the BER monitor starts to count the number of invalid synchronization headers within a 125-us period. If more than 16 invalid synchronization headers are observed in a 125-us period, the BER monitor provides the status signal to the FPGA fabric, indicating a high bit error. The following table describes the 10GBASE-R BER checker parameters.

Table 13-29: 10GBASE-R BER Checker Parameters

Parameter	Range	Description
Enable rx_10g_highber port (10GBASE-R)	**On/Off**	When you turn this option **On**, the TX 10G PCS datapath includes the `rx_10g_highber` output port. This signal is asserted to indicate a BER of >10^{-4}. A count of 16 errors in 125- m s period indicates a BER > 10^{-4}. This signal is only available for the 10GBASE-R protocol.
Enable rx_10g_highber_clr_cnt port (10GBASE-R)	**On/Off**	When you turn this option **On**, the TX 10G PCS datapath includes the `rx_10g_highber_clr_cnt` input port. When asserted, the BER counter resets to 0. This signal is only available for the 10GBASE-R protocol.
Enable rx_10g_clr_errblk_count port (10GBASE-R)	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_clr_errblk_count` input port. When asserted, error block counter that counts the number of RX errors resets to 0. This signal is only available for the 10GBASE-R protocol.

64b/66b Encoder and Decoder

The 64b/66b encoder and decoder conform to the 10GBASE-R protocol specification as described in IEEE 802.3-2008 Clause-49. The 64b/66b encoder sub-block receives data from the TX FIFO and encodes the 64-bit data and 8-bit control characters to the 66-bit data block required by the 10GBASE-R protocol. The transmit state machine in the 64b/66b encoder sub-block checks the validity of the 64-bit data from the MAC layer and ensures proper block sequencing.

The 64b/66b decoder sub-block converts the received data from the descrambler into 64-bit data and 8-bit control characters. The receiver state machine sub-block monitors the status signal from the BER monitor. The following table describes the 64/66 encoder and decoder parameters.

Table 13-30: 64b/66b Encoder and Decoder Parameters

Parameter	Range	Description
Enable TX sync header error insertion	**On/Off**	When you turn this option **On**, the 10G PCS records. This parameter is valid for the Interlaken and 10GBASE-R protocols.
Enable TX 64b/66b encoder	**On/Off**	When you turn this option **On**, the 10G PCS includes the TX 64b/66b encoder.
Enable TX 64b/66b encoder	**On/Off**	When you turn this option **On**, the 10G PCS includes the RX 64b/66b decoder.

Scrambler and Descrambler Parameters

TX scrambler randomizes data to create transitions to create DC-balance and facilitate CDR circuits based on the $x^{58} + x^{39} + 1$ polynomial. The scrambler operates in the following two modes:

- Synchronous—The Interlaken protocol requires synchronous mode.
- Asynchronous (also called self-synchronized)—The 10GBASE-R protocol requires this mode as specified in IEEE 802.3-2008 Clause-49.

The descrambler block descrambles received data to regenerate unscrambled data using the x58+x39+1 polynomial. The following table describes the scrambler and descrambler parameters.

Table 13-31: Scrambler and Descrambler Parameters

Parameter	Range	Description
Enable TX scrambler	**On/Off**	When you turn this option **On**, the TX 10G PCS datapath includes the scrambler function. This option is available for the Interlaken and 10GBASE-R protocols.
TX scrambler seed	User-specified 15-bit value	You must provide a different seed for each lane. This parameter is only required for the Interlaken protocol.
Enable RX scrambler	**On/Off**	When you turn this option **On**, the RX 10G PCS datapath includes the scrambler function. This option is available for the Interlaken and 10GBASE-R protocols.
Enable rx_10g_descram_err port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_descram_err` port.

Interlaken Disparity Generator and Checker

The Disparity Generator monitors the data transmitted to ensure that the running disparity remains within a ±96-bit bound. It adds the 67th bit to indicate whether or not the data is inverted. The Disparity Checker monitors the status of the 67th bit of the incoming word to determine whether or not to invert bits[63:0] of the received word. The following table describes Interlaken disparity generator and checker parameters.

Table 13-32: Interlaken Disparity Generator and Checker Parameters

Parameter	Range	Description
Enable Interlaken TX disparity generator	**On/Off**	When you turn this option **On**, the 10G PCS includes the disparity generator. This option is available for the Interlaken protocol.

Parameter	Range	Description
Enable Interlaken RX disparity generator	**On/Off**	When you turn this option **On**, the 10G PCS includes the disparity checker. This option is available for the Interlaken protocol.

Block Synchronization

The block synchronizer determines the block boundary of a 66-bit word for the 10GBASE-R protocol or a 67-bit word for the Interlaken protocol. The incoming data stream is slipped one bit at a time until a valid synchronization header (bits 65 and 66) is detected in the received data stream. After the predefined number of synchronization headers is detected, the block synchronizer asserts `rx_10g_blk_lock` to other receiver PCS blocks down the receiver datapath and to the FPGA fabric. The block synchronizer is designed in accordance with both the Interlaken protocol specification and the 10GBASE-R protocol specification as described in IEEE 802.3-2008 Clause-49.

Table 13-33: Bit Reversal and Polarity Inversion Parameters

Parameter	Range	Description
Enable RX block synchronizer	**On/Off**	When you turn this option **On**, the 10G PCS includes the RX block synchronizer. This option is available for the Interlaken and 10GBASE-R protocols.
Enable rx_10g_blk_lock port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10G_blk_lock` output port. This signal is asserted to indicate the receiver has achieved block synchronization. This option is available for the Interlaken, 10GBASE-R, and other protocols that user the PCS lock state machine to achieve and monitor block synchronization.
Enable rx_10g_blk_sh_err port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10G_blk_sh_err` output port. This signal is asserted to indicate that an invalid sync header has been received. This signal is active after block lock is achieved. This option is available for the Interlaken, 10GBASE-R, and other protocols that user the PCS lock state machine to achieve and monitor block synchroniza-tion.

Gearbox

The gearbox adapts the PMA data width to a wider PCS data width when the PCS is not two or four times the PMA width.

Table 13-34: Gearbox Parameters

Parameter	Range	Description
Enable TX data polarity inversion	**On/Off**	When you turn this option **On**, the gearbox inverts the polarity of TX data allowing you to correct incorrect placement and routing on the PCB.
Enable TX data bitslip	**On/Off**	When you turn this option **On**, the TX gearbox operates in bitslip mode.
Enable RX data polarity inversion	**On/Off**	When you turn this option **On**, the gearbox inverts the polarity of RX data allowing you to correct incorrect placement and routing on the PCB.
Enable RX data bitslip	**On/Off**	When you turn this option **On**, the 10G PCS RX block synchronizer operates in bitslip mode.
Enable tx_10g_bitslip port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `tx_10g_bitslip` input port. The data slips 1 bit for every positive edge of the `tx_10g_bitslip` input. The maximum shift is < *pcswidth*> -1 bits, so that if the PCS is 64 bits wide, you can shift 0-63 bits.
Enable rx_10g_bitslip port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_bitslip` input port. The data slips 1 bit for every positive edge of the `rx_10g_bitslip` input. he maximum shift is < *pcswidth*> -1 bits, so that if the PCS is 64 bits wide, you can shift 0-63 bits.

PRBS Verifier

You can use the PRBS pattern generators for verification or diagnostics. The pattern generator blocks support the following patterns:

- Pseudo-random binary sequence (PRBS)
- Pseudo-random pattern
- Square wave

Table 13-35: PRBS Parameters

Parameter	Range	Description
Enable rx_10g_prbs ports	**On/Off**	When you turn this option **On**, the PCS includes the `rx_10g_prbs_done` , `rx_10g_prbs_err` and `rx_10g_prbs_err_clr`signals to provide status on PRBS operation.

Related Information

Transceiver Archictecture in Stratix V Devices

10G PCS Pattern Generators

The 10G PCS supports the PRBS, pseudo-random pattern, and square wave pattern generators. You enable the pattern generator or verifiers in the 10G PCS, by writing a 1 to the `TX Test Enable` and `RX Test Enable` bits. The following table lists the offsets and registers of the pattern generators and verifiers in the 10G PCS.

Note:
- The 10G PRBS generator inverts its pattern before transmission. The 10G PRBS verifier inverts the received pattern before verification. You may need to invert the patterns if you connect to third-party PRBS pattern generators and checkers.
- All undefined register bits are reserved.

Table 13-36: Pattern Generator Registers

Offset	Bits	R/W	Name	Description
0x12D	[15:0]	R/W	`Seed A for PRP`	Bits [15:0] of seed A for the pseudo-random pattern.
0x12E	[15:0]			Bits [31:16] of seed A for the pseudo-random pattern.
0x12F	[15:0]			Bits [47:21] of seed A for the pseudo-random pattern.
0x130	[9:0]			Bits [57:48] of seed A for the pseudo-random pattern.
0x131	[15:0]	R/W	`Seed B for PRP`	Bits [15:0] of seed B for the pseudo-random pattern.
0x132	[15:0]			Bits [31:16] of seed B for the pseudo-random pattern.
0x133	[15:0]			Bits [47:32] of seed B for the pseudo-random pattern.
0x134	[9:0]			Bits [57:48] of seed B for the pseudo-random pattern.

Offset	Bits	R/W	Name	Description
0x135	[15:12]	R/W	Square Wave Pattern	Specifies the number of consecutive 1s and 0s. The following values are available: 1, 4, 5, 6, 8, and 10.
	[10]	R/W	TX PRBS 7 Enable	Enables the PRBS-7 polynomial in the transmitter.
	[8]	R/W	TX PRBS 23 Enable	Enables the PRBS-23 polynomial in the transmitter.
	[6]	R/W	TX PRBS 9 Enable	Enables the PRBS-9 polynomial in the transmitter.
	[4]	R/W	TX PRBS 31 Enable	Enables the PRBS-31 Polynomial in the transmitter.
	[3]	R/W	TX Test Enable	Enables the pattern generator in the transmitter.
	[1]	R/W	TX Test Pattern Select	Selects between the square wave or pseudo-random pattern generator. The following encodings are defined: • 1'b1: Square wave • 1'b0: Pseudo-random pattern or PRBS
	[0]	R/W	Data Pattern Select	Selects the data pattern for the pseudo-random pattern. The following encodings are defined: • 1'b1: Two Local Faults. Two, 32-bit ordered sets are XORd with the pseudo-random pattern. • 1'b0: All 0's
0x137	[2]	R/W	TX PRBS Clock Enable	Enables the transmitter PRBS clock.
	[1]	R/W	TX Square Wave Clock Enable	Enables the square wave clock.

Offset	Bits	R/W	Name	Description
0x15E	[14]	R/W	RX PRBS 7 Enable	Enables the PRBS-7 polynomial in the receiver.
	[13]	R/W	RX PRBS 23 Enable	Enables the PRBS-23 polynomial in the receiver.
	[12]	R/W	RX PRBS 9 Enable	Enables the PRBS-9 polynomial in the receiver.
	[11]	R/W	RX PRBS 31 Enable	Enables the PRBS-31 polynomial in the receiver.
	[10]	R/W	RX Test Enable	Enables the PRBS pattern verifier in the receiver.
0x164	[10]	R/W	RX PRBS Clock Enable	Enables the receiver PRBS Clock.
0x169	[0]	R/W	RX Test Pattern Select	Selects between a square wave or pseudo-random pattern. The following encodings are defined: • 1'b1: Square wave • 1'b0: Pseudo-random pattern or PRBS

PRBS Pattern Generator

To enable the PRBS pattern generator, write 1'b1 to the `RX PRBS Clock Enable` and `TX PRBS Clock Enable` bits.

The following table shows the available PRBS patterns:

Table 13-37: 10G PCS PRBS Patterns

Pattern	Polynomial
PRBS-31	$X^{31}+x^{28}+1$
PRBS-9	$X^{9}+x^{5}+1$
PRBS-23	$X^{23}+x^{18}+1$
PRBS-7	$X^{7}+x^{6}+1$

Pseudo-Random Pattern Generator

The pseudo-random pattern generator is specifically designed for the 10GBASE-R and 1588 protocols. To enable this pattern generator, write the following bits:

- Write 1'b0 to the `TX Test Pattern Select` bit.
- Write 1'b1 to the `TX Test Enable` bit.

In addition you have the following options:

- You can toggle the `Data Pattern Select` bit switch between two data patterns.
- You can change the value of `Seed A` and `Seed B`.

Unlike the PRBS pattern generator, the pseudo-random pattern generator does not require a configurable clock.

Square Wave Generator

To enable the square wave, write the following bits:

- Write 1'b1 to the `TX Test Enable` bit.
- Write 1'b1 to the `Square Wave Clock Enable` bit.
- Write 1'b1 to the `TX Test Select` bit.
- Write the `Square Wave Pattern` to 1, 4, 5, 6, 8 or 10 consecutive 1s or 0s.

The RX datapath does not include a verifier for the square wave and does drive a clock.

Interfaces for Stratix V Native PHY

This section describes the common, Standard and 10G PCS interfaces for the Stratix V Native PHY.

The Native PHY includes several interfaces that are common to all parameterizations. It also has separate interfaces for the Standard and 10G PCS datapaths. If you use dynamic reconfiguration to change between the Standard and 10G PCS datapaths, your top-level HDL file includes the port for both the Standard and 10G PCS datapaths. In addition, the Native PHY allows you to enable ports, even for disabled blocks to facilitate dynamic reconfiguration.

The Native PHY uses the following prefixes for port names:

- Standard PCS ports—tx_std_, rx_std_
- 10G PCS ports—tx_10g_, rx_10g_
- PMA ports—tx_pma_, rx_pma_

The port descriptions use the following variables to represent parameters:

- <*n*>—The number of lanes
- <*p*>—The number of PLLs
- <*r*>—the number of CDR references clocks selected

Common Interface Ports for Stratix V Native PHY

This section describes the interface ports for the Stratix V native PHY.

Common interface consists of reset, clock signals, serial interface ports, control and status ports, parallel data ports, PMA ports and reconfig interface ports. The following figure illustrates these ports.

Figure 13-5: Stratix V Native PHY Common Interfaces

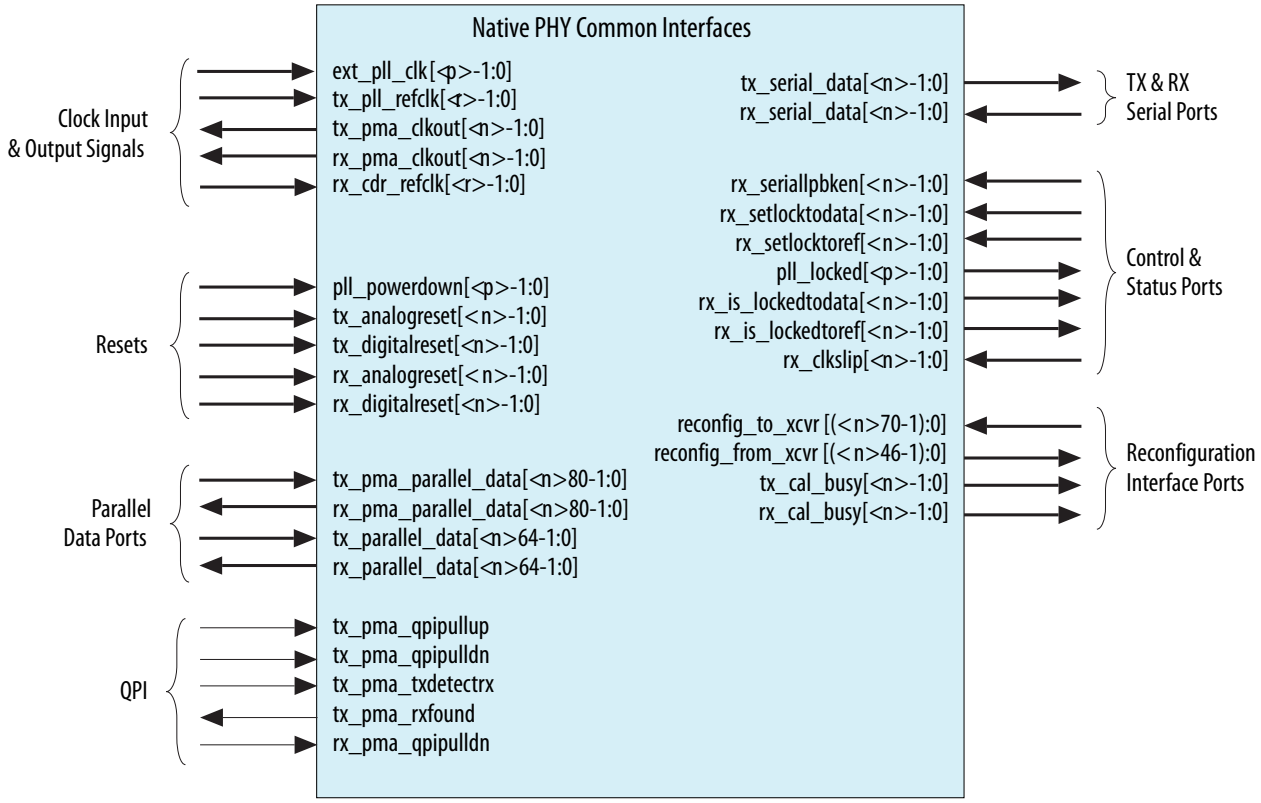


Table 13-38: Native PHY Common Interfaces

Name	Direction	Description
Clock Inputs and Output Signals		
`tx_pll_refclk[ <r> -1:0]`	Input	The reference clock input to the TX PLL.
`tx_pma_clkout[ <n> -1:0]`	Output	TX parallel clock output from PMA
`rx_pma_clkout[ <n> -1:0]`	Output	RX parallel clock (recovered clock) output from PMA
`rx_cdr_refclk[ <n> -1:0]`	Input	Input reference clock for the RX PFD circuit.
`ext_pll_clk[ <p> -1:0]`	Input	This optional signal is created when you select the **Use external TX PLL** option. If you instantiate a fractional PLL which is external to the Native PHY IP, then connect the output clock of this PLL to `ext_pll_clk`.
Resets		

Name	Direction	Description
pll_powerdown[<*p*> -1:0]	Input	When asserted, resets the TX PLL. Active high, edge sensitive reset signal. By default, the Stratix V Native Transceiver PHY IP Cores creates a separate pll_powerdown signal for each logical PLL. However, the Fitter may merge the PLLs if they are in the same transceiver bank. PLLs can only be merged if their pll_powerdown signals are driven from the same source. If the PLLs are in separate transceiver banks, you can choose to drive the pll_powerdown signals separately.
tx_analogreset[<*n*> -1:0]	Input	When asserted, resets for TX PMA, TX clock generation block, and serializer. Active high, edge sensitive reset signal.
tx_digitalreset[<*n*> -1:0]	Input	When asserted, resets the digital components of the TX datapath. Active high, edge sensitive reset signal.If your design includes bonded TX PCS channels, refer to *Timing Constraints for Reset Signals when Using Bonded PCS Channels* for a SDC constraint you must include in your design.
rx_analogreset[<*n*> -1:0]	Input	When asserted, resets the RX CDR, deserializer, Active high, edge sensitive reset signal.
rx_digitalreset[<*n*> -1:0]	Input	When asserted, resets the digital components of the RX datapath. Active high, edge sensitive reset signal.
Parallel Data Ports		
tx_pma_parallel_data[<*n*> 80-1:0]	Input	TX parallel data for the PMA Direct datapath. Driven directly from the FPGA fabric to the PMA. Not used when you enable either the Standard or 10G PCS datapath.
rx_pma_parallel_data[<*n*> 80-1:0]	Output	RX PMA parallel data driven from the PMA to the FPGA fabric. Not used when you enable either the Standard or 10G PCS datapath.

Name	Direction	Description
tx_parallel_data[<*n*> 64-1:0]	Input	PCS TX parallel data. Used when you enable either the Standard or 10G datapath. For the Standard datapath, if you turn on Enable simplified data interface , tx_parallel_data includes only the data and control signals necessary for the current configuration. Dynamic reconfiguration of the interface is not supported. For the 10G PCS, if the parallel data interface is less than 64 bits wide, the low-order bits of tx_parallel_data are valid. For the 10G PCS operating in 66:40 Basic mode, the 66 bus is formed as follows: { tx_parallel_data[63:0],tx_10g_control[0], tx_10g_control[1]}. For the Standard PCS, refer to *Table 12-39: Signal Definiations for tx_parallel_data with and with 8B/10B Encoding* for bit definitions. Refer to *Table 12-40: Location of Valid Data Words for tx_parallel_data for Various FPGA Fabric to PCS Parameterizations* for various parameterizations.
rx_parallel_data[<*n*> 64-1:0]	Output	PCS RX parallel data. Used when you enable either the Standard or 10G datapath. For the Standard datapath, if you turn on Enable simplified data interface , rx_parallel_data includes only the data and control signals necessary for the current configuration. Dynamic reconfiguration of the interface is not supported. For the 10G PCS, if the parallel data interface is less than 64 bits wide, the low-order bits of rx_parallel_data are valid. For the 10G PCS operating in 66:40 mode, the 66 bus is formed as follows: { rx_parallel_data[63:0],rx_10g_control[0], rx_10g_control[1]}. For the Standard PCS, refer to *Table 12-41: Signal Definitions for rx_parallel_data with and without 8B/10B Encoding* for bit definitions. Refer to *Table 12-42: Location of Valid Data Words for rx_parallel_data for Various FPGA Fabric to PCS Parameterizations* for various parameterizations.
QPI		

Name	Direction	Description
tx_pma_qpipullup	Input	Control input port for Quick Path Interconnect (QPI) applications. When asserted, the transmitted pulls the output signal to high state. Use this port only for QPI applications.
tx_pma_qpipulldn	Input	Control input port for Quick Path Interconnect (QPI) applications. This is an active high signal. When asserted, the transmitter pulls the output signal in low state. Use this port only for QPI applications.
tx_pma_txdetectrx	Input	When asserted, the RX detect block in the TX PMA detects the presence of a receiver at the other end of the channel. After receiving a tx_pma_txdetectrx request, the receiver detect block initiates the detection process. Only for QPI applications.
tx_pma_rxfound	Output	Indicates the status of an RX detection in the TX PMA. Only for QPI applications.
rx_pma_qpipulldn	Input	Control input port for Quick Path Interconnect (QPI) applications. This is an active high signal. When asserted, the receiver pulls the input signal in low state. Use this port only for QPI applications.
TX and RX Serial Ports		
tx_serial_data[<*n*> -1:0]	Output	TX differential serial output data.
rx_serial_data[<*n*> -1:0]	Input	RX differential serial output data.
Control and Status Ports		
rx_seriallpbken[<*n*> -1:0]	Input	When asserted, the transceiver enters loopback mode. Loopback drives TX data to the RX interface.
rx_set_locktodata[<*n*> -1:0]	Input	When asserted, programs the RX CDR to manual lock to data mode in which you control the reset sequence using the rx_setlocktoref and rx_setlocktodata. Refer to *Reset Sequence for CDR in Manual Lock Mode* in *Transceiver Reset Control in Stratix V Devices* for more information about manual control of the reset sequence.

Name	Direction	Description
rx_set_locktoref[<*n*> -1:0]	Input	When asserted, programs the RX CDR to manual lock to reference mode in which you control the reset sequence using the rx_setlocktoref and rx_setlocktodata. Refer to *Reset Sequence for CDR in Manual Lock Mode* in *Transceiver Reset Control in Stratix V Devices* for more information about manual control of the reset sequence.
pll_locked[<*p*> -1:0]	Output	When asserted, indicates that the PLL is locked to the input reference clock.
rx_is_lockedtodata[<*n*> -1:0]	Output	When asserted, the CDR is locked to the incoming data.
rx_is_lockedtoref[<*n*> -1:0]	Output	When asserted, the CDR is locked to the incoming reference clock.
rx_clkslip[<*n*> -1:0]	Input	When you turn this option on, the deserializer performs clock slip operation to acheive word alignment. The clock slip operation alternates between skipping 1 serial bit and pausing the serial clock for 2 cycles to achieve word alignment. As a result, the period of the parallel clock could be extended by 2 unit intervals (UI) during the clock slip operation. This is an optional control input signal.
Reconfig Interface Ports		
reconfig_to_xcvr [(<*n*> 70-1) :0]	Input	Reconfiguration signals from the Transceiver Reconfiguration Controller. <*n*> grows linearly with the number of reconfiguration interfaces.
reconfig_from_xcvr [(<*n*> 46-1) :0]	Output	Reconfiguration signals to the Transceiver Reconfiguration Controller. <*n*> grows linearly with the number of reconfiguration interfaces.
tx_cal_busy[<*n*> -1:0]	Output	When asserted, indicates that the initial TX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP. You must hold the channel in reset until calibration completes.
rx_cal_busy[<*n*> -1:0]	Output	When asserted, indicates that the initial RX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP.

Table 13-39: Signal Definitions for `tx_parallel_data` with and without 8B/10B Encoding

The following table shows the signals within `tx_parallel_data` that correspond to data, control, and status signals. The `tx_parallel_data` bus is always 64 bits to enable reconfigurations between the Standard and 10G PCS datapaths. If you only enable the Standard datapath, the 20, high-order bits are not used.

TX Data Word	Description
Signal Definitions with 8B/10B Enabled	
`tx_parallel_data[7:0]`	TX data bus
`tx_parallel_data[8]`	TX data control character
`tx_parallel_data[9]`	Force disparity, validates disparity field.
`tx_parallel_data[10]`	Specifies the current disparity as follows: • 1'b0 = positive • 1'b1 = negative
Signal Definitions with 8B/10B Disabled	
`tx_parallel_data[9:0]`	TX data bus
`tx_parallel_data[10]`	Unused

Table 13-40: Location of Valid Data Words for `tx_parallel_data` for Various FPGA Fabric to PCS Parameterizations

The following table shows the valid 11-bit data words with and without the byte deserializer for single- and double-word FPGA fabric to PCS interface widths.

Configuration	Bus Used Bits
Single word data bus, byte deserializer disabled	[10:0] (word 0)
Single word data bus, byte serializer enabled	[32:22], [10:0] (words 0 and 2)
Double word data bus, byte serializer disabled	[21:0] (words 0 and 1)
Double word data bus, byte serializer enabled	[43:0] (words 0-3)

Table 13-41: Signal Definitions for `rx_parallel_data` with and without 8B/10B Encoding

This table shows the signals within `rx_parallel_data` that correspond to data, control, and status signals.

RX Data Word	Description
Signal Definitions with 8B/10B Enabled	
`rx_parallel_data[7:0]`	RX data bus
`rx_parallel_data[8]`	RX data control character
`rx_parallel_data[9]`	Error Detect
`rx_parallel_data[10]`	Word Aligner / synchronization status
`rx_parallel_data[11]`	Disparity error
`rx_parallel_data[12]`	Pattern detect

RX Data Word	Description
rx_parallel_data[14:13]	The following encodings are defined: • 2'b00: Normal data • 2'b01: Deletion • 2'b10: Insertion • 2'b11: Underflow
rx_parallel_data[15]	Running disparity value
Signal Definitions with 8B/10B Disabled	
rx_parallel_data[9:0]	RX data bus
rx_parallel_data[10]	Word Aligner / synchronization status
rx_parallel_data[11]	Disparity error
rx_parallel_data[12]	Pattern detect
rx_parallel_data[14:13]	The following encodings are defined: • 2'b00: Normal data • 2'b01: Deletion • 2'b10: Insertion (or Underflow with 9'h1FE or 9'h1F7) • 2'b11: Overflow
rx_parallel_data[15]	Running disparity value

Table 13-42: Location of Valid Data Words for rx_parallel_data for Various FPGA Fabric to PCS Parameterizations

The following table shows the valid 16-bit data words with and without the byte deserializer for single- and double-word FPGA fabric to PCS interface widths.

Configuration	Bus Used Bits
Single word data bus, byte deserializer disabled	[15:0] (word 0)
Single word data bus, byte serializer enabled	[47:32], [15:0] (words 0 and 2)
Double word data bus, byte serializer disabled	[31:0] (words 0 and 1)
Double word data bus, byte serializer enabled	[63:0] (words 0-3)

Related Information

Timing Constraints for Bonded PCS and PMA Channels on page 18-11

Standard PCS Interface Ports

This section describes the PCS interface.

Figure 13-6: Standard PCS Interfaces

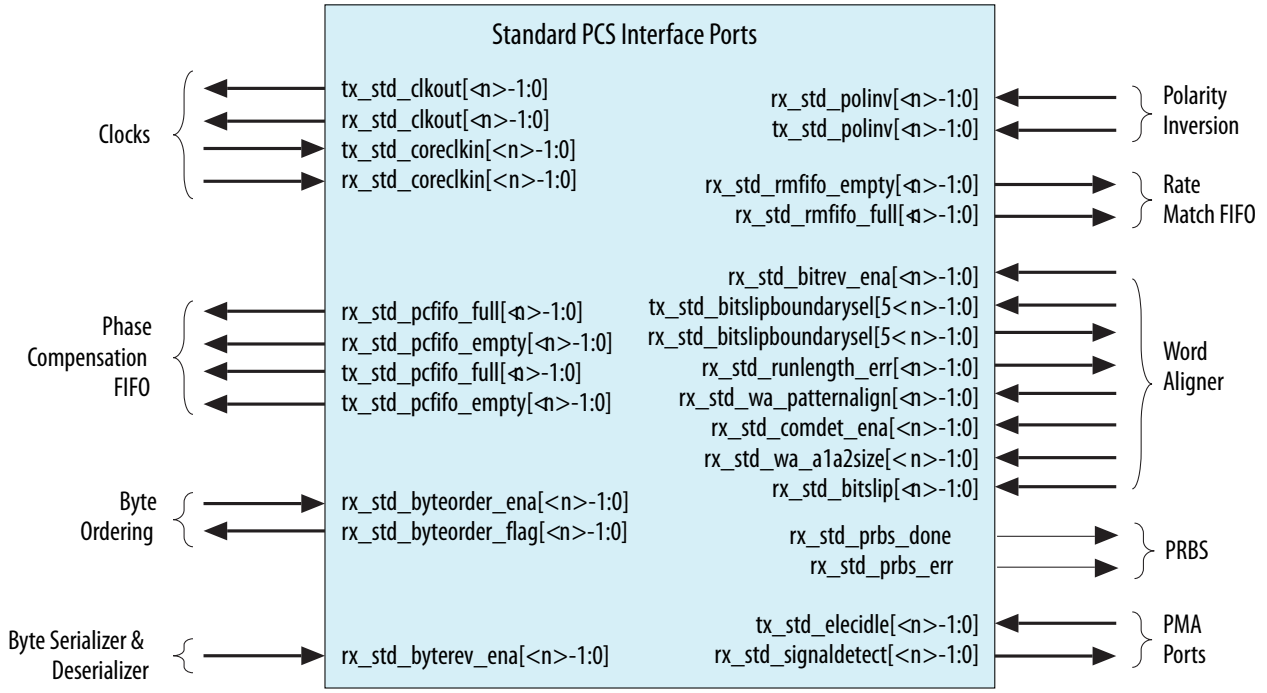


Table 13-43: Standard PCS Interface Ports

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
Clocks			
tx_std_clkout[<n>-1:0]	Output	—	TX Parallel clock output.
rx_std_clkout[<n>-1:0]	Output	—	RX parallel clock output. The CDR circuitry recovers RX parallel clock from the RX data stream.
tx_std_coreclkin[<n>-1:0]	Input	—	TX parallel clock input from the FPGA fabric that drives the write side of the TX phase compensation FIFO.
rx_std_coreclkin[<n>-1:0]	Input	—	RX parallel clock that drives the read side of the RX phase compensation FIFO.
Phase Compensation FIFO			
rx_std_pcfifo_full[<n>-1:0]	Output	Yes	RX phase compensation FIFO full status flag.

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
rx_std_pcfifo_empty[<n>-1:0]	Output	Yes	RX phase compensation FIFO status empty flag.
tx_std_pcfifo_full[<n>-1:0]	Output	Yes	TX phase compensation FIFO status full flag.
tx_std_pcfifo_empty[<n>-1:0]	Output	Yes	TX phase compensation FIFO status empty flag.
Byte Ordering			
rx_std_byteorder_ena[<n>-1:0]	Input	No	Byte ordering enable. When this signal is asserted, the byte ordering block initiates a byte ordering operation if the **Byte ordering control mode** is set to **manual**. Once byte ordering has occurred, you must deassert and reassert this signal to perform another byte ordering operation. This signal is an synchronous input signal; however, it must be asserted for at least 1 cycle of rx_std_clkout.
rx_std_byteorder_flag[<n>-1:0]	Output	Yes	Byte ordering status flag. When asserted, indicates that the byte ordering block has performed a byte order operation. This signal is asserted on the clock cycle in which byte ordering occurred. This signal is synchronous to the rx_std_clkout clock. You must a synchronizer this signal.
Byte Serializer and Deserializer			
rx_std_byterev_ena[<n>-1:0]	Input	No	This control signal is available in when the PMA width is 16 or 20 bits. When asserted, enables byte reversal on the RX interface.
8B/10B			

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
rx_std_polinv[<n>-1:0]	Input	No	Polarity inversion for the 8B/10B decoder, When set, the RX channels invert the polarity of the received data. You can use this signal to correct the polarity of differential pairs if the transmission circuitry or board layout mistakenly swapped the positive and negative signals. The polarity inversion function operates on the word aligner input.
tx_std_polinv[<n>-1:0]	Input	No	Polarity inversion, part of 8B10B encoder, When set, the TX interface inverts the polarity of the TX data.
Rate Match FIFO			
rx_std_rmfifo_empty[<n>-1:0]	Output	No	Rate match FIFO empty flag. When asserted, the rate match FIFO is empty. This port is only used for XAUI, GigE, and Serial RapidIO in double width mode. In double width mode, the FPGA data width is twice the PCS data width to allow the fabric to run at half the PCS frequency
rx_std_rmfifo_full[<n>-1:0]	Output	No	Rate match FIFO full flag. When asserted the rate match FIFO is full. You must synchronize this signal. This port is only used for XAUI, GigE, and Serial RapidIO in double width mode.
Word Aligner			
rx_std_bitrev_ena[<n>-1:0]	Input	No	When asserted, enables bit reversal on the RX interface. Bit order may be reversed if external transmission circuitry transmits the most significant bit first. When enabled, the receive circuitry receives all words in the reverse order. The bit reversal circuitry operates on the output of the word aligner.
tx_std_bitslipboundarysel[5<n>-1:0]	Input	No	BitSlip boundary selection signal. Specifies the number of bits that the TX bit slipper must slip.

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
rx_std_bitslipboundarysel[5<n>-1:0]	Output	No	This signal operates when the word aligner is in bitslip word alignment mode. It reports the number of bits that the RX block slipped to achieve deterministic latency.
rx_std_runlength_err[<n>-1:0]	Output	No	When asserted, indicates a run length violation. Asserted if the number of consecutive 1s or 0s exceeds the number specified in the parameter editor GUI.
rx_st_wa_patternalign	Input	No	Active when you place the word aligner in manual mode. In manual mode, you align words by asserting rx_st_wa_patternalign. rx_st_wa_patternalign is edge sensitive. For more information refer to the *Word Aligner* section in the *Transceiver Architecture in Arria V Devices*.
rx_std_wa_a1a2size[<n>-1:0]	Input	No	Used for the SONET protocol. Assert when the A1 and A2 framing bytes must be detected. A1 and A2 are SONET backplane bytes and are only used when the PMA data width is 8 bits.
rx_std_bitslip[<n>-1:0]	Input	No	Used when word aligner mode is bitslip mode. For every rising edge of the rx_std_bitslip signal, the word boundary is shifted by 1 bit. Each bitslip removes the earliest received bit from the received data. This is an asynchronous input signal and inside there is a synchronizer to synchronize it with rx_pma_clk/rx_clkout.
PRBS			
rx_std_prbs_done	Output	Yes	When asserted, indicates the verifier has aligned and captured consecutive PRBS patterns and the first pass through a polynomial is complete. The generator has restarted the polynomial.

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
rx_std_prbs_err	Output	Yes	When asserted, indicates an error only after the rx_std_prbs_done signal has been asserted. This signal pulses for every error that occurs. Errors can only occur once per word. To clear the PRBS pattern and deassert the rx_std_prbs_done signal by writing to the memory-mapped register PRBS Error Clear that you access through the Transceiver Reconfiguration Controller IP Core.
Miscellaneous			
tx_std_elecidle[<n>-1:0]	Input		When asserted, enables a circuit to detect a downstream receiver. This signal must be driven low when not in use because it causes the TX PMA to enter electrical idle mode with the TX serial data signals in tristate mode.
rx_std_signaldetect[<n>-1:0]	Output	No	Signal threshold detect indicator. When asserted, it indicates that the signal present at the receiver input buffer is above the programmed signal detection threshold value. You must synchronize this signal.

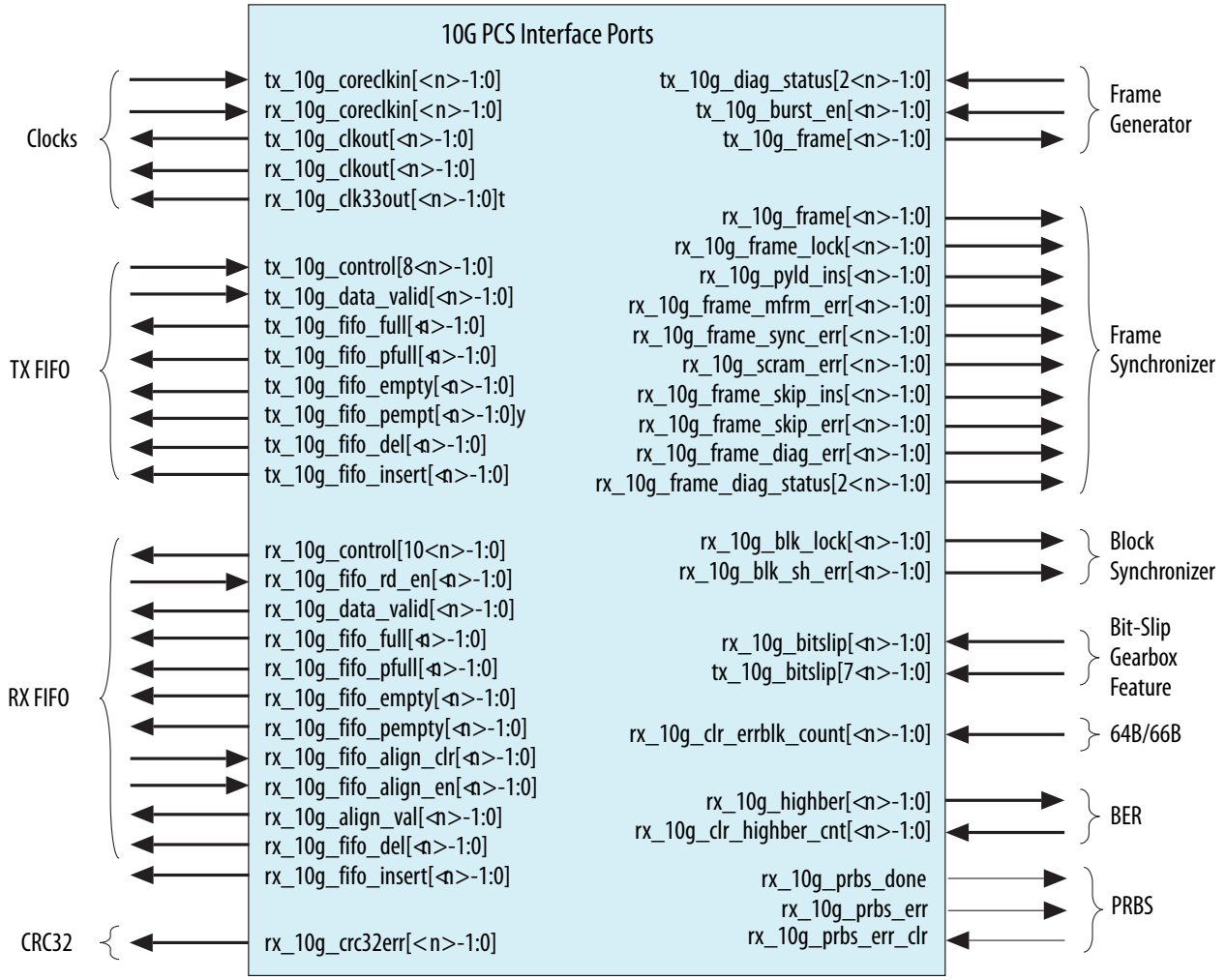
Related Information

Transceiver Architecture in Arria V Devices

10G PCS Interface

The following figure illustrates the top-level signals of the 10G PCS. If you enable both the 10G PCS and Standard PCS your top-level HDL file includes all the interfaces for both.

Figure 13-7: Stratix V Native PHY 10G PCS Interfaces



The following table describes the signals available for the 10G PCS datapath. When you enable both the 10G and Standard datapaths, both sets of signals are included in the top-level HDL file for the Native PHY.

Table 13-44: 10G PCS Interface Signals

The signals in the following table are shown when the Phase Compensation FIFO is used in FIFO mode.

Name	Direction	Description
Clocks		
`tx_10g_coreclkin [<n>-1:0]`	Input	TX parallel clock input that drive the write side of the TX FIFO.
`rx_10g_coreclkin [<n>-1:0]`	Input	RX parallel clock input that drives the read side of the RX FIFO. .

Name	Direction	Description
tx_10g_clkout [<*n*>-1:0]	Output	TX parallel clock output for the TX PCS.
rx_10g_clkout [<*n*>-1:0]	Output	RX parallel clock output which is recovered from the RX data stream.
rx_10g_clk33out [<*n*>-1:0]	Output	This clock is driven by the RX deserializer. Its frequency is RX CDR PLL clock frequency divided by 33 or equivalently the RX PMA data rate divided by 66. It is typically used for ethernet applications that use 66b/64b decoding.
		TX FIFO
tx_10g_control [9<*n*>-1:0]	Input	TX control signals for the Interlaken, 10GBASE-R, and Basic protocols. Synchronous to tx_10g_coreclk_in. The following signals are defined: Interlaken mode: • [8]: Active-high synchronous error insertion control bit • [7:3]: Not Used

Name	Direction	Description
tx_10g_control [9<*n*>-1:0] (continued)		• [2]: Inversion signal, must always be set to 1'b0. • [1]: Sync Header, 1 indicates a control word • [0]: Sync Header, 1 indicates a data word 10G BaseR mode: • [8]: Active-high synchronous error insertion control signal • [7]: MII control signal for tx_data[63:56] • [6]: MII control signal for tx_data[55:48] • [5]: MII control signal fortx_data[47:40] • [4]: MII control signal for tx_data[39:32] • [3]: MII control signal for tx_data[31:24] • [2]: MII control signal for tx_data[23:16] • [1]: MII control signal for tx_data[15:8] • [0]: MII control signal for tx_data[7:0] Basic mode: 67-bit word width: • [8:3]: Not used • [2]: Inversion Bit - must always be set to 1'b0. • [1]: Sync Header, 1 indicates a control word) • [0]: Sync Header, 1 indicates a data word) Basic mode: 66-bit word width: • [8:2]: Not used • [1]: bit 0 of transmit data • [0]: bit 1 of transmit data 66-bit transmit data format: {tx_parallel_data[63:0], tx_10g_control[0], tx_10g_control[1]}.
tx_10g_data_valid [<*n*>-1:0]	Input	When asserted, indicates if tx_data is valid. Synchronous to tx_10g_coreclk_in. Use of this signal depends upon the protocol you are implementing, as follows: • 10G BASE-R: Tie to 1'b1 • Interlaken: Acts as control for FIFO write enable. You should tie this signal to tx_10g_fifo_pempty. • Basic with phase compensation FIFO: Tie to 1'b1 as long as tx_coreclkin = data_rate/pld_pcs interface width. Otherwise, tie this signal to tx_10g_fifo_pempty. • Basic with phase compensation FIFO in register mode. This mode only allows a 1:1 gear box ratio such as 32:32 and 64:64; consequently, you can tie tx_10g_data_valid to 1'b1.
tx_10g_fifo_full [<*n*>-1:0]	Output	When asserted, indicates that the TX FIFO is full. Synchronous to tx_10g_coreclkin.

Name	Direction	Description
tx_10g_fifo_pfull [<*n*>-1:0]	Output	When asserted, indicates that the TX FIFO is partially full. Synchronous to tx_10g_coreclkin.
tx_10g_fifo_empty [<*n*>-1:0]	Output	TX FIFO empty flag. Synchronous to tx_10g_clkout. This signal is pulse-stretched; you must use a synchronizer.
tx_10g_fifo_pempty [<*n*>-1:0]	Output	TX FIFO partially empty flag. Synchronous to tx_10g_clkout. This signal is pulse-stretched; you must use a synchronizer.
tx_10g_fifo_del [<*n*>-1:0]	Output	When asserted, indicates that a word has been deleted from the rate match FIFO. This signal is used for the 10GBASE-R protocol. This signal is synchronous to tx_10g_coreclkin.
tx_10g_fifo_insert [<*n*>-1:0]	Output	When asserted, indicates that a word has been inserted into the rate match FIFO. This signal is used for the 10GBASE-R protocol. This signal is pulse-stretched, you must use a synchronizer. This signal is synchronous to tx_clkout.
		RX FIFO

Name	Direction	Description
rx_10g_control [10<*n*>-1:0]	Output	RX control signals for the Interlaken, 10GBASE-R, and Basic protocols. These are synchronous to rx_10g_coreclkin. The following signals are defined: Interlaken mode: • [9]: Active-high synchronous status signal that indicates when block lock and frame lock are achieved. • [8]: Active-high synchronous status signal that indicates a synchronization header, metaframe or CRC32 error. • [7]: Active-high synchronous status signal that indicates the Diagnostic Word location within a metaframe. • [6]: Active-high synchronous status signal that indicates the SKIP Word location within a metaframe. • [5]: Active-high synchronous status signal that indicates the Scrambler State Word location within a metaframe. • [4]: Active-high synchronous status signal that indicates the Synchronization Word location within a metaframe. • [3]: Active-high synchronous status signal that indicates the Payload Word location within a metaframe. • [2]: Inversion signal, when asserted indicates that the polarity of the signal has been inverted. • [1]: Synchronization header, 1 indicates control word. • [0]: Synchronization header, 1 indicates data word. 10GBASE-R mode: • [9]: Active-high synchronous status signal indicating when Block Lock is achieved • [8]: Active-high status signal that indicates a Idle/OS deletion • [7]: MII control signal for rx_data[63:56] • [6]: MII control signal for rx_data[55:48] • [5]: MII control signal for rx_data[47:40] • [4]: MII control signal for rx_data[39:32] • [3]: MII control signal for rx_data[31:24] • [2]: MII control signal for rx_data[23:16] • [1]: MII control signal for rx_data[15:8] • [0]: MII control signal for rx_data[7:0]

Name	Direction	Description
rx_10g_control [10<*n*>-1:0] (continued)		Basic mode: 67-bit mode with Block Sync: • [9]: Active-high synchronous status signal that indicates when Block Lock is achieved. • [8]: Active-high synchronous status signal that indicates a sync header error • [7:3]: Not used [2]: Used • [1]: Synchronization header, a 1 indicates control word • [0]: Synchronization header, a 1 indicates data word Basic mode: 66-bit mode with Block Sync: [9]: Active-high synchronous status signal that indicates when Block Lock is achieved. [8]: Active-high synchronous status signal that indicates a sync header error. [7:2]: Not used • [1]: Synchronization header, a 1 indicates control word • [0]: Synchronization header, a 1 indicates data word Basic mode: 67-bit mode without Block Sync: [9:3]: Not used 66-bit mode without Block Sync: [9:2]: Not used • [1]: Synchronization header, a 1 indicates control word • [0]: Synchronization header, a 1 indicates data word Basic mode: 64-bit, 50-bit, 40-bit and 32-bit modes: [9:0]: Not used
rx_10g_fifo_rd_en [<*n*>-1:0]	Input	Active high read enable signal for RX FIFO. Asserting this signal reads 1 word from the RX FIFO.
rx_10g_data_valid [<*n*>-1:0]	Output	Active valid data signal with the following use: • 10GBASE-R: Always high • Interlaken: Toggles indicating when rx_data is valid. • Basic - Phase compensation: Toggles indicating when rx_data is valid. • Basic - Register: Toggles indicating when rx_data is valid.
rx_10g_fifo_full [<*n*>-1:0]	Output	Active high RX FIFO full flag. Synchronous to rx_10g_clkout. This signal is pulse-stretched; you must use a synchronizer.

Name	Direction	Description
rx_10g_fifo_pfull [<*n*>-1:0]	Output	RX FIFO partially full flag. Synchronous to rx_10g_clkout. This signal is pulse-stretched; you must use a synchronizer.
rx_10g_fifo_empty [<*n*>-1:0]	Output	Active high RX FIFO empty flag. Synchronous to rx_10g_coreclkin.
rx_10g_fifo_pempty [<*n*>-1:0]	Output	Active high. RX FIFO partially empty flag. Synchronous to rx_10g_coreclkin.
rx_10g_fifo_align_clr [<*n*>-1:0]	Input	For the Interlaken protocol, this signal clears the current word alignment when the RX FIFO acts as a deskew FIFO. When it is asserted, the RX FIFO is reset and searches for a new alignment pattern. Synchronous to rx_10g_coreclkin.
rx_10g_fifo_align_en [<*n*>-1:0]	Input	For the Interlaken protocol, you must assert this signal to enable the RX FIFO for alignment. Synchronous to rx_10g_coreclkin.
rx_10g_align_val [<*n*>-1:0]	Output	For the Interlaken protocol, an active high indication that the alignment pattern has been found. Synchronous to rx_10g_coreclkin.
Rx_10g_fifo_del [<*n*>-1:0]	Output	When asserted, indicates that a word has been deleted from the TX FIFO. This signal is used for the 10GBASE-R protocol. This signal is pulse-stretched; you must use a synchronizer. Synchronous to rx_10g_clkout.
Rx_10g_fifo_insert [<*n*>-1:0]	Output	Active-high 10G BaseR RX FIFO insertion flag. Synchronous to rx_10g_coreclkin. When asserted, indicates that a word has been inserted into the TX FIFO. This signal is used for the 10GBASE-R protocol.
CRC32		
rx_10g_crc32err [<*n*>-1:0]	Output	For the Interlaken protocol, asserted to indicate that the CRC32 Checker has found a CRC32 error in the current metaframe. Is is asserted at the end of current metaframe. This signal is pulse-stretched; you must use a synchronizer. Synchronous to rx_10g_clkout.
Frame Generator		

Name	Direction	Description
tx_10g_diag_status [2<n>-1:0]	Input	For the Interlaken protocol, provides diagnostic status information reflecting the lane status message contained in the Framing Layer Diagnostic Word (bits[33:32]). This message is inserted into the next Diagnostic Word generated by the Frame Generation Block. The message must be held static for 5 cycles before and 5 cycles after the tx_frame pulse. Synchronous to tx_10g_clkout.
tx_10g_burst_en [<n>-1:0]	Input	For the Interlaken protocol, controls frame generator reads from the TX FIFO. Latched once at the beginning of each metaframe.When 0, the frame generator inserts SKIPs. When 1, the frame generator reads data from the TX FIFO. Must be held static for 5 cycles before and 5 cycles after the tx_frame pulse. Synchronous to tx_10g_clkout.
tx_10g_frame [<n>-1:0]	Output	For the Interlaken protocol, asserted to indicate the beginning of a new metaframe inside the frame generator. This signal is pulse-stretched; you must use a synchronizer to synchronize with tx_10g_clkout.
Frame Synchronizer		
rx_10g_frame [<n>-1:0]	Output	For the Interlaken protocol, asserted to indicate the beginning of a new metaframe inside the frame synchronizer. This signal is pulse-stretched, you must use a synchronizer. This signal is pulse-stretched; you must use a synchronizer to synchronize with rx_10g_clkout.
rx_10g_frame_lock [<n>-1:0]	Output	For the Interlaken protocol, asserted to indicate that the frame synchronizer state machine has achieved frame lock. This signal is pulse-stretched, you must use a synchronizer. This signal is pulse-stretched; you must use a synchronizer to synchronize with rx_10g_clkout.
Rx_10g_pyld_ins [<n>-1:0]	Output	For the Interlaken protocol, asserted to indicate a SKIP Word was not received by the frame synchronizer in a SKIP Word location within the metaframe. This signal is pulse-stretched, you must use a synchronizer. This signal is pulse-stretched; you must use a synchronizer to synchronize with rx_10g_clkout.
rx_10g_frame_mfrm_err [<n>-1:0]	Output	For the Interlaken protocol, asserted to indicate an error has occurred in the metaframe. This signal is pulse-stretched, you must use a synchronizer to synchronize with rx_10g_clkout.

Name	Direction	Description
rx_10g_frame_sync_err [<*n*>-1:0]	Output	For the Interlaken protocol, asserted to indicate a synchronization Control Word error was received in a synchronization Control Word location within the metaframe. This signal is sticky if block lock is lost and does not update until block lock is re-established.This signal is pulse-stretched; you must use a synchronizer to synchronize with rx_10g_clkout.
rx_10g_scram_err [<*n*>-1:0]	Output	For the Interlaken protocol, asserted to indicate, Scrambler Control Word errors in a Scrambler Control Word location within the metaframe. This signal is sticky during the loss of block lock and does not update until block lock is re-established. This signal is pulse-stretched; you must use a synchronizer to synchronize with rx_10g_clkout.
rx_10g_frame_skip_ins [<*n*>-1:0]	Output	For the Interlaken protocol, asserted to indicate to a SKIP Word was received by the frame synchronizer in a non-SKIP Word location within the metaframe. This signal is pulse-stretched; you must use a synchronizer to synchronize with rx_10g_clkout.
rx_10g_frame_skip_err [<*n*>-1:0]	Output	For the Interlaken protocol, asserted to indicate a Skip Control Word error was received in a Skip Control Word location within the metaframe. This signal is sticky during the loss of block lock and does not update until block lock is re-established. This signal is pulse-stretched; you must use a synchronizer to synchronize with rx_10g_clkout.
rx_10g_frame_diag_ err[<*n*>-1:0]	Output	For the Interlaken protocol, asserted to indicate a Diagnostic Control Word error was received in a Diagnostic Control Word location within the metaframe. This signal is sticky during the loss of block lock and does not update until block lock is re-established. This signal is pulse-stretched; you must use a synchronizer to synchronize with rx_10g_clkout.
rx_10g_frame_diag_status [*2*<*n*>-1:0]	Output	For the Interlaken protocol, reflects the lane status message contained in the framing layer Diagnostic Word (bits[33:32]). This information is latched when a valid Diagnostic Word is received in a Diagnostic Word Metaframe location. This signal is pulse-stretched; you must use a synchronizer to synchronize with rx_10g_ clkout.
Block Synchronizer		

Name	Direction	Description
rx_10g_blk_lock [<*n*>-1:0]	Output	Active-high status signal that is asserted when block synchronizer acquires block lock. Valid for the 10GBASE-R and Interlaken protocols, and any basic mode that uses the lock state machine to achieve and monitor block synchronization for word alignment. Once the block synchronizer acquires block lock, it takes at least 16 errors for rx_10g_blk_lock to be deasserted.
rx_10g_blk_sh_err [<*n*>-1:0]	Output	Error status signal from block synchronizer indicating an invalid synchronization header has been received. Valid for the 10GBASE-R and Interlaken protocols, and any legal basic mode that uses the lock state machine to achieve and monitor block synchronization for word alignment. Active only after block lock is achieved. This signal is generated by rx_pma_clk and is pulse-stretched by 3 clock cycles. You must use a synchronizer.
Bit-Slip Gearbox Feature Synchronizer		
rx_10g_bitslip [<*n*>-1:0]	Input	User control bit-slip in the RX Gearbox. Slips one bit per rising edge pulse.
tx_10g_bitslip [7<*n*>-1:0]	Input	TX bit-slip is controlled by tx_bitslip port. Shifts the number of bit location specified by tx_bitslip. The maximum shift is <*pcswidth-1*>.
64b/66b		
rx_10g_clr_errblk_count [<*n*>-1:0]	Input	For the 10GBASE-R protocol, asserted to clear the error block counter which counts the number of times the RX state machine enters the RX error state.
BER		
rx_10g_highber [<*n*>-1:0]	Output	For the 10GBASE-R protocol, status signal asserted to indicate a bit error ratio of $>10^{-4}$. A count of 16 in 125us indicates a bit error ratio of $>10^{-4}$. Once asserted, it remains high for at least 125 us.
rx_10g_clr_highber_cnt [<*n*>-1:0]	Input	For the 10GBASE-R protocol, status signal asserted to clear the BER counter which counts the number of times the BER state machine enters the BER_BAD_SH state. This signal has no effect on the operation of the BER state machine.
PRBS		
rx_10g_prbs_done	Output	When asserted, indicates the verifier has aligned and captured consecutive PRBS patterns and the first pass through a polynomial is complete.

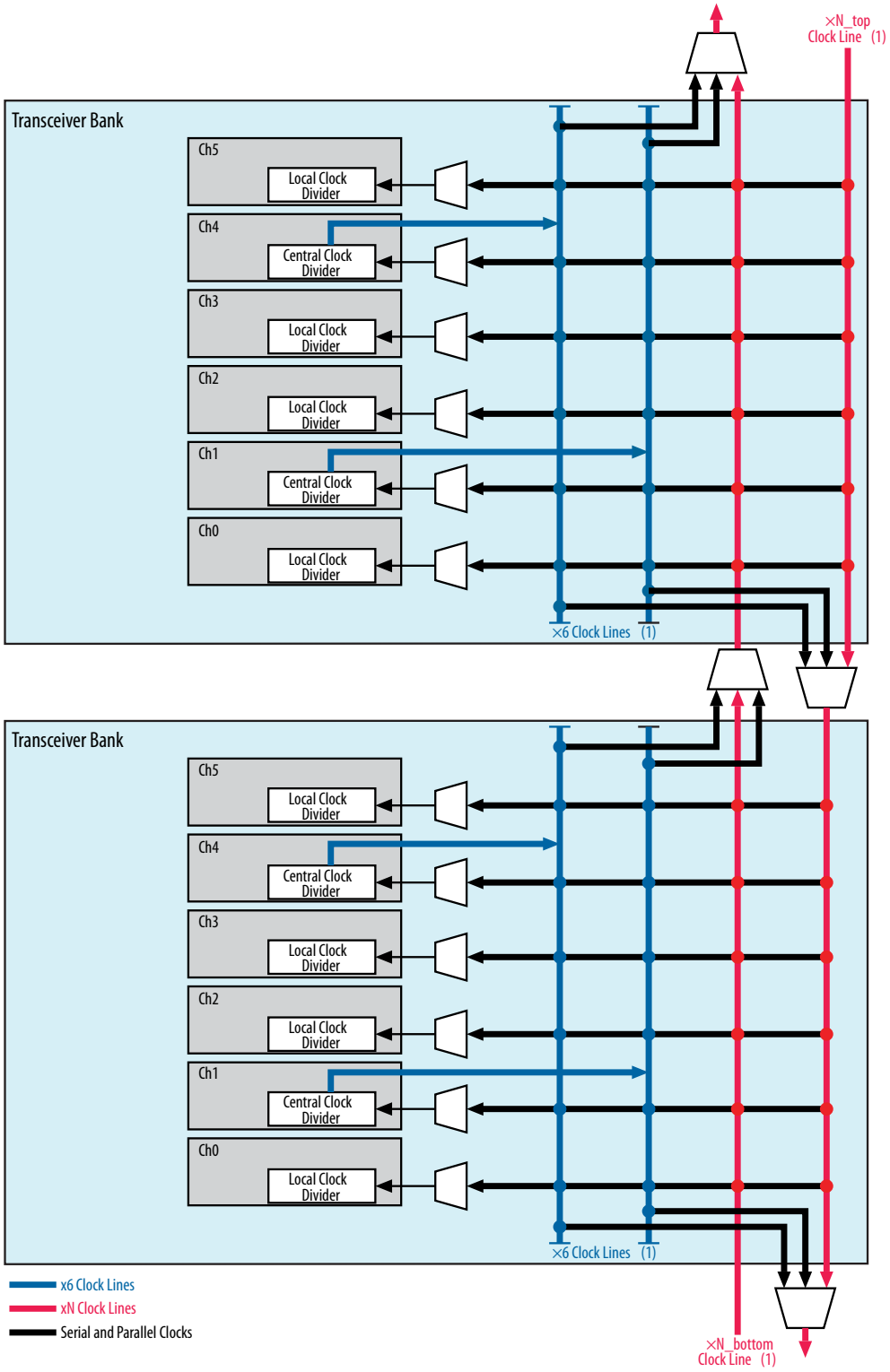
Name	Direction	Description
rx_10g_prbs_err	Output	When asserted, indicates an error only after the rx_10g_prbs_done signal has been asserted. This signal pulses for every error that occurs. An error can only occur once per word. This signal indicates errors for both the PRBS and pseudo-random patterns. Synchronous to rx_10g_coreclkin.
rx_10g_prbs_err_clr	Input	When asserted, clears the PRBS pattern and de-asserts the rx_10g_prbs_done signal. Synchronous to rx_10g_coreclkin.

×6/×N Bonded Clocking

The Native PHY supports bonded clocking in which a single TX PLL generates the clock that drives the transmitter for up to 27 contiguous channels. Bonded configurations conserve PLLs and reduce channel-to-channel clock skew. Bonded channels do not support dynamic reconfiguration of the transceiver.

When you specify **×6/×N** bonding, the transceiver channels that reside in the same bank as the TX PLL are driven over the x6 clock line. Channels outside of the this bank are driven on the **×N** clock lines, as the following figure illustrates.

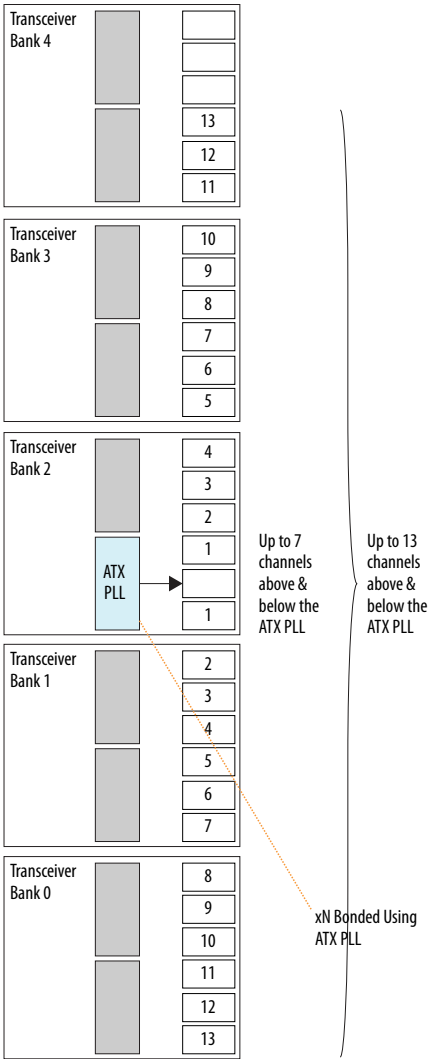
Figure 13-8: x6 and xN Routing of Clocks



Note: (1) The the x6 and xN clock lines also carry both serial and parallel clocks.

Bonded clocks allow you to use the same PLL for up to 13 contiguous channels above and below the TX PLL for a total of 27 bonded channels as the following figure illustrates.

Figure 13-9: Channel Span for xN Bonded Channels



You can use the `tx_clkout` from any channel to transfer data, control, and status signals between the FPGA fabric and the transceiver channels. Using the `tx_clkout` from the central channel results in overall lower clock skew across lanes. In the FPGA fabric, you can drive the `tx_clkout` from the connected channel to all other channels in the bonded group. For bonded clocking, connecting more than one `tx_clkout` from the transceiver channel to the FPGA fabric results in a Fitter error. You can also choose the `tx_pll_refclk` to transfer data, control, and status signals between the FPGA fabric and the transceiver channels. Because this reference clock is also the input to the TX PLL, it has the required 0 ppm difference with respect to `tx_clkout`.

ATX, CMU and Fractional PLLs

For data rates above 8 Gbps, Altera recommends the ATX PLL because it has better jitter performance. Refer to "Clock Network Maximum Data Rate Transmitter Specifications" in the *Stratix V Device Datasheet* for detailed information about maximum data rates for the three different PLLs. The supported data rates are somewhat higher when a design specifies up to 7 contiguous channels above and below the ATX PLL rather than the maximum of 13 contiguous channels above and below the ATX PLL.

You can also use the CMU or fractional PLLs at lower data rates. If you select the CMU PLL as the TX PLL it must be placed in physical channel 1 or 4 of the transceiver bank. That channel is not available as an RX channel because the CMU PLL is not available to recover the clock from received data. Consequently, the using the CMU PLL creates a gap in the contiguous channels.

Related Information

- **Stratix V Device Datasheet**
- **Transceiver Clocking in Stratix V Devices**

xN Non-Bonded Clocking

Non-bonded clocking routes only the high-speed serial clock from the TX PLL to the transmitter channels. The local clock divider of each channel generates the low-speed parallel clock. Non-bonded channels support dynamic reconfiguration of the transceiver.

xN non-bonded clocking has the following advantages:

- Supports data rate negotiation between link partners on a per-channel basis.
- Supports data rates are not simple integer multiples of a single base data rate.
- Supports PLL and channel reconfiguration.

The Native PHY preset for CPRI specifies non-bonded clocks. In multi-channel configurations, CPRI can use both ATX PLLs in a transceiver bank to generate two base data rates. When necessary, CPRI uses dynamic reconfiguration to change the local clock dividers to generate the negotiated clock rate.

The channel span for xN non-bonded clocks is almost identical to the span for bonded clocks as illustrated in **Figure 13-9**. However, the center channel that provides central clock divider cannot be used as a data channel because this channel cannot generate the parallel clock. The maximum channel span is 26 channels. There is a single-channel break in the contiguous channel sequence.

Related Information

Transceiver Clocking in Stratix V Devices

SDC Timing Constraints of Stratix V Native PHY

This section describes SDC examples and approaches to identify false timing paths.

The Intel Quartus Prime software reports timing violations for asynchronous inputs to the Standard PCS and 10G PCS. Because many violations are for asynchronous paths, they do not represent actual timing failures. You may choose one of the following three approaches to identify these false timing paths to the Intel Quartus Prime or TimeQuest software.

In all of these examples, you must substitute you actual signal names for the signal names shown.

Example 13-1: Using the set_false_path Constraint to Identify Asynchronous Inputs

You can cut these paths in your Synopsys Design Constraints (.sdc) file by using the set_false_path command as shown in following example.

```
set_false_path -through {*10gtxbursten*} -to [get_registers
*10g_tx_pcs*SYNC_DATA_REG*]

set_false_path -through {*10gtxdiagstatus*} -to [get_registers
*10g_tx_pcs*SYNC_DATA_REG*]

set_false_path -through {*10gtxwordslip*} -to [get_registers
*10g_tx_pcs*SYNC_DATA_REG*]

set_false_path -through {*10gtxbitslip*} -to [get_registers
*10g_tx_pcs*SYNC_DATA_REG*]

set_false_path -through {*10grxbitslip*} -to [get_registers
*10g_rx_pcs*SYNC_DATA_REG*]

set_false_path -through {*10grxclrbercount*} -to [get_registers
*10g_rx_pcs*SYNC_DATA_REG*]

set_false_path -through {*10grxclrerrblkcnt*} -to [get_registers
*10g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*10grxprbserrclr*} -to [get_registers
*10g_rx_pcs*SYNC_DATA_REG*]

set_false_path -through {*8gbitslip*} -to  [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*8gbytordpld*} -to [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*8gcmpfifoburst*} -to [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*8gphfifoburstrx*} -to [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]

set_false_path -through {*8gsyncsmen*} -to [get_registers
*8g*pcs*SYNC_DATA_REG*]
set_false_path -through {*8gwrdisablerx*} -to [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*rxpolarity*} -to [get_registers *SYNC_DATA_REG*]
set_false_path -through {*pldeidleinfersel*} -to [get_registers
*SYNC_DATA_REG*]
```

Example 13-2: Using the max_delay Constraint to Identify Asynchronous Inputs

You can use the set_max_delay constraint on a given path to create a constraint for asynchronous signals that do not have a specific clock relationship but require a maximum path delay. The following example illustrates this approach.

```
# Example: Apply 10ns max delay

set_max_delay -from *tx_from_fifo* -to *8g*pcs*SYNC_DATA_REG1 10
```

Example 13-3: Using the set_false TimeQuest Constraint to Identify Asynchronous Inputs

You can use the set_false path command only during Timequest timing analysis. The following example illustrates this approach.

```
#if {$::TimeQuestInfo(nameofexecutable) eq "quartus_fit"} {

#} else {

#set_false_path -from [get_registers {*tx_from_fifo*}] -through
{*txbursten*} -to [get_registers *8g_*_pcs*SYNC_DATA_REG
```

Dynamic Reconfiguration for Stratix V Native PHY

Dynamic reconfiguration calibrates each channel to compensate for variations due to process, voltage, and temperature (PVT).

As silicon progresses towards smaller process nodes, circuit performance is affected more by variations due to PVT. These process variations result in analog voltages that can be offset from required ranges. The calibration performed by the dynamic reconfiguration interface compensates for variations due to PVT.

For more information about transceiver reconfiguration refer to Chapter 16, Transceiver Reconfiguration Controller IP Core.

Example 13-4: Informational Messages for the Transceiver Reconfiguration Interface

For non-bonded clocks, each channel and each TX PLL has a separate dynamic reconfiguration interfaces. The MegaWizard Plug-In Manager provides informational messages on the connectivity of these interfaces. The following example shows the messages for the Stratix V Native PHY with four duplex channels, four TX PLLs, in a non-bonded configuration.

```
PHY IP will require 8 reconfiguration interfaces for connection to the
external reconfiguration controller.
Reconfiguration interface offsets 0-3 are connected to the transceiver
channels.
Reconfiguration interface offsets 4–7 are connected to the transmit PLLs.
```

Example 13-5: Overriding Logical Channel 0 Channel Assignment Restrictions in Stratix V Device for ×6 or ×N Bonding

If you are using ×6 or ×N bonding, transceiver dynamic reconfiguration requires that you assign the starting channel number. Logical channel 0 should be assigned to either physical transceiver channel 1 or channel 4 of a transceiver bank. However, if you have already created a PCB with a different lane assignment for logical lane 0, you can use the workaound shown in the following example to remove this restriction. The following example redefines the pma_bonding_master parameter using the Intel Quartus Prime Assignment Editor. In this example, the pma_bonding_master was originally assigned to physical channel 1. (The original assignment could also have been to physical channel 4.) The to parameter reassigns the pma_bonding_master

to the Deterministic Latency PHY instance name. You must substitute the instance name from your design for the instance name shown in quotation marks

```
set_parameter -name pma_bonding_master "\"1\"" -to "<PHY IP instance name>"
```

Simulation Support

The Intel Quartus Prime release provides simulation and compilation support for the Stratix V Native PHY IP Core. Refer to Running a Simulation Testbench for a description of the directories and files that the Intel Quartus Prime software creates automatically when you generate your Stratix V Transceiver Native PHY IP Core.

Slew Rate Settings

The following transceiver slew rate settings are allowed in Intel Quartus Prime software.

Table 13-45: Slew Rate Settings for Stratix V devices

Protocol / Datarate	Allowed Intel Quartus Prime Settings	IBIS-AMI Settings
PCI Express Gen3, Gen2, CEI	4	*_30ps
PCI Express Gen1, XAUI	3	*_50ps
Gigabit Ethernet	1	*_160ps
SATA_1500 sub-protocol	3	*_50ps
SATA_3000 sub-protocol	3	*_50ps
SATA_6000 sub-protocol	4	*_30ps
=<1 Gbps	1, 2, 3	*_160ps, *_90ps, *_50ps
1 Gbps - 3 Gbps	2, 3	*_90ps, *_50ps
3 Gbps - 6 Gbps	3, 4	*_50ps, *_30ps
>6 Gbps	5	*_15ps

Assigning an invalid slew rate will result in an error message similar to the one below:

Error (15001): Assignment XCVR_TX_SLEW_RATE_CTRL of value "4" conflicts with the valid parameter values for pm_tx_slew_rate_ctrl

- Protocol declarations take priority over datarate. For example, XAUI has a per-lane datarate of 3.125 Gbps, but only a setting of "3" is allowed. A setting of "4" is not allowed for XAUI.
- For protocols not listed in the table, you should use the slew settings associated with your datarate.
- The IBIS-AMI slew rate figure is defined as the approximate transmitter 20% - 80% rise time. The "ps" figure should not be considered quantitative and is an approximate label only.
- The IBIS-AMI models will allow you to simulate any slew rate setting for any datarate or protocol.

Arria V Transceiver Native PHY IP Core 14

2020.06.02

UG-01080 Subscribe Send Feedback

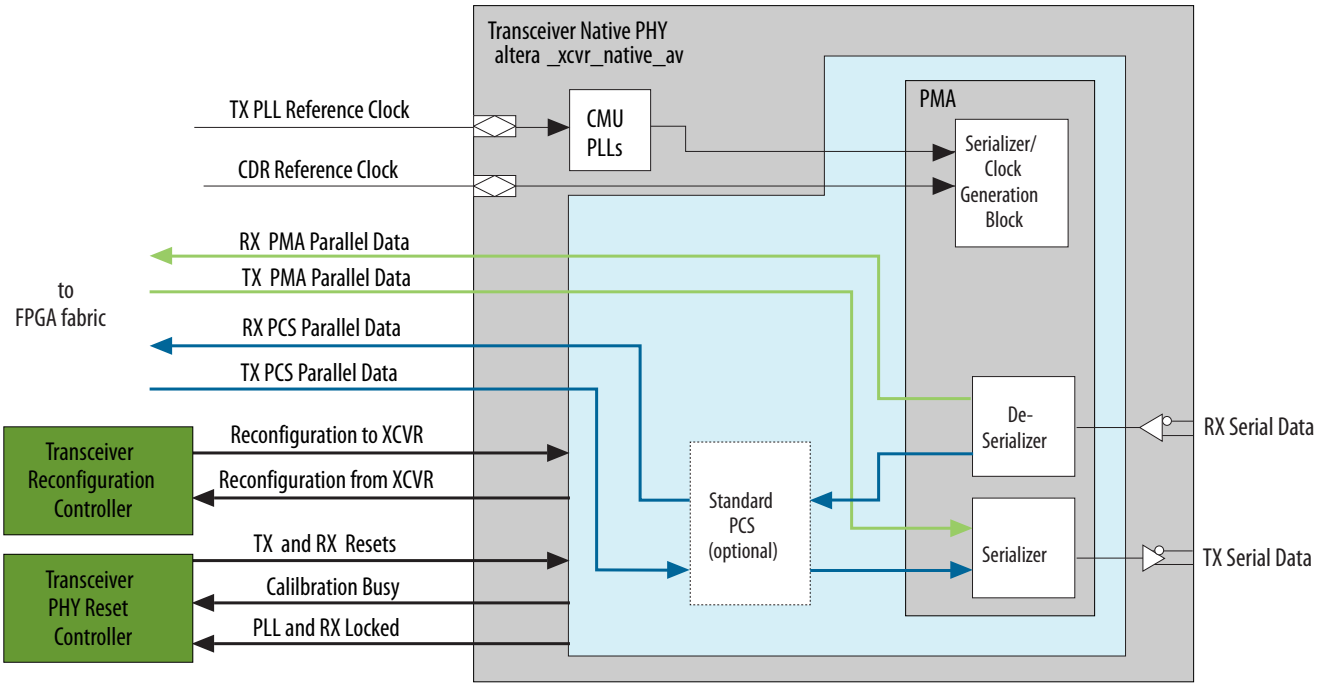
The Arria V Transceiver Native PHY IP Core provides direct access to all control and status signals of the transceiver channels. Unlike other PHY IP Cores, the Native PHY IP Core does not include an Avalon Memory-Mapped (Avalon-MM) interface. Instead, it exposes all signals directly as ports. The Arria V Transceiver Native PHY IP Core provides the following datapaths:

- Standard PCS—When you enable the Standard PCS, you can select the PCS functions and control and status ports that your transceiver PHY requires.
- PMA Direct—When you select PMA Direct mode, the Native PHY provides direct access to the PMA from the FPGA fabric; consequently, the latency for transmitted and received data is lower. However, you must implement any PCS function that your design requires in the FPGA fabric. PMA Direct mode is supported for Arria V GT, ST, and GZ devices only.

The Native Transceiver PHY does not include an embedded reset controller. You can either design custom reset logic or incorporate Altera's "Transceiver PHY Reset Controller IP Core" to implement reset functionality. The Native Transceiver PHY's primary use in Arria V GT devices is for data rates greater than 6.5536 Gbps.

As the following figure illustrates, TX PLL and clock data recovery (CDR) reference clocks from the pins of the device are input to the PLL module and CDR logic. When enabled, the Standard PCS drives TX parallel data and receives RX parallel data. In PMA Direct mode, the PMA serializes TX data it receives from the fabric and drives RX data to the fabric.

Figure 14-1: Arria Native Transceiver PHY IP Core



In a typical design, the separately instantiated Transceiver PHY Reset Controller drives reset signals to Native PHY and receives calibration and locked status signal from the Native PHY. The Native PHY reconfiguration buses connect the external Transceiver Reconfiguration Controller for calibration and dynamic reconfiguration of the channel and PLLs.

You specify the initial configuration when you parameterize the IP core. The Transceiver Native PHY IP Core connects to the "Transceiver Reconfiguration Controller IP Core" to dynamically change reference clocks, PLL connectivity, and the channel configurations at runtime.

Device Family Support

IP cores provide either final or preliminary support for target Altera device families. These terms have the following definitions:

- *Final support*—Verified with final timing models for this device.
- *Preliminary support*—Verified with preliminary timing models for this device.

Table 14-1: Device Family Support

Device Family	Support
Arria V devices	Final
Other device families	No support

Performance and Resource Utilization

This section describes performance and resource utilization for the IP core.

Because the Standard PCS and PMA are implemented in hard logic, the Arria V Native PHY IP Core requires minimal resources.

Parameterizing the Arria V Native PHY

By default, the Arria V Native PHY Transceiver PHY IP defaults to the PMA direct datapath and an internal PLL. You can change the default configuration to include the PCS or an external fractional PLL.

1. Under **Tools** > **IP Catalog**, select **Arria V** as the device family.
2. Under **Tools** > **IP Catalog** > **Interface Protocols** > **Transceiver PHY**, select **Arria V Native PHY**.
3. Use the tabs on the MegaWizard Plug-In Manager to select the options required for the protocol.
4. Click **Finish** to generate your customized Arria V Native PHY IP Core.

Note: The Arria V Transceiver Native PHY provides presets for CPRI, GIGE, and the Low Latency Standard PCS. The presets specify the parameters required to the protocol specified.

General Parameters

This section lists the parameters available on the **General Options** tab.

Table 14-2: General and Datapath Options

Name	Range	Description
Device speed grade	**3fastest–6_H6**	Specifies the speed grade.
Message level for rule violations	**error warning**	Allows you to specify the message level, as follows: • **error**: Intel Quartus Prime checker will not create an instance with invalid parameters. You must change incompatible parameter selections to proceed. • **warning**: Intel Quartus Prime checker will allow instance creation with invalid parameters, but the instance will not compile successfully.
Datapath Options		
Enable TX datapath	**On/Off**	When you turn this option **On**, the core includes the TX datapath.
Enable RX datapath	**On/Off**	When you turn this option **On**, the core includes the RX datapath.
Enable Standard PCS	**On/Off**	When you turn this option **On**, the core includes the **Standard PCS**.

Name	Range	Description
Number of data channels	1-36	Specifies the total number of data channels in each direction.
Bonding mode	**Bonded or xN** **Non-bonded or x1**	In **Non–bonded** mode, each channel is assigned a PLL. If one PLL drives multiple channels, PLL merging is required. During compilation, the Intel Quartus Prime Fitter, merges all the PLLs that meet PLL merging requirements. Refer to **Merging TX PLLs In Multiple Transceiver PHY Instances** on page 17-58 to observe PLL merging rules. Select **×N** to use the same clock source for up to 6 channels in a single transceiver bank or the same clock source for all the transceivers on one side of the device. ×N bonding results in reduced clock skew. You must use contiguous channels when you select ×N bonding. For more information about the clock architecture of bonding, refer to "Transmitter Clock Network" in *Transceiver Clocking in Arria V Devices* in volume 2 of the *Arria V Device Handbook*.
Enable simplified data interface	**On/Off**	When you turn this option **On**, the data interface provides only the relevant interface to the FPGA fabric for the selected configuration. You can only use this option for static configurations. When you turn this option **Off**, the data interface provides the full physical interface to the fabric. Select this option if you plan to use dynamic reconfiguration that includes changing the interface to the FPGA fabric. Refer to "Active Bits for Each Fabric Interface Width" for guidance.

Related Information

Transceiver Clocking in Arria V Devices

PMA Parameters

This section describes the options available for the PMA.

For more information about the PMA, refer to the *PMA Architecture* section in the *Transceiver Architecture in Arria V Devices*. Some parameters have ranges where the value is specified as Device Dependent. For such parameters, the possible range of frequencies and bandwidths depends on the device, speed grade, and other design characteristics. Refer to *Device Datasheet for Arria V Devices* for specific data for Arria V devices.

Table 14-3: PMA Options

Parameter	Range	Description
Data rate	Device Dependent	Specifies the data rate. The maximum data rate is 12.5 Gbps.
PMA direct interface width[13]	**8, 10, 16, 20, 64, 80**	Specifies the PMA to FPGA fabric interface width for PMA Direct mode.
TX local clock division factor	**1, 2, 4, 8**	Specifies the value of the divider available in the transceiver channels to divide the input clock to generate the correct frequencies for the parallel and serial clocks. This divisor divides the fast clock from the PLL in nonbonded configurations.
PLL base data rate	Device Dependent	Shows the base data rate of the clock input to the TX PLL.The **PLL base data rate** is computed from the **TX local clock division factor** multiplied by the **data rate**. Select a **PLL base data rate** that minimizes the number of PLLs required to generate all the clocks for data transmission. By selecting an appropriate **PLL base data rate**, you can change data rates by changing the **TX local clock division factor** used by the clock generation block.

Related Information

- **Transceiver Architecture in Arria V Devices**
- **Device Datasheet for Arria V Devices**

TX PMA Parameters

This section describes the TX PMA options you can specify.

Note: For more information about PLLs in Arria V devices, refer to the *Arria V PLLs* section in *Clock Networks and PLLs in Arria V Devices.*

[13] PMA Direct mode is supported for Arria V GT, ST, and GZ devices only.

Table 14-4: TX PMA Parameters

Parameter	Range	Description
Enable TX PLL dynamic reconfiguration	**On/Off**	When you turn this option **On**, you can dynamically reconfigure the PLL. This option is also required to simulate TX PLL reconfiguration. If you turn this option **On**, the Intel Quartus Prime Fitter prevents PLL merging by default; however, you can specify merging using the `XCVR_TX_PLL_RECONFIG_GROUP` QSF assignment.
Use external TX PLL	**On/Off**	When you turn this option **On**, the Native PHY does not automatically instantiate a TX PLL. Instead, you must instantiate an external PLL and connect it to the `ext_pll_clk[<p> -1 : 0]` port of the Arria V Native PHY. Use the Arria V Transceiver PLL IP Core to instantiate a CMU PLL. Use Altera Phase-Locked Loop (ALTERA_PLL) Megafunction to instantiate a fractional PLL.
Number of TX PLLs	**1–4**	Specifies the number of TX PLLs that can be used to dynamically reconfigure channels to run at multiple data rates. If your design does not require transceiver TX PLL dynamic reconfiguration, set this value to 1. The number of actual physical PLLs that are implemented depends on the selected clock network. Each channel can dynamically select between n PLLs, where n is the number of PLLs specified for this parameter. **Note:** Refer to *Transceiver Clocking in Arria V Devices* chapter for more details.
Main TX PLL logical index	**0–3**	Specifies the index of the TX PLL used in the initial configuration.
Number of TX PLL reference clocks	**1–5**	Specifies the total number of reference clocks that are used by all the PLLs.

Related Information

Transceiver Clocking in Arria V Devices

TX PLL Parameters

This section allows you to define multiple TX PLLs for your Native PHY. The Native PHY GUI provides a separate tab for each TX PLL.

Table 14-5: TX PLL Parameters

Parameter	Range	Description
PLL type	**CMU**	This is the only PLL type available.
PLL base data rate	Device Dependent	Shows the base data rate of the clock input to the TX PLL.The **PLL base data rate** is computed from the **TX local clock division factor** multiplied by the **Data rate**. Select a **PLL base data rate** that minimizes the number of PLLs required to generate all the clocks for data transmission. By selecting an appropriate **PLL base data rate**, you can change data rates by changing the **TX local clock division factor** used by the clock generation block.
Reference clock frequency	Device Dependent	Specifies the frequency of the reference clock for the **Selected reference clock source** index you specify. You can define a single frequency for each PLL. You can use the Transceiver Reconfiguration Controller to dynamically change the reference clock input to the PLL. Note that the list of frequencies updates dynamically when you change the **Data rate**. The Input clock frequency drop down menu is populated with all valid frequencies derived as a function of the **Data rate** and **Base data rate**.
Selected reference clock source	**0–4**	You can define up to 5 reference clock sources for the PLLs in your core. The **Reference clock frequency** selected for index **0**, is assigned to TX PLL<0>. The **Reference clock frequency** selected for index **1**, is assigned to TX PLL<1>, and so on.
Selected clock network	**x1 ×N**	Selects the clock network for the TX PLL. In non-bonded mode, each channel is assigned to one PLL. PLL merging is required when multiple channels are assigned to one PLL. During compilation, the Intel Quartus Prime Fitter, merges all the PLLs that meet PLL merging requirements. Refer to **Merging TX PLLs In Multiple Transceiver PHY Instances** on page 17-58 for more details.

RX PMA Parameters

Note: For more information about the CDR circuitry, refer to the Receiver PMA Datapath section in the *Transceiver Architecture in Arria V Devices* .

Table 14-6: RX PMA Parameters

Parameter	Range	Description
Enable CDR dynamic reconfiguration	**On/Off**	When you turn this option **On**, you can dynamically change the data rate of the CDR circuit.
Number of CDR reference clocks	**1–5**	Specifies the number of reference clocks for the CDRs.
Selected CDR reference clock	**0–4**	Specifies the index of the selected CDR reference clock.
Selected CDR reference clock frequency	Device Dependent	Specifies the frequency of the clock input to the CDR.
PPM detector threshold	**+/- 1000 PPM**	Specifies the maximum PPM difference the CDR can tolerate between the input reference clock and the recovered clock.
Enable rx_pma_clkout port	**On/Off**	When you turn this option **On**, the RX parallel clock which is recovered from the serial received data is an output of the PMA.
Enable rx_is_lockedto-data port	**On/Off**	When you turn this option **On**, the `rx_is_lockedtodata` port is an output of the PMA.
Enable rx_is_lockedtoref port	**On/Off**	When you turn this option **On**, the `rx_is_lockedtoref` port is an output of the PMA.
Enable rx_set_lockedto-data and rx_set_ locktoref ports	**On/Off**	When you turn this option **On**, the `rx_set_lockedtdata` and `rx_set_lockedtoref` ports are outputs of the PMA.
Enable rx_pma_bitslip_ port	**On/Off**	When you turn this option **On**, the `rx_pma_bitslip` is an input to the core. The deserializer slips one clock edge each time this signal is asserted. You can use this feature to minimize uncertainty in the serialization process as required by protocols that require a datapath with deterministic latency such as CPRI.
Enable rx_seriallpbken port	**On/Off**	When you turn this option **On**, the `rx_seriallpbken` is an input to the core. When your drive a 1 on this input port, the PMA operates in serial loopback mode with TX data looped back to the RX channel.

The following table lists the best case latency for the most significant bit of a word for the RX deserializer for the PMA Direct datapath. PMA Direct mode is supported for Arria V GT, ST, and GZ devices only.

Table 14-7: Latency for RX Deserialization in Arria V Devices

FPGA Fabric Interface Width	Arria V Latency in UI
8 bits	19
10 bits	23
16 bits	35
20 bits	43
64 bits	99
80 bits	123

The following table lists the best- case latency for the LSB of the TX serializer for all supported interface widths for the PMA Direct datapath.

Table 14-8: Latency for TX Serialization n Arria V Devices

FPGA Fabric Interface Width	Arria V Latency in UI
8 bits	43
10 bits	53
16 bits	67
20 bits	83
64 bits	131
80 bits	163

The following table shows the bits used for all FPGA fabric to PMA interface widths. Regardless of the FPGA Fabric Interface Width selected, all 80 bits are exposed for the TX and RX parallel data ports. However, depending upon the interface width selected not all bits on the bus will be active. The following table shows which bits are active for each FPGA Fabric Interface Width selection. For example, if your interface is 16 bits, the active bits on the bus are [17:0] and [7:0] of the 80 bit bus. The non-active bits are tied to ground.

Table 14-9: Active Bits for Each Fabric Interface Width

FPGA Fabric Interface Width	Bus Bits Used
8 bits	[7:0]
10 bits	[9:0]
16 bits	{[17:10], [7:0]}
20 bits	[19:0]
40 bits	[39:0]
64 bits	{[77:70], [67:60], [57:50], [47:40], [37:30], [27:20], [17:10], [7:0]}
80 bits	[79:0]

Related Information

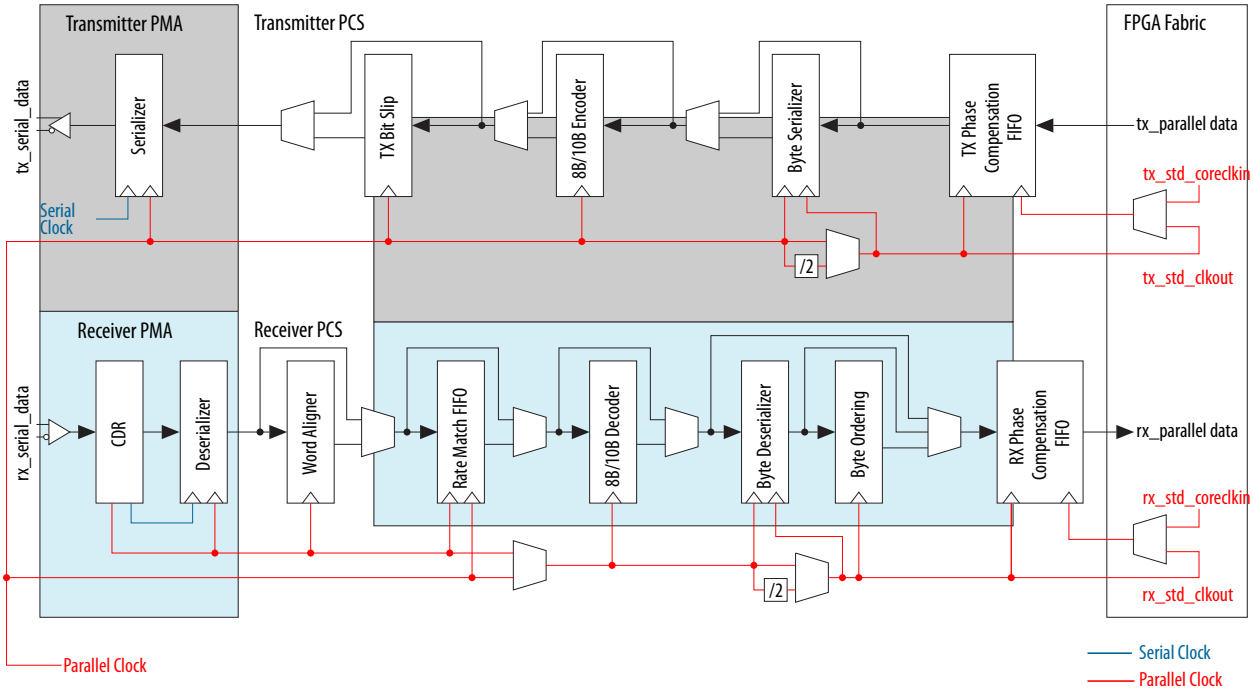
Transceiver Architecture in Arria V Devices

Standard PCS Parameters

This section describes the standard PCS parameters.

The following figure shows the complete datapath and clocking for the Standard PCS. You use parameters available in the GUI to enable or disable the individual blocks in the Standard PCS.

Figure 14-2: The Standard PCS Datapath



Note: For more information about the Standard PCS, refer to the *PCS Architecture* section in the *Transceiver Architecture in Arria V Devices*.

The following table describes the general and datapath options for the Standard PCS.

Table 14-10: General and Datapath Parameters

Parameter	Range	Description
Standard PCS protocol mode	**basic cpri gige**	Specifies the protocol that you intend to implement with the Native PHY. The protocol mode selected guides the MegaWizard in identifying legal settings for the Standard PCS datapath. Use the following guidelines to select a protocol mode: • **basic**–select this mode for when none of the other options are appropriate. You should also select this mode to enable diagnostics, such as loopback. • **cpri**–select this mode if you intend to implement CPRI or another protocol that requires deterministic latency. Altera recommends that you select the appropriate CPRI preset for the CPRI protocol. • **gige**–select this mode if you intend to implement either the 1.25 Gbps or 2.5 Gbps Ethernet protocol. Altera recommends that you select the appropriate preset for the Ethernet protocol.
Standard PCS/PMA interface width	**8, 10,16, 20**	Specifies the width of the datapath that connects the FPGA fabric to the PMA. The transceiver interface width depends upon whether you enable 8B/10B. To simplify connectivity between the FPGA fabric and PMA, the bus bits used are not contiguous for 16 and 32bit buses. Refer to Active Bits for Each Fabric Interface Width for the bits used.
FPGA fabric/Standard TX PCS interface width	**8, 10,16, 20, 32, 40**	Shows the FPGA fabric to TX PCS interface width which is calculated from the **Standard PCS/PMA interface width .**
FPGA fabric/Standard RX PCS interface width	**8, 10,16, 20, 32, 40**	Shows the FPGA fabric to RX PCS interface width which is calculated from the **Standard PCS/PMA interface width .**
Enable 'Standard PCS' low latency mode	**On/Off**	When you turn this option **On**, all PCS functions are disabled except for the phase compensation FIFO, byte serializer and byte deserializer. This option creates the lowest latency Native PHY that allows dynamic reconfigure between multiple PCS datapaths.

Phase Compensation FIFO

The phase compensation FIFO assures clean data transfer to and from the FPGA fabric by compensating for the clock phase difference between the lowspeed parallel clock and FPGA fabric interface clock.

Note: For more information refer to the *Receiver Phase Compensation FIFO* and *Transmitter Phase Compensation FIFO* sections in the *Transceiver Architecture in Arria V Devices*.

Table 14-11: Phase Compensation FIFO Parameters

Parameter	Range	Description
TX FIFO mode	**low_latency** **register_fifo**	The following 2 modes are possible: • **low_latency**: This mode adds 3–4 cycles of latency to the TX datapath. • **register_fifo**: In this mode the FIFO is replaced by registers to reduce the latency through the PCS. Use this mode for protocols that require deterministic latency, such as CPRI.
RX FIFO mode	**low_latency** **register_fifo**	The following 2 modes are possible: • **low_latency**: This mode adds 2–3 cycles of latency to the TX datapath. • **register_fifo**: In this mode the FIFO is replaced by registers to reduce the latency through the PCS. Use this mode for protocols that require deterministic latency, such as CPRI.
Enable tx_std_pcfifo_full port	**On/Off**	When you turn this option **On**, the TX Phase compensation FIFO outputs a FIFO full status flag.
Enable tx_std_pcfifo_empty port	**On/Off**	When you turn this option **On**, the TX Phase compensation FIFO outputs a FIFO empty status flag.
Enable rx_std_pcfifo_full port	**On/Off**	When you turn this option **On**, the RX Phase compensation FIFO outputs a FIFO full status flag.
Enable rx_std_pcfifo_empty port	**On/Off**	When you turn this option **On**, the RX Phase compensation FIFO outputs a FIFO empty status flag.
Enable rx_std_rmfifo_empty port	**On/Off**	When you turn this option **On**, the rate match FIFO outputs a FIFO empty status flag. The rate match FIFO compensates for small clock frequency differences between the upstream transmitter and the local receiver clocks by inserting or removing skip (SKP) symbols or ordered sets from the interpacket gap (IPG) or idle stream.
Enable rx_std_rmfifo_full port	**On/Off**	When you turn this option **On**, the rate match FIFO outputs a FIFO full status flag.

Related Information

Transceiver Architecture in Arria V Devices

Byte Ordering Block Parameters

This section describes the byte ordering block parameters.

The RX byte ordering block realigns the data coming from the byte deserializer. This block is necessary when the PCS to FPGA fabric interface width is greater than the PCS datapath. Because the timing of the RX PCS reset logic is indeterminate, the byte ordering at the output of the byte deserializer may or may not match the original byte ordering of the transmitted data.

Note: For more information refer to the Byte Ordering section in the *Transceiver Architecture in Arria V Devices*.

Table 14-12: Byte Ordering Block Parameters

Parameter	Range	Description
Enable RX byte ordering	**On/Off**	When you turn this option **On**, the PCS includes the byte ordering block.
Byte ordering control mode	**Manual** **Auto**	Specifies the control mode for the byte ordering block. The following modes are available: • **Manual**: Allows you to control the byte ordering block • **Auto**: The word aligner automatically controls the byte ordering block once word alignment is achieved.
Byte ordering pattern width	**8–10**	Shows width of the pattern that you must specify. This width depends upon the PCS width and whether or not 8B/10B encoding is used as follows:

Width	**8B/10B**	**Pad Pattern**
8, 16,32	No	8 bits
10,20,40	No	10 bits
8,16,32	Yes	9 bits

Byte ordering symbol count	**1–2**	Specifies the number of symbols the word aligner should search for. When the PMA is 16 or 20 bits wide, the byte ordering block can optionally search for 1 or 2 symbols.
Byte order pattern (hex)	User-specified 8-10 bit pattern	Specifies the search pattern for the byte ordering block.

Parameter	Range	Description
Byte order pad value (hex)	User–specified 8-10 bit pattern	Specifies the pad pattern that is inserted by the byte ordering block. This value is inserted when the byte order pattern is recognized. The byte ordering pattern should occupy the least significant byte (LSB) of the parallel TX data. If the byte ordering block identifies the programmed byte ordering pattern in the most significant byte (MSB) of the byte-deserialized data, it inserts the appropriate number of user-specified pad bytes to push the byte ordering pattern to the LSB position, restoring proper byte ordering.
Enable rx_std_byteorder_ena port	**On/Off**	Enables the optional `rx_std_byte_order_ena` control input port. When this signal is asserted, the byte ordering block initiates a byte ordering operation if the **Byte ordering control mode** is set to **manual**. Once byte ordering has occurred, you must deassert and reassert this signal to perform another byte ordering operation. This signal is an synchronous input signal; however, it must be asserted for at least 1 cycle of `rx_std_clkout`.
Enable rx_std_byteorder_flag port	**On/Off**	Enables the optional `rx_std_byteorder_flag` status output port. When asserted, indicates that the byte ordering block has performed a byte order operation. This signal is asserted on the clock cycle in which byte ordering occurred. This signal is synchronous to the `rx_std_clkout` clock.

Related Information

Transceiver Architecture in Arria V Devices

Byte Serializer and Deserializer

The byte serializer and deserializer allow the PCS to operate at twice the data width of the PMA serializer. This feature allows the PCS to run at a lower frequency and accommodate a wider range of FPGA interface widths.

Note: For more information refer to the Byte Serializer and Byte Deserializer sections in the *Transceiver Architecture in Arria V Devices*.

Table 14-13: Byte Serializer and Deserializer Parameters

Parameter	Range	Description
Enable TX byte serializer	**On/Off**	When you turn this option **On**, the PCS includes a TX byte serializer which allows the PCS to run at a lower clock frequency to accommodate a wider range of FPGA interface widths.

Parameter	Range	Description
Enable RX byte deserializer	**On/Off**	When you turn this option **On**, the PCS includes an RX byte deserializer which allows the PCS to run at a lower clock frequency to accommodate a wider range of FPGA interface widths.

Related Information

Transceiver Architecture in Arria V Devices

8B/10B

The 8B/10B encoder generates 10-bit code groups from the 8-bit data and 1-bit control identifier.

In 8-bit 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. The 8B/10B decoder decodes the data into an 8-bit data and 1-bit control identifier.

Note: For more information refer to the *8B/10B Encoder* and *8B/10B Decoder* sections in the *Transceiver Architecture in Arria V Devices*.

Table 14-14: 8B/10B Encoder and Decoder Parameters

Parameter	Range	Description
Enable TX 8B/10B encoder	**On/Off**	When you turn this option **On**, the PCS includes the 8B/10B encoder.
Enable TX 8B/10B disparity control	**On/Off**	When you turn this option **On**, the PCS includes disparity control for the 8B/10B encoder. You force the disparity of the 8B/10B encoder using the `tx_forcedisp` and `tx_dispval` control signal.
Enable RX 8B/10B decoder	**On/Off**	When you turn this option **On**, the PCS includes the 8B/10B decoder.

Related Information

Transceiver Architecture in Arria V Devices

Rate Match FIFO

The rate match FIFO compensates for the very small frequency differences between the local system clock and the RX recovered clock.

For more information refer to the Rate Match FIFO sections in the *Transceiver Architecture in Arria V Devices*.

Table 14-15: Rate Match FIFO Parameters

Parameter	Range	Description
Enable RX rate match FIFO	**On/Off**	When you turn this option **On**, the PCS includes a FIFO to compensate for the very small frequency differences between the local system clock and the RX recovered clock.
RX rate match insert/ delete +ve pattern (hex)	User-specified 20 bit pattern	Specifies the +ve (positive) disparity value for the RX rate match FIFO as a hexadecimal string.
RX rate match insert/ delete -ve pattern (hex)	User-specified 20 bit pattern	Specifies the -ve (negative) disparity value for the RX rate match FIFO as a hexadecimal string.

When you enable the simplified data interface and enable the rate match FIFO status ports, the rate match FIFO bits map to the high-order bits of the data bus as listed in the following table. This table uses the following definitions:

- Basic double width: The **Standard PCS protocol mode** GUI option is set to **basic**. The FPGA data width is twice the PCS data width to allow the fabric to run at half the PCS frequency.
- Serial^{TM} RapidIO double width: You are implementing the Serial RapidIO protocol. The FPGA data width is twice the PCS data width to allow the fabric to run at half the PCS frequency.

Note: If you have the auto-negotiation state machine in your transceiver design, please note that the rate match FIFO is capable of inserting or deleting the first two bytes (K28.5//D2.2) of /C2/ ordered sets during auto-negotiation. However, the insertion or deletion of the first two bytes of /C2/ ordered sets can cause the auto-negotiation link to fail. For more information, visit **Altera Knowledge Base Support Solution**.

Table 14-16: Status Flag Mappings for Simplified Native PHY Interface

Status Condition	Protocol	Mapping of Status Flags to RX Data	Value
Full	PHY IP Core for PCI Express (PIPE) Basic double width	`RXD[62:62] = rx_rmfifostatus[1:0]`, or `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[30:29] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b11 = full`
	XAUI, GigE, Serial RapidIO double width	`rx_std_rm_fifo_full`	`1'b1 = full`
	All other protocols	Depending on the FPGA fabric to PCS interface width either: `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b11 = full`
Empty	PHY IP Core for PCI Express (PIPE) Basic double width	`RXD[62:62] = rx_rmfifostatus[1:0]`, or `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[30:29] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`(2'b10) (PAD or EDB)` (14) PAD = K23.7 or 9'h1F7 EDB = K30.7 or 9'h1FE
	XAUI, GigE, Serial RapidIO double width	`rx_std_rm_fifo_empty`	`1'b1 = empty`
	All other protocols	Depending on the FPGA fabric to PCS interface width either: `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`(2'b10) AND (PAD or EDB)` PAD = K23.7 or 9'h1F7 EDB = K30.7 or 9'h1FE

(14) PAD and EBD are control characters. PAD character is typically used fo fill in the remaining lanes in a multi-lane link when one of the link goes to logical idle state. EDB indicates End Bad Packet.

Status Condition	Protocol	Mapping of Status Flags to RX Data	Value
Insertion	Basic double width Serial RapidIO double width	`RXD[62:62] = rx_rmfifostatus[1:0]`, or `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[30:29] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b10`
	All other protocols	Depending on the FPGA fabric to PCS interface width either: `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b10`
Deletion	Basic double width Serial RapidIO double width	`RXD[62:62] = rx_rmfifostatus[1:0]`, or `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[30:29] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b01`
	All other protocols	Depending on the FPGA fabric to PCS interface width either: `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b01`

Related Information

Transceiver Architecture in Arria V Devices

Word Aligner and BitSlip Parameters

The word aligner aligns the data coming from RX PMA deserializer to a given word boundary. When the word aligner operates in bitslip mode, the word aligner slips a single bit for every rising edge of the bit slip control signal.

Note: For more information refer to the Word Aligner section in the *Transceiver Architecture in Arria V Devices*.

Table 14-17: Word Aligner and BitSlip Parameters

Parameter	Range	Description
Enable TX bit slip	**On/Off**	When you turn this option **On**, the PCS includes the bitslip function. The outgoing TX data can be slipped by the number of bits specified by the `tx_bitslipboundarysel` control signal.
Enable tx_std_bitslipboundarysel control input port.	**On/Off**	When you turn this option **On**, the PCS includes the optional `tx_std_bitslipboundarysel` control input port.
RX word aligner mode	**bit_slip** **sync_sm** **Manual**	Specifies one of the following 3 modes for the word aligner: • **bit_slip**: You can use bit slip mode to shift the word boundary. For every rising edge of the `rx_bitslip` signal, the word boundary is shifted by 1 bit. Each bitslip removes the earliest received bit from the received data. • **sync_sm**: In synchronous state machine mode, a programmable state machine controls word alignment. You can only use this mode with 8B/10B encoding. The data width at the word aligner can be 10 or 20 bits. When you select this word aligner mode, the synchronous state machine has hysteresis that is compatible with XAUI. However, when you select **cpri** for the **Standard PCS Protocol Mode**, this option selects the deterministic latency word aligner mode. • **Manual**: This mode enables word alignment by asserting the `rx_std_wa_pattern`. This is an edge sensitive signal.
RX word aligner pattern length	**7, 8, 10, 16, 20, 32, 40**	Specifies the length of the pattern the word aligner uses for alignment. The pattern is specified in LSBtoMSB order.
RX word aligner pattern (hex)	User-specified	Specifies the word aligner pattern in hex.
Number of word alignment patterns to achieve sync	`1–256`	Specifies the number of valid word alignment patterns that must be received before the word aligner achieves synchronization lock. The default is 3.

Parameter	Range	Description
Number of invalid words to lose sync	1–256	Specifies the number of invalid data codes or disparity errors that must be received before the word aligner loses synchronization. The default is 3.
Number of valid data words to decrement error count	1–256	Specifies the number of valid data codes that must be received to decrement the error counter. If the word aligner receives enough valid data codes to decrement the error count to 0, the word aligner returns to synchronization lock.
Run length detector word count	0–63	Specifies the maximum number of contiguous 0s or 1s in the data stream before the word aligner reports a run length violation.
Enable rx_std_wa_patternalign port	**On/Off**	Enables the optional `rx_std_wa_patternalign` control input port.
Enable rx_std_wa_a1a2size port	**On/Off**	Enables the optional `rx_std_wa_a1a2size` control input port.
Enable rx_std_bitslipboundarysel port	**On/Off**	Enables the optional `rx_std_wa_bitslipboundarysel` status output port.
Enable rx_std_bitslip port	**On/Off**	Enables the optional `rx_std_wa_bitslip` control input port.
Enable rx_std_runlength_err port	**On/Off**	Enables the optional `rx_std_wa_runlength_err` control input port.

Related Information

Transceiver Architecture in Arria V Devices

Bit Reversal and Polarity Inversion

The bit reversal and polarity inversion functions allow you to reverse bit order, byte order, and polarity to correct errors and to accommodate different layouts of data.

Table 14-18: Bit Reversal and Polarity Inversion Parameters

Parameter	Range	Description
Enable TX bit reversal	**On/Off**	When you turn this option **On**, the word aligner reverses TX parallel data before transmitting it to the PMA for serialization. You can only change this static setting using the Transceiver Reconfiguration Controller.

Parameter	Range	Description
Enable RX bit reversal	**On/Off**	When you turn this option **On**, the `rx_std_bitrev_ena` port controls bit reversal of the RX parallel data after it passes from the PMA to the PCS.
Enable RX byte reversal	**On/Off**	When you turn this option **On**, the word aligner reverses the byte order before transmitting data. This function allows you to reverse the order of bytes that were erroneously swapped. The PCS can swap the ordering of both 8 and10 bit words.
Enable TX polarity inversion	**On/Off**	When you turn this option **On**, the `tx_std_polinv` port controls polarity inversion of TX parallel data before transmitting the parallel data to the PMA.
Enable RX polarity inversion	**On/Off**	When you turn this option **On**, asserting `rx_std_polinv` controls polarity inversion of RX parallel data after PMA transmission.
Enable rx_std_bitrev_ena port	**On/Off**	When you turn this option **On**, asserting `rx_std_bitrev_ena` control port causes the RX data order to be reversed from the normal order, LSB to MSB, to the opposite, MSB to LSB. This signal is an asynchronous input.
Enable rx_std_byterev_ena port	**On/Off**	When you turn this option **On**, asserting `rx_std_byterev_ena` input control port swaps the order of the individual 8 or 10bit words received from the PMA.
Enable tx_std_polinv port	**On/Off**	When you turn this option **On**, the `tx_std_polinv` input is enabled. You can use this control port to swap the positive and negative signals of a serial differential link if they were erroneously swapped during board layout.

Parameter	Range	Description
Enable rx_std_polinv port	**On/Off**	When you turn this option **On**, the `rx_std_polinv` input is enabled. You can use this control port to swap the positive and negative signals of a serial differential link if they were erroneously swapped during board layout.
Enable tx_std_elecidle port	**On/Off**	When you turn this option **On**, the `tx_std_elecidle` input port is enabled. When this signal is asserted, it forces the transmitter to electrical idle.
Enable rx_std_signaldetect port	**On/Off**	When you turn this option **On**, the optional `rx_std_signaldetect` output port is enabled. This signal is required for the PCI Express protocol. If enabled, the signal threshold detection circuitry senses whether the signal level present at the RX input buffer is above the signal detect threshold voltage that you specified. For SATA / SAS applications, enable this port and set the following QSF assignments to the transceiver receiver pin: • `set_instance_assignment -name XCVR_RX_SD_ENABLE ON` • `set_instance_assignment -name XCVR_RX_SD_THRESHOLD 7` • `set_instance_assignment -name XCVR_RX_COMMON_MODE_VOLTAGE VTT_OP55V` • `set_instance_assignment -name XCVR_RX_SD_OFF 1` • `set_instance_assignment -name XCVR_RX_SD_ON 2` **Note:** The above QSF assignments provide you an example, but the actual setting may vary as per the recommendations from the characterization report. You are also required to set up the platform device level pairing.

Interfaces

The Native PHY includes several interfaces that are common to all parameterizations.

The Native PHY allows you to enable ports, even for disabled blocks to facilitate dynamic reconfiguration.

The Native PHY uses the following prefixes for port names:

- Standard PCS ports—`tx_std`, `rx_std`

The port descriptions use the following variables to represent parameters:

- <*n*>—The number of lanes
- <*p*>—The number of PLLs
- <*r*>—The number of CDR references clocks selected

Common Interface Ports

This section describes the common interface ports for the IP core.

Common interface consists of reset, clock signals, serial interface ports, control and status ports, parallel data ports, PMA ports and reconfig interface ports.

Figure 14-3: Common Interface Ports

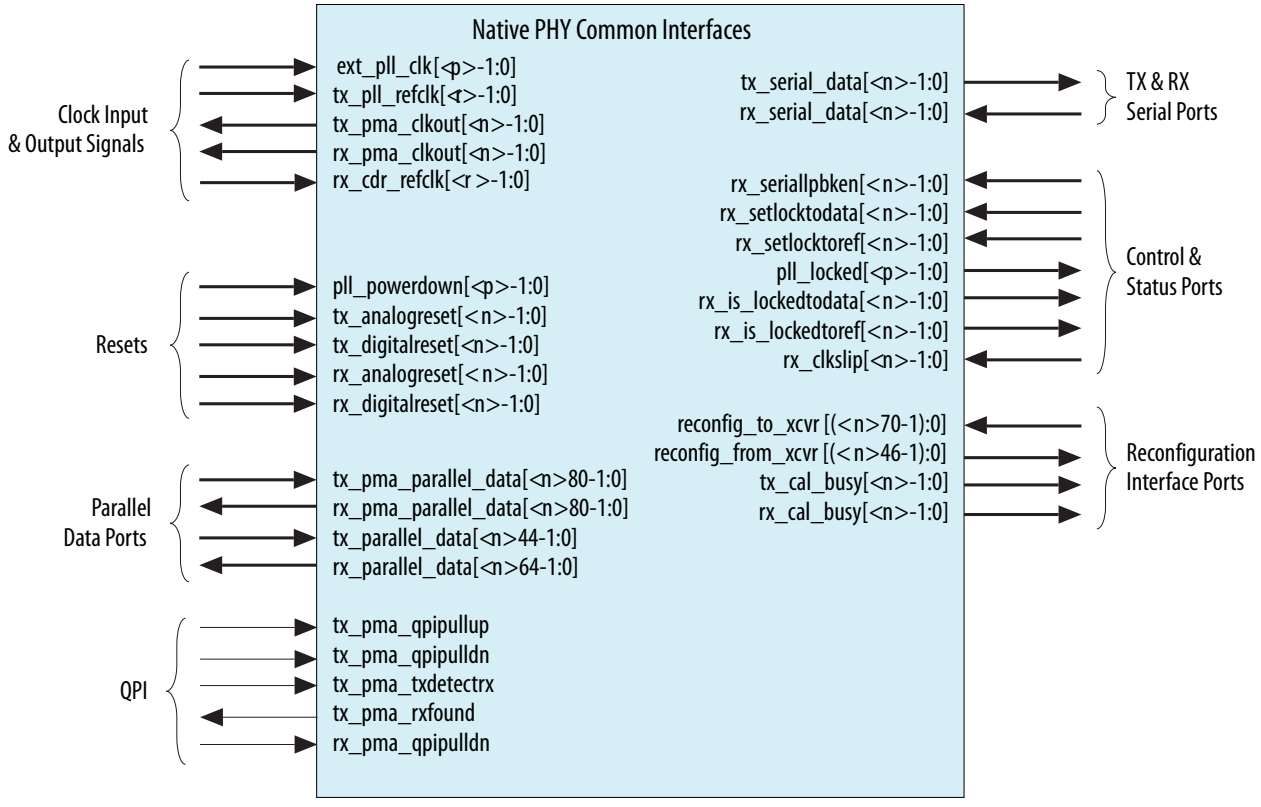


Table 14-19: Native PHY Common Interfaces

Name	Direction	Description
Clock Inputs and Output Signals		
`tx_pll_refclk[<r>-1:0]`	Input	The reference clock input to the TX PLL.
`tx_pma_clkout[<n>-1:0]`	Output	TX parallel clock output from PMA. This clock is only available in PMA direct mode.
`rx_pma_clkout[<n>-1:0]`	Output	RX parallel clock (recovered clock) output from PMA
`rx_cdr_refclk[<n>-1:0]`	Input	Input reference clock for the RX PFD circuit.
`ext_pll_clk[ <p> -1:0]`	Input	This optional signal is created when you select the **Use external TX PLL** option. If you instantiate a fractional PLL which is external to the Native PHY IP, then connect the output clock of this PLL to `ext_pll_clk`.
Resets		
`pll_powerdown[<p>-1:0]`	Input	When asserted, resets the TX PLL. Active high, edge sensitive reset signal. By default, the Arria V Native Transceiver PHY IP Core creates a separate `pll_powerdown` signal for each logical PLL. However, the Fitter may merge the PLLs if they are in the same transceiver bank. PLLs can only be merged if their `pll_powerdown` signals are driven from the same source. If the PLLs are in separate transceiver banks, you can choose to drive the `pll_powerdown` signals separately.

Name	Direction	Description
tx_analogreset[<n>-1:0]	Input	When asserted, resets for TX PMA, TX clock generation block, and serializer. Active high, edge sensitive reset signal. **Note:** For Arria V devices, while compiling a multi-channel transceiver design, you will see a compile warning (12020) in Intel Quartus Prime software related to the signal width of tx_analogreset. You can safely ignore this warning. Also, per-channel TX analog reset is not supported in Intel Quartus Prime software. Channel 0 TX analog resets all the transceiver channels.
tx_digitalreset[<n>-1:0]	Input	When asserted, resets the digital components of the TX datapath. Active high, edge sensitive, asynchronous reset signal. If your design includes bonded TX PCS channels, refer to Timing Constraints for Reset Signals when Using Bonded PCS Channels for a SDC constraint you must include in your design.
rx_analogreset[<n>-1:0]	Input	When asserted, resets the RX CDR, deserializer. Active high, edge sensitive, asynchronous reset signal.
rx_digitalreset[<n>-1:0]	Input	When asserted, resets the digital components of the RX datapath. Active high, edge sensitive, asynchronous reset signal.
Parallel data ports		
tx_pma_parallel_data[79:0]	Input	TX parallel data for the PMA Direct datapath. Driven directly from the FPGA fabric to the PMA. Not used when you enable the Standard PCS datapath.
rx_pma_parallel_data[79:0]	Output	RX PMA parallel data driven from the PMA to the FPGA fabric. Not used when you enable the Standard PCS datapath.

Name	Direction	Description
tx_parallel_data[43:0]	Input	PCS TX parallel data representing 4, 11-bit words. Used when you enable the Standard datapath. Refer to **Table 14-20**for bit definitions. Refer to **Table 14-21** various parameterizations.
rx_parallel_data[63:0]	Output	PCS RX parallel data, representing 4, 16-bit words. Used when you enable the Standard datapath. Refer to **Table 14-22** for bit definitions. Refer to **Table 14-23** for various parameterizations.
TX and RX serial ports		
tx_serial_data[<n>-1:0]	Output	TX differential serial output data.
rx_serial_data[<n>-1:0]	Input	RX differential serial output data.
Control and Status ports		
rx_seriallpbken[<n>-1:0]	Input	When asserted, the transceiver enters serial loopback mode. Loopback drives serial TX data to the RX interface.
rx_set_locktodata[<n>-1:0]	Input	When asserted, programs the RX CDR to manual lock to data mode in which you control the reset sequence using the rx_set_locktoref and rx_set_locktodata. Refer to "Transceiver Reset Sequence" in *Transceiver Reset Control in Arria V Devices* for more information about manual control of the reset sequence.
rx_set_locktoref[<n>-1:0]	Input	When asserted, programs the RX CDR to manual lock to reference mode in which you control the reset sequence using the rx_set_locktoref and rx_set_locktodata. Refer to Refer to "Transceiver Reset Sequence" in *Transceiver Reset Control in Arria V Devices* for more information about manual control of the reset sequence.
pll_locked[<p>-1:0]	Output	When asserted, indicates that the PLL is locked to the input reference clock.
rx_is_lockedtodata[<n>-1:0]	Output	When asserted, the CDR is locked to the incoming data.

Name	Direction	Description
rx_is_lockedtoref[<n>-1:0]	Output	When asserted, the CDR is locked to the incoming reference clock.
rx_clkslip[<n>-1:0]	Input	When you turn this signal on, the deserializer performs a clock slip operation to achieve word alignment. The clock slip operation alternates between skipping 1 serial bit and pausing the serial clock for 2 cycles to achieve word alignment. As a result, the period of the parallel clock can be extended by 2 unit intervals (UI) during the clock slip operation. This is an optional control input signal.
Reconfig Interface Ports		
reconfig_to_xcvr [(<n>70-1):0]	Input	Reconfiguration signals from the Transceiver Reconfiguration Controller. <n> grows linearly with the number of reconfiguration interfaces.
reconfig_from_xcvr [(<n>46-1):0]	Output	Reconfiguration signals to the Transceiver Reconfiguration Controller. <n> grows linearly with the number of reconfiguration interfaces.
tx_cal_busy[<n>-1:0]	Output	When asserted, indicates that the initial TX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP. You must hold the channel in reset until calibration completes.
rx_cal_busy[<n>-1:0]	Output	When asserted, indicates that the initial RX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP.

Table 14-20: Signal Definitions for tx_parallel_data with and without 8B/10B Encoding

The following table shows the signals within tx_parallel_data that correspond to data, control, and status signals.

TX Data Word	Description
Signal Definitions with 8B/10B Enabled	
tx_parallel_data[7:0]	TX data bus
tx_parallel_data[8]	TX data control character

TX Data Word	Description
tx_parallel_data[9]	Force disparity, validates disparity field.
tx_parallel_data[10]	Specifies the current disparity as follows: • 1'b0 = positive • 1'b1 = negative
Signal Definitions with 8B/10B Disabled	
tx_parallel_data[9:0]	TX data bus
tx_parallel_data[10]	Unused

Table 14-21: Location of Valid Data Words for tx_parallel_data for Various FPGA Fabric to PCS Parameterizations

The following table shows the valid 11-bit data words with and without the byte deserializer for single- and double-word FPGA fabric to PCS interface widths.

Configuration	Bus Used Bits
Single word data bus, byte deserializer disabled	[10:0] (word 0)
Single word data bus, byte serializer enabled	[32:22], [10:0] (words 0 and 2)
Double word data base, bye serializer disabled	[21:0] (words 0 and 1)
Double word data base, bye serializer disabled	[43:0] (words 0-3)

Table 14-22: Signal Definitions for rx_parallel_data with and without 8B/10B Encoding

This table shows the signals within rx_parallel_data that correspond to data, control, and status signals.

RX Data Word	Description
Signal Definitions with 8B/10B Enabled	
rx_parallel_data[7:0]	RX data bus
rx_parallel_data[8]	RX data control character
rx_parallel_data[9]	Error detect
rx_parallel_data[10]	Word alignment / synchronization status
rx_parallel_data[11]	Disparity error
rx_parallel_data[12]	Pattern detect
rx_parallel_data[14:13]	The following encodings are defined: • 2'b00: Normal data • 2'b01: Deletion • 2'b10: Insertion • 2'b11: Underflow
rx_parallel_data[15]	Running disparity value
Signal Definitions with 8B/10B Disabled	
rx_parallel_data[9:0]	RX data bus

RX Data Word	Description
rx_parallel_data[10]	Word alignment / synchronization status
rx_parallel_data[11]	Disparity error
rx_parallel_data[12]	Pattern detect
rx_parallel_data[14:13]	The following encodings are defined: • 2'b00: Normal data • 2'b01: Deletion • 2'b10: Insertion (or Underflow with 9'h1FE or 9'h1F7) • 2'b11: Overflow
rx_parallel_data[15]	Running disparity value

Table 14-23: Location of Valid Data Words for rx_parallel_data for Various FPGA Fabric to PCS Parameterizations

The following table shows the valid 16-bit data words with and without the byte deserializer for single- and double-word FPGA fabric to PCS interface widths.

Configuration	Bus Used Bits
Single word data bus, byte deserializer disabled	[15:0] (word 0)
Single word data bus, byte serializer enabled	[47:32], [15:0] (words 0 and 2)
Double word data base, bye serializer disabled	[31:0] (words 0 and 1)
Double word data base, bye serializer disabled	[63:0] (words 0-3)

Related Information

Transceiver Architecture in Arria V Devices

Standard PCS Interface Ports

This section describes the PCS interface.

Figure 14-4: Standard PCS Interfaces

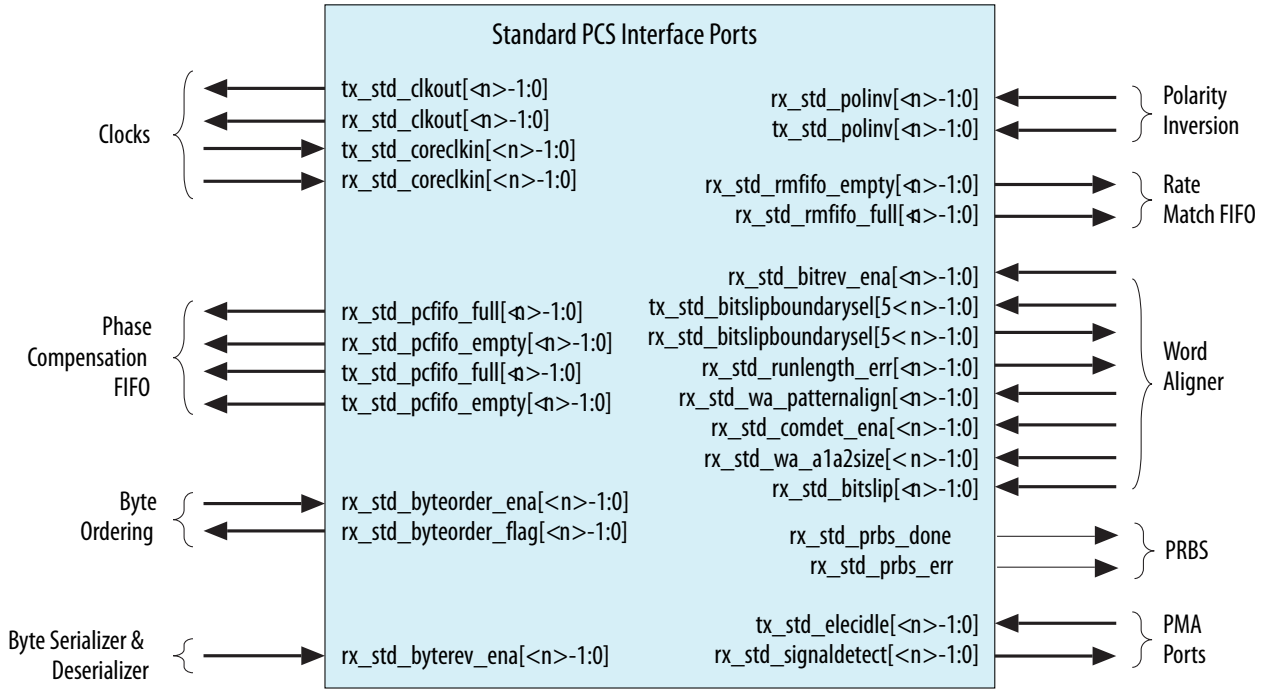


Table 14-24: Standard PCS Interface Ports

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
Clocks			
tx_std_clkout[<n>-1:0]	Output	—	TX Parallel clock output.
rx_std_clkout[<n>-1:0]	Output	—	RX parallel clock output. The CDR circuitry recovers RX parallel clock from the RX data stream.
tx_std_coreclkin[<n>-1:0]	Input	—	TX parallel clock input from the FPGA fabric that drives the write side of the TX phase compensation FIFO.
rx_std_coreclkin[<n>-1:0]	Input	—	RX parallel clock that drives the read side of the RX phase compensation FIFO.
Phase Compensation FIFO			
rx_std_pcfifo_full[<n>-1:0]	Output	Yes	RX phase compensation FIFO full status flag.

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
rx_std_pcfifo_empty[<n>-1:0]	Output	Yes	RX phase compensation FIFO status empty flag.
tx_std_pcfifo_full[<n>-1:0]	Output	Yes	TX phase compensation FIFO status full flag.
tx_std_pcfifo_empty[<n>-1:0]	Output	Yes	TX phase compensation FIFO status empty flag.
Byte Ordering			
rx_std_byteorder_ena[<n>-1:0]	Input	No	Byte ordering enable. When this signal is asserted, the byte ordering block initiates a byte ordering operation if the **Byte ordering control mode** is set to **manual**. Once byte ordering has occurred, you must deassert and reassert this signal to perform another byte ordering operation. This signal is an synchronous input signal; however, it must be asserted for at least 1 cycle of rx_std_clkout.
rx_std_byteorder_flag[<n>-1:0]	Output	Yes	Byte ordering status flag. When asserted, indicates that the byte ordering block has performed a byte order operation. This signal is asserted on the clock cycle in which byte ordering occurred. This signal is synchronous to the rx_std_clkout clock. You must a synchronizer this signal.
Byte Serializer and Deserializer			
rx_std_byterev_ena[<n>-1:0]	Input	No	This control signal is available in when the PMA width is 16 or 20 bits. When asserted, enables byte reversal on the RX interface.
8B/10B			

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
rx_std_polinv[<n>-1:0]	Input	No	Polarity inversion for the 8B/10B decoder, When set, the RX channels invert the polarity of the received data. You can use this signal to correct the polarity of differential pairs if the transmission circuitry or board layout mistakenly swapped the positive and negative signals. The polarity inversion function operates on the word aligner input.
tx_std_polinv[<n>-1:0]	Input	No	Polarity inversion, part of 8B10B encoder, When set, the TX interface inverts the polarity of the TX data.
Rate Match FIFO			
rx_std_rmfifo_empty[<n>-1:0]	Output	No	Rate match FIFO empty flag. When asserted, the rate match FIFO is empty. This port is only used for XAUI, GigE, and Serial RapidIO in double width mode. In double width mode, the FPGA data width is twice the PCS data width to allow the fabric to run at half the PCS frequency
rx_std_rmfifo_full[<n>-1:0]	Output	No	Rate match FIFO full flag. When asserted the rate match FIFO is full. You must synchronize this signal. This port is only used for XAUI, GigE, and Serial RapidIO in double width mode.
Word Aligner			
rx_std_bitrev_ena[<n>-1:0]	Input	No	When asserted, enables bit reversal on the RX interface. Bit order may be reversed if external transmission circuitry transmits the most significant bit first. When enabled, the receive circuitry receives all words in the reverse order. The bit reversal circuitry operates on the output of the word aligner.
tx_std_bitslipboundarysel[5<n>-1:0]	Input	No	BitSlip boundary selection signal. Specifies the number of bits that the TX bit slipper must slip.

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
rx_std_bitslipboundarysel[5<n>-1:0]	Output	No	This signal operates when the word aligner is in bitslip word alignment mode. It reports the number of bits that the RX block slipped to achieve deterministic latency.
rx_std_runlength_err[<n>-1:0]	Output	No	When asserted, indicates a run length violation. Asserted if the number of consecutive 1s or 0s exceeds the number specified in the parameter editor GUI.
rx_st_wa_patternalign	Input	No	Active when you place the word aligner in manual mode. In manual mode, you align words by asserting rx_st_wa_patternalign. rx_st_wa_patternalign is edge sensitive. For more information refer to the *Word Aligner* section in the *Transceiver Architecture in Arria V Devices*.
rx_std_wa_a1a2size[<n>-1:0]	Input	No	Used for the SONET protocol. Assert when the A1 and A2 framing bytes must be detected. A1 and A2 are SONET backplane bytes and are only used when the PMA data width is 8 bits.
rx_std_bitslip[<n>-1:0]	Input	No	Used when word aligner mode is bitslip mode. For every rising edge of the rx_std_bitslip signal, the word boundary is shifted by 1 bit. Each bitslip removes the earliest received bit from the received data. This is an asynchronous input signal and inside there is a synchronizer to synchronize it with rx_pma_clk/rx_clkout.
PRBS			
rx_std_prbs_done	Output	Yes	When asserted, indicates the verifier has aligned and captured consecutive PRBS patterns and the first pass through a polynomial is complete. The generator has restarted the polynomial.

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
rx_std_prbs_err	Output	Yes	When asserted, indicates an error only after the rx_std_prbs_done signal has been asserted. This signal pulses for every error that occurs. Errors can only occur once per word. To clear the PRBS pattern and deassert the rx_std_prbs_done signal by writing to the memory-mapped register PRBS Error Clear that you access through the Transceiver Reconfiguration Controller IP Core.
Miscellaneous			
tx_std_elecidle[<n>-1:0]	Input		When asserted, enables a circuit to detect a downstream receiver. This signal must be driven low when not in use because it causes the TX PMA to enter electrical idle mode with the TX serial data signals in tristate mode.
rx_std_signaldetect[<n>-1:0]	Output	No	Signal threshold detect indicator. When asserted, it indicates that the signal present at the receiver input buffer is above the programmed signal detection threshold value. You must synchronize this signal.

Related Information

Transceiver Architecture in Arria V Devices

SDC Timing Constraints

This section describes SDC timing constraints.

The Intel Quartus Prime software reports timing violations for asynchronous inputs to the Standard PCS and 10G PCS. Because many violations are for asynchronous paths, they do not represent actual timing failures. You may choose one of the following three approaches to identify these false timing paths to the Intel Quartus Prime or TimeQuest software.

- You can cut these paths in your Synopsys Design Constraints (**.sdc**) file by using the set_false_path command as shown in the following example.

Example 14-1: Using the set_false_path Constraint to Identify Asynchronous Inputs

```
set_false_path -through {*8gbitslip*} -to  [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*8gbytordpld*} -to [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*8gcmpfifoburst*} -to [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*8gphfifoburstrx*} -to [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]

set_false_path -through {*8gsyncsmen*} -to [get_registers
*8g*pcs*SYNC_DATA_REG*]
set_false_path -through {*8gwrdisablerx*} -to [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*rxpolarity*} -to [get_registers *SYNC_DATA_REG*]
set_false_path -through {*pldeidleinfersel*} -to [get_registers
*SYNC_DATA_REG*]
```

- You can use the `set_max_delay` constraint on a given path to create a constraint for asynchronous signals that do not have a specific clock relationship but require a maximum path delay.

Example 14-2: Using the max_delay Constraint to Identify Asynchronous Inputs

```
# Example: Apply 10ns max delay
set_max_delay -from *tx_from_fifo* -to *8g*pcs*SYNC_DATA_REG1 10
```

- You can use the `set_false` path command only during Timequest timing analysis.

Example 14-3: Using the set_false TimeQuest Constraint to Identify Asynchronous Inputs

```
#if {$::TimeQuestInfo(nameofexecutable) eq "quartus_fit"} {
#} else {
#set_false_path -from [get_registers {*tx_from_fifo*}] -through
{*txbursten*} -to [get_registers *8g_*_pcs*SYNC_DATA_REG
```

Note: In in all of these examples, you must substitute you actual signal names for the signal names shown.

Dynamic Reconfiguration

Dynamic reconfiguration calibrates each channel to compensate for variations due to process, voltage, and temperature (PVT).

As silicon progresses towards smaller process nodes, circuit performance is affected more by variations due to process, voltage, and temperature (PVT). These process variations result in analog voltages that can be offset from required ranges. The calibration performed by the dynamic reconfiguration interface compensates for variations due to PVT.

For nonbonded clocks, each channel and each TX PLL has a separate dynamic reconfiguration interfaces. The MegaWizard Plug-In Manager provides informational messages on the connectivity of these

interfaces. The following example shows the messages for the Arria V Native PHY with four duplex channels, four TX PLLs, in a nonbonded configuration.

For more information about transceiver reconfiguration refer to Transceiver Reconfiguration Controller IP Core.

Example 14-4: Informational Messages for the Transceiver Reconfiguration Interface

```
PHY IP will require 8 reconfiguration interfaces for connection to the
external reconfiguration controller.
Reconfiguration interface offsets 0-3 are connected to the transceiver
channels.
Reconfiguration interface offsets 4–7 are connected to the transmit PLLs.
```

Related Information

Transceiver Architecture in Arria V Devices

Simulation Support

The Intel Quartus Prime release provides simulation and compilation support for the Arria V Native PHY IP Core. Refer to Running a Simulation Testbench for a description of the directories and files that the Intel Quartus Prime software creates automatically when you generate your Arria V Transceiver Native PHY IP Core.

Slew Rate Settings

The following transceiver slew rate settings are allowed in Intel Quartus Prime software.

Table 14-25: Slew Rate Settings for Arria V GX/SX/GT/ST devices

Protocol / Datarate	Allowed Intel Quartus Prime Settings	IBIS-AMI Settings
10GBASE-R, CEI 6G	5	Protocol - (2) CEI
PCI Express Gen 2	4	Protocol - (1) PCIe
PCI Express Gen1, XAUI	3	Protocol - (4) XAUI
Gigabit Ethernet	2	Protocol - (3) Gigabit Ethernet
<1 Gbps	2	Protocol - (0) Basic, Slew - (0)
1 Gbps - 3 Gbps	2, 3	Protocol - (0) Basic, Slew - (0 or 1)
3 Gbps - 5 Gbps	3, 4	Protocol - (0) Basic, Slew - (1 or 2)
>5 Gbps	5	Protocol - (0) Basic, Slew - (2)

Note: For protocols not mentioned in the above table, the Intel Quartus Prime and IBIS-AMI settings will be in accordance to the specified data rates. For example, for SATA 3Gbps; allowed Intel Quartus Prime and IBIS-AMI settings are 3 and Protocol - (4) XAUI respectively.

Arria V GZ Transceiver Native PHY IP Core 15

2020.06.02

UG-01080 Subscribe Send Feedback

Unlike other PHY IP Cores, the Native PHY IP Core does not include an Avalon Memory-Mapped (Avalon-MM) interface. Instead, it exposes all signals directly as ports. The Arria V GZ Transceiver Native PHY IP Core provides the following three datapaths:

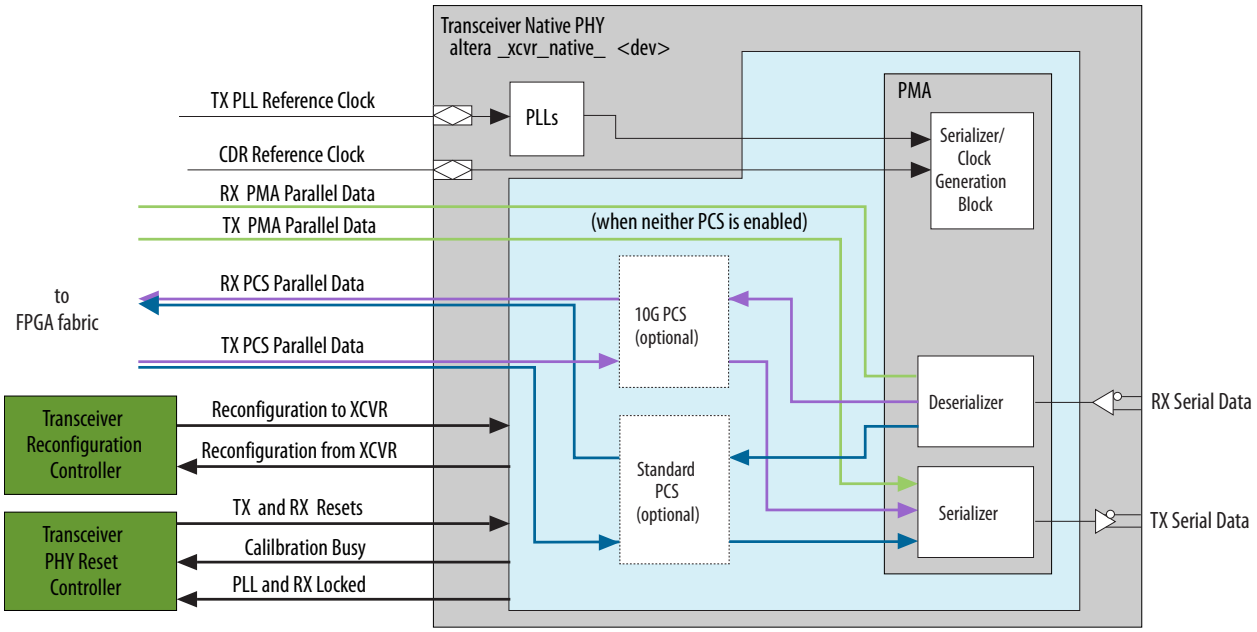
- Standard PCS
- 10G PCS
- PMA Direct

You can enable the Standard PCS, the 10G PCS, or both if your design uses the Transceiver Reconfiguration Controller to change dynamically between the two PCS datapaths. The transceiver PHY does not include an embedded reset controller. You can either design custom reset logic or incorporate Altera's "Transceiver PHY Reset Controller IP Core" to implement reset functionality.

In PMA Direct mode, the Native PHY provides direct access to the PMA from the FPGA fabric; consequently, the latency for transmitted and received data is very low. However, you must implement any PCS function that your design requires in the FPGA fabric.

The following figure illustrates the use of the Arria V GZ Transceiver Native PHY IP Core. As this figure illustrates, TX PLL and clock data recovery (CDR) reference clocks from the pins of the device are input to the PLL module and CDR logic. When enabled, the 10G or Standard PCS drives TX parallel data and receives RX parallel data. When neither PCS is enabled the Native PHY operates in PMA Direct mode.

Figure 15-1: Arria V GZ Native Transceiver PHY IP Core



In a typical design, the separately instantiated Transceiver PHY Reset Controller drives reset signals to Native PHY and receives calibration and locked status signal from the Native PHY. The Native PHY reconfiguration buses connect the external Transceiver Reconfiguration Controller for calibration and dynamic reconfiguration of the PLLs.

You specify the initial configuration when you parameterize the IP core. The Transceiver Native PHY IP Core connects to the "Transceiver Reconfiguration Controller IP Core" to dynamically change reference clocks and PLL connectivity at runtime.

Device Family Support for Arria V GZ Native PHY

This section describes the device family support available in the Arria V GZ native PHY.

IP cores provide either final or preliminary support for target Altera device families. These terms have the following definitions:

- Final support—Verified with final timing models for this device.
- Preliminary support—Verified with preliminary timing models for this device.

Table 15-1: Device Family Support

This tables lists the level of support offered by the Arria V GZ Transceiver Native PHY IP Core for Altera device families.

Device Family	Support
Arria V GZ devices	Final
Other device families	No support

Performance and Resource Utilization for Arria V GZ Native PHY

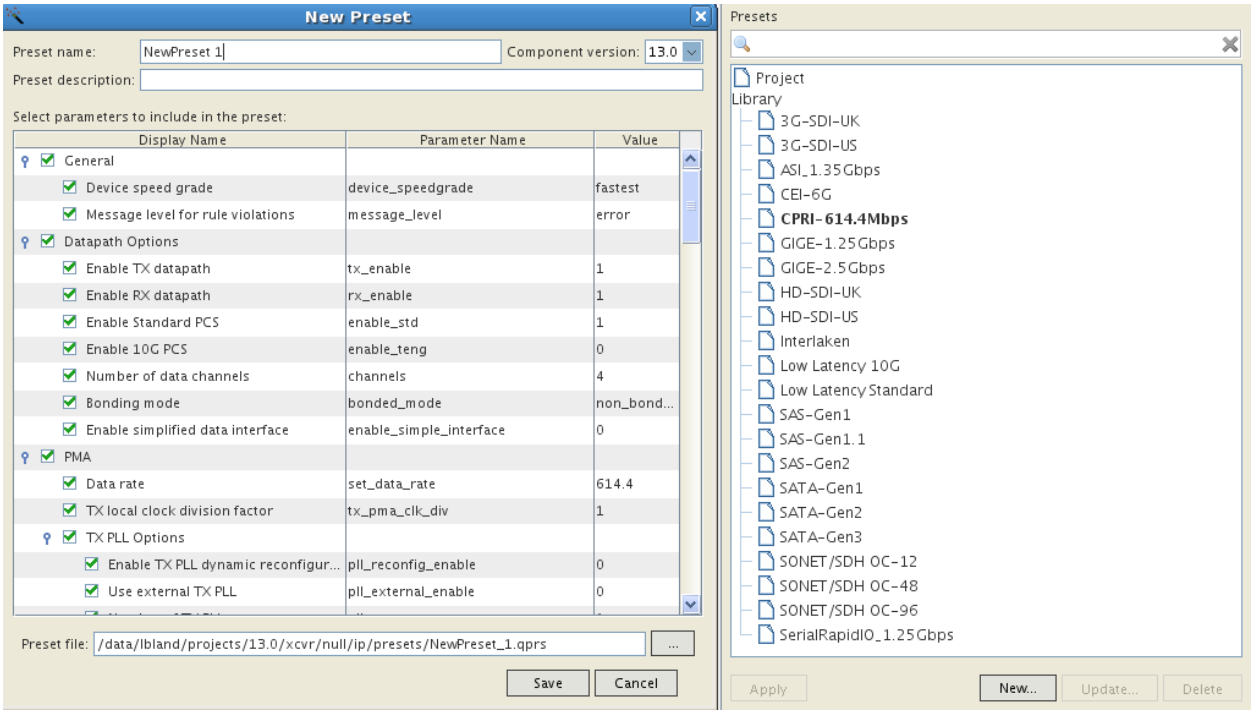
Because the 10G PCS, Standard PCS, and PMA are implemented in hard logic, the Arria V GZ Native PHY IP Core uses less than 1% of the available ALMs, memory, primary and secondary logic registers.

Parameter Presets

Presets allow you to specify a group of parameters to implement a particular protocol or application.

If you apply a preset, the parameters with specific required values are set for you. When applied, the preset is in boldface and remains as such unless you change some of the preset parameters. Selecting a preset does not prevent you from changing any parameter to meet the requirements of your design. The following figure illustrates the Preset panel and form to create custom presets.

Figure 15-2: Preset Panel and Form To Create Custom Presets



Parameterizing the Arria V GZ Native PHY

This section provides a list of instructions on how to configure the Arria V GZ native PHY IP core.

Complete the following steps to configure the Arria V GZ Native PHY IP Core:

1. Under **Tools** > **IP Catalog**, select **Arria V GZ** as the device family.
2. Under **Tools** > **IP Catalog** > **Interface Protocols** > **Transceiver PHY**, select **Arria V GZ Native PHY**.
3. Use the tabs on the MegaWizard Plug-In Manager to select the options required for the protocol.
4. Click **Finish**.

Clicking **Finish** generates your customized Arria V GZ Native PHY IP Core.

General Parameters for Arria V GZ Native PHY

This section describes the datapath parameters in the General Options tab for the Arria V GZ native PHY.

Table 15-2: General and Datapath Options

The following table lists the parameters available on the General Options tab. Note that you can enable the Standard PCS, the 10G PCS, or both if you intend to reconfigure between the two available PCS datapaths.

Name	Range	Description
Device speed grade	**fastest - 3_H3**	Specifies the speed grade.
Message level for rule violations	**error** **warning**	When you select the **error** message level, the Intel Quartus Prime rules checker reports an error if you specify incompatible parameters. If you select the **warning** message level, the Intel Quartus Prime rules checker reports a warning instead of an error.
Datapath Options		
Enable TX datapath	**On/Off**	When you turn this option **On**, the core includes the TX datapath.
Enable RX datapath	**On/Off**	When you turn this option **On**, the core includes the RX datapath.
Enable Standard PCS	**On/Off**	When you turn this option **On**, the core includes the Standard PCS . You can enable both the Standard and 10G PCS if you plan to dynamically reconfigure the Native PHY.
Enable 10G PCS	**On/Off**	When you turn this option **On**, the core includes the 10G PCS. You can enable both the **Standard** and **10G PCS** if you plan to dynamically reconfigure the Native PHY.
Number of data channels	Device Dependent	Specifies the total number of data channels in each direction. From 1-32 channels are supported.

Name	Range	Description
Bonding mode	**Non-bonded or x1** **Bonded or ×6/xN** **fb_compensation**	In **Non-bonded or x1** mode, each channel is paired with a PLL. If one PLL drives multiple channels, PLL merging is required. During compilation, the Intel Quartus Prime Fitter, merges all the PLLs that meet PLL merging requirements. Refer to **Merging TX PLLs In Multiple Transceiver PHY Instances** on page 17-58 to observe PLL merging rules. Select **×6** to use the same clock source for up to 6 channels in a single transceiver bank, resulting in reduced clock skew. You must use contiguous channels when you select **×6** bonding. In addition, you must place logical channel 0 in either physical channel 1 or 4. Physical channels 1 and 4 are indirect drivers of the ×6 clock network. Select **fb_compensation** (feedback compensation) to use the same clock source for multiple channels across different transceiver banks to reduce clock skew. For more information about bonding, refer to "Bonded Channel Configurations Using the PLL Feedback Compensation Path" in Transceiver Clocking in Arria V devices chapter of the *Arria V Device Handbook*.
Enable simplified data interface	**On/Off**	When you turn this option **On**, the Native PHY presents only the relevant data bits. When you turn this option **Off**, the Native PHY presents the full raw interface to the fabric. If you plan to dynamically reconfigure the Native PHY, you must turn this option **Off** and you need to understand the mapping of data to the FPGA fabric. Refer to **Table 15-10** for more information. When you turn this option **On** , the Native PHY presents an interface that includes only the data necessary for the single configuration specified.

Related Information

Transceiver Clocking in Arria V Devices

PMA Parameters for Arria V GZ Native PHY

This section describes the PMA parameters for the Arria V GZ native PHY.

Table 15-3: PMA Options

The following table describes the options available for the PMA. For more information about the PMA, refer to the *PMA Architecture* section in the *Transceiver Architecture in Arria V GZ Devices*.

Some parameters have ranges where the value is specified as Device Dependent. For such parameters, the possible range of frequencies and bandwidths depends on the device, speed grade, and other design characteristics. Refer to the *Arria V GZ Device Datasheet* for specific data for Arria V GZ devices.

Parameter	Range	Description
Data rate	Device Dependent	Specifies the data rate.
TX local clock division factor	**1**, **2**, **4**, **8**	Specifies the value of the divider available in the transceiver channels to divide the input clock to generate the correct frequencies for the parallel and serial clocks.
TX PLL base data rate	Device Dependent	Specifies the base data rate for the clock input to the TX PLL. Select a **base data rate** that minimizes the number of PLLs required to generate all the clocks required for data transmission. By selecting an appropriate **base data rate**, you can change data rates by changing the divider used by the clock generation block.
PLL base data rate	Device Dependent	Shows the base data rate of the clock input to the TX PLL. The **PLL base data rate** is computed from the TX local clock division factor multiplied by the **data rate**. Select a **PLL base data rate** that minimizes the number of PLLs required to generate all the clocks for data transmission. By selecting an appropriate **PLL base data rate**, you can change data rates by changing the **TX local clock division factor** used by the clock generation block.

TX PMA Parameters

Table 15-4: TX PMA Parameters

The following table describes the TX PMA options you can specify.

For more information about the TX CMU, ATX, and fractional PLLs, refer to the *Arria V GZ PLLs* section in *Transceiver Architecture in Arria V GZ Devices*.

Parameter	Range	Description
Enable TX PLL dynamic reconfiguration	**On/Off**	When you turn this option **On**, you can dynamically reconfigure the PLL to use a different reference clock input. This option is also required to simulate TX PLL reconfiguration. If you turn this option **On**, the Intel Quartus Prime Fitter prevents PLL merging by default; however, you can specify merging using the `XCVR_TX_PLL_RECONFIG_GROUP` QSF assignment.
Use external TX PLL	**On/Off**	When you turn this option **On**, the Native PHY does not automatically instantiate a TX PLL. Instead, you must instantiate an external PLL and connect it to the `ext_pll_clk[<p> -1 : 0]` port of the Arria V GZ Native PHY. Use the Arria V GZ Transceiver PLL IP Core to instantiate a CMU or ATX PLL. Use Altera Phase-Locked Loop (ALTERA_ PLL) Megafunction to instantiate a fractional PLL.
Number of TX PLLs	**1-4**	Specifies the number of TX PLLs that can be used to dynamically reconfigure channels to run at multiple data rates. If your design does not require transceiver TX PLL dynamic reconfiguration, set this value to 1. The number of actual physical PLLs that are implemented depends on the selected clock network. Each channel can dynamically select between n PLLs, where n is the number of PLLs specified for this parameter. **Note:** Refer to *Transceiver Clocking in Arria V Devices* chapter for more details.
Main TX PLL logical index	**0-3**	Specifies the index of the TX PLL used in the initial configuration.
Number of TX PLL reference clocks	**1-5**	Specifies the total number of reference clocks that are shared by all of the PLLs.

TX PLL<n>

Table 15-5: TX PLL Parameters

The following table describes how you can define multiple TX PLLs for your Native PHY. The Native PHY GUI provides a separate tab for each TX PLL.

Parameter	Range	Description
PLL type	**CMU** **ATX**	You can select either the **CMU** or **ATX** PLL. The **CMU** PLL has a larger frequency range than the **ATX** PLL. The **ATX** PLL is designed to improve jitter performance and achieves lower channel-to-channel skew; however, it supports a narrower range of data rates and reference clock frequencies. Another advantage of the **ATX** PLL is that it does not use a transceiver channel, while the **CMU** PLL does. Because the **CMU** PLL is more versatile, it is specified as the default setting. An error message displays in the message pane if the settings chosen for **Data rate** and **Input clock frequency** are not supported for selected PLL.
PLL base data rate	Device Dependent	Shows the base data rate of the clock input to the TX PLL.The **PLL base data rate** is computed from the **TX local clock division factor** multiplied by the **Data rate**. Select a **PLL base data rate** that minimizes the number of PLLs required to generate all the clocks for data transmission. By selecting an appropriate **PLL base data rate**, you can change data rates by changing the **TX local clock division factor** used by the clock generation block.
Reference clock frequency	Device Dependent	Specifies the frequency of the reference clock for the **Selected reference clock source** index you specify. You can define a single frequency for each PLL. You can use the Transceiver Reconfiguration Controller shown in Arria V GZ Native Transceiver PHY IP Core to dynamically change the reference clock input to the PLL. Note that the list of frequencies updates dynamically when you change the **Data rate**. The Input clock frequency drop down menu is populated with all valid frequencies derived as a function of the data rate and base data rate. However, if **fb_compensation** is selected as the bonding mode then the input reference clock frequency is limited to the data rate divided by the PCS-PMA interface width.
Selected reference clock source	**0-4**	You can define up to 5 frequencies for the PLLs in your core. The **Reference clock frequency** selected for index 0 , is assigned to TX PLL<0>. The **Reference clock frequency** selected for index 1 , is assigned to TX PLL<1>, and so on.

RX CDR Options

Table 15-6: RX PMA Parameters

The following table describes the RX CDR options you can specify. For more information about the CDR circuitry, refer to the *Receiver Clock Data Recovery Unit* section in *Clock Networks and PLLs in Arria V Devices*.

Parameter	Range	Description
Enable CDR dynamic reconfiguration	**On/Off**	When you turn this option **On**, you can dynamically change the reference clock input the CDR circuit. This option is also required to simulate TX PLL reconfiguration.
Number of CDR reference clocks	**1-5**	Specifies the number of reference clocks for the CDRs.
Selected CDR reference clock	**0-4**	Specifies the index of the selected CDR reference clock.
Selected CDR reference clock frequency	Device Dependent	Specifies the frequency of the clock input to the CDR.
PPM detector threshold	**+/- 1000 PPM**	Specifies the maximum PPM difference the CDR can tolerate between the input reference clock and the recovered clock.
Enable rx_pma_clkout port	**On/Off**	When you turn this option **On**, the RX parallel clock which is recovered from the serial received data is an output of the PMA.
Enable rx_is_lockedtodata port	**On/Off**	When you turn this option **On**, the `rx_is_lockedtodata` port is an output of the PMA.
Enable rx_is_lockedtoref port	**On/Off**	When you turn this option **On**, the `rx_is_lockedtoref` port is an output of the PMA.
Enable rx_set_lockedtodata and rx_set_locktoref ports	**On/Off**	When you turn this option **On**, the `rx_set_lockedtodata` and rx_set_lockedtoref ports are outputs of the PMA.
Enable rx_clkslip port	**On/Off**	When you turn this option **On**, the `rx_clkslip` control input port is enabled. The deserializer slips one clock edge each time this signal is asserted. You can use this feature to minimize uncertainty in the serialization process as required by protocols that require a datapath with deterministic latency such as CPRI.
Enable rx_seriallpbken port	**On/Off**	When you turn this option **On**, the `rx_seriallpbken` is an input to the core. When your drive a 1 on this input port, the PMA operates in loopback mode with TX data looped back to the RX channel.

PMA Optional Ports

Table 15-7: RX PMA Parameters

The following table describes the optional ports you can include in your IP Core. The QPI are available to implement the Intel Quickpath Interconnect.

For more information about the CDR circuitry, refer to the *Receiver Clock Data Recovery Unit* section in *Clock Networks and PLLs in Arria V GZ Devices*.

Parameter	Range	Description
Enable tx_pma_qpipullup port (QPI)	**On/Off**	When you turn this option **On**, the core includes `tx_pma_qpipullup` control input port. This port is only used for QPI applications.
Enable tx_pma_qpipulldn port (QPI)	**On/Off**	When you turn this option **On**, the core includes `tx_pma_qpipulldn` control input port. This port is only used for QPI applications.
Enable tx_pma_txdetectrx port (QPI)	**On/Off**	When you turn this option **On**, the core includes `tx_pma_txdetectrx` control input port. This port is only used for QPI applications. The RX detect block in the TX PMA detects the presence of a receiver at the other end of the channel. After receiving a `tx_pma_txdetectrx` request, the receiver detect block initiates the detection process.
Enable tx_pma_rxfound port (QPI)	**On/Off**	When you turn this option **On**, the core includes `tx_pma_rxfound` output status port. This port is only used for QPI applications. The RX detect block in the TX PMA detects the presence of a receiver at the other end of the channel. `tx_pma_rxfound` indicates the result of detection.
Enable rx_pma_qpipulldn port (QPI)	**On/Off**	When you turn this option **On**, the core includes the rx_pma_qpipulldn port. This port is only used for QPI applications.
Enable rx_pma_clkout port	**On/Off**	When you turn this option **On**, the RX parallel clock which is recovered from the serial received data is an output of the PMA.
Enable rx_is_lockedtodata port	**On/Off**	When you turn this option **On**, the `rx_is_lockedtodata` port is an output of the PMA.
Enable rx_is_lockedtoref port	**On/Off**	When you turn this option **On**, the `rx_is_lockedtoref` port is an output of the PMA.
Enable rx_set_lockedtodata and **rx_set_locktoref ports**	**On/Off**	When you turn this option **On**, the `rx_set_lockedtodata` and rx_set_lockedtoref ports are outputs of the PMA.

Parameter	Range	Description
Enable rx_pma_bitslip_port	**On/Off**	When you turn this option **On**, the `rx_pma_bitslip` is an input to the core. The deserializer slips one clock edge each time this signal is asserted. You can use this feature to minimize uncertainty in the serialization process as required by protocols that require a datapath with deterministic latency such as CPRI.
Enable rx_seriallpbken port	**On/Off**	When you turn this option **On**, the `rx_seriallpbken` is an input to the core. When your drive a 1 on this input port, the PMA operates in loopback mode with TX data looped back to the RX channel.

The following table lists the best case latency for the most significant bit of a word for the RX deserializer for the PMA Direct datapath. For example, for an 8-bit interface width, the latencies in UI are 11 for bit 7, 12 for bit 6, 13 for bit 5, and so on.

Table 15-8: Latency for RX Deserialization in Arria V GZ Devices

FPGA Fabric Interface Width	Arria V GZ Latency in UI
8 bits	11
10 bits	13
16 bits	19
20 bits	23
32 bits	35
40 bits	43
64 bits	99
80 bits	123

Table 15-9: Latency for TX Serialization in Arria V GZ Devices

The following table lists the best- case latency for the LSB of the TX serializer for all supported interface widths for the PMA Direct datapath.

FPGA Fabric Interface Width	Arria V GZ Latency in UI
8 bits	44
10 bits	54
16 bits	68
20 bits	84
32 bits	100
40 bits	124
64 bits	132
80 bits	164

The following tables lists the bits used for all FPGA fabric to PMA interface widths. Regardless of the FPGA Fabric Interface Width selected, all 80 bits are exposed for the TX and RX parallel data ports. However, depending upon the interface width selected not all bits on the bus will be active. The following table lists which bits are active for each FPGA Fabric Interface Width selection. For example, if your interface is 16 bits, the active bits on the bus are [17:0] and [7:0] of the 80 bit bus. The non-active bits are tied to ground.

Table 15-10: Active Bits for Each Fabric Interface Width

FPGA Fabric Interface Width	Bus Bits Used
8 bits	[7:0]
10 bits	[9:0]
16 bits	{[17:10], [7:0]}
20 bits	[19:0]
32 bits	{[37:30], [27:20], [17:10], [7:0]}
40 bits	[39:0]
64 bits	{[77:70], [67:60], [57:50], [47:40], [37:30], [27:20], [17:10], [7:0]}
80 bits	[79:0]

Related Information

- **Arria V Device Handbook Volume 2: Transceivers**
- **Device Datasheet for Arria V Devices**

Standard PCS Parameters for the Native PHY

This section shows the complete datapath and clocking for the Standard PCS and defines the parameters available in the GUI to enable or disable the individual blocks in the Standard PCS.

Figure 15-3: The Standard PCS Datapath

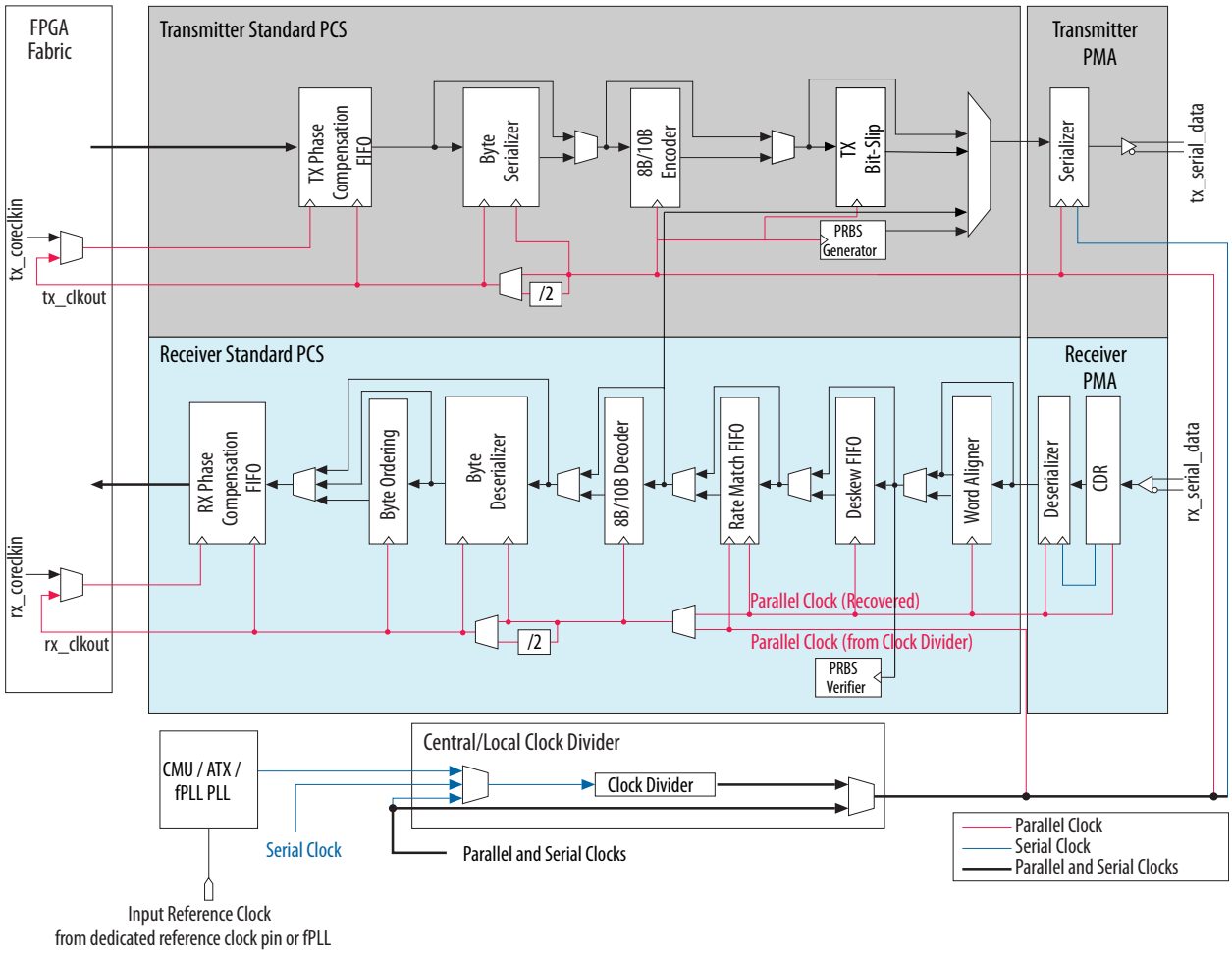


Table 15-11: General and Datapath Parameters

The following table describes the general and datapath options for the Standard PCS.

Parameter	Range	Description
Standard PCS protocol mode	**basic** **cpri** **gige** **srio_2p1**	Specifies the protocol that you intend to implement with the Native PHY. The protocol mode selected guides the MegaWizard in identifying legal settings for the Standard PCS datapath. Use the following guidelines to select a protocol mode: • **basic** -select this mode for when none of the other options are appropriate. You should also select this mode to enable diagnostics, such as loopback. • **cpri** select this mode if you intend to implement CPRI or another protocol that requires deterministic latency. Altera recommends that you select the appropriate CPRI preset for the CPRI protocol. • **gige** -select this mode if you intend to implement Gigabit Ethernet. Altera recommends that you select the appropriate GIGE preset for the Ethernet bandwidth you intend to implement. • **srio_2p1** -select this mode if you intend to implement the Serial RapidIO protocol.
Standard PCS/PMA interface width	**8, 10, 16, 20, 32, 40 64, 80**	Specifies the width of the datapath that connects the FPGA fabric to the PMA. The transceiver interface width depends upon whether you enable 8B/10B. To simplify connectivity between the FPGA fabric and PMA, the bus bits used are not contiguous for 16- and 32-bit buses. 16-, 32-, and 64-bit buses. Refer to **Table 15-10** for the bits used.
FPGA fabric/Standard TX PCS interface width	**8, 10, 16, 20**	Shows the FPGA fabric to TX PCS interface width which is calculated from the Standard PCS/PMA interface width.
FPGA fabric/Standard RX PCS interface width	**8, 10, 16, 20**	Shows the FPGA fabric to RX PCS interface width which is calculated from the Standard PCS/PMA interface width.
Enable Standard PCS low latency mode	**On/ Off**	When you turn this option **On**, all PCS functions are disabled. This option creates a the lowest latency Native PHY that allows dynamic reconfigure between multiple PCS datapaths.

Phase Compensation FIFO

The phase compensation FIFO assures clean data transfer to and from the FPGA fabric by compensating for the clock phase difference between the low-speed parallel clock and FPGA fabric interface clock. The following table describes the options for the phase compensation FIFO.

Table 15-12: Phase Compensation FIFO Parameters

Parameter	Range	Description
TX FIFO mode	**low_latency** **register_fifo**	The following 2 modes are possible: • **low_latency** : This mode adds 3-4 cycles of latency to the TX datapath. • **register_fifo** : In this mode the FIFO is replaced by registers to reduce the latency through the PCS. Use this mode for protocols that require deterministic latency, such as CPRI.
RX FIFO mode	**low_latency** **register_fifo**	The following 2 modes are possible: • **low_latency** : This mode adds 2-3 cycles of latency to the RX datapath. • **register_fifo** : In this mode the FIFO is replaced by registers to reduce the latency through the PCS. Use this mode for protocols that require deterministic latency, such as CPRI.
Enable tx_std_pcfifo_full port	**On/Off**	When you turn this option **On**, the TX Phase compensation FIFO outputs a FIFO full status flag.
Enable tx_std_pcfifo_empty port	**On/Off**	When you turn this option **On**, the TX Phase compensation FIFO outputs a FIFO empty status flag.
Enable rx_std_pcfifo_full port	**On/Off**	When you turn this option **On**, the RX Phase compensation FIFO outputs a FIFO full status flag.
Enable rx_std_pcfifo_empty port	**On/ Off**	When you turn this option **On**, the RX Phase compensation FIFO outputs a FIFO empty status flag.

Byte Ordering Block Parameters

The RX byte ordering block realigns the data coming from the byte deserializer. This block is necessary when the PCS to FPGA fabric interface width is greater than the PCS datapath. Because the timing of the RX PCS reset logic is indeterminate, the byte ordering at the output of the byte deserializer may or may not match the original byte ordering of the transmitted data. The following table describes the byte ordering block parameters.

Parameter	Range	Description
Enable RX byte ordering	**On/Off**	When you turn this option **On**, the PCS includes the byte ordering block.

Parameter	Range	Description
Byte ordering control mode	**Manual** **Auto**	Specifies the control mode for the byte ordering block. The following modes are available: • **Manual** : Allows you to control the byte ordering block • **Auto** : The word aligner automatically controls the byte ordering block once word alignment is achieved.

Byte ordering pattern width	**8-10**	Shows width of the pad that you must specify. This width depends upon the PCS width and whether nor not 8B/10B encoding is used as follows:
		`Width 8B/10B Pad Pattern`
		`8/16,32 No 8 bits`
		`10,20,40 No 10 bits`
		`8,16,32 Yes 9 bits`

Byte ordering symbol count	**1-2**	Specifies the number of symbols the word aligner should search for. When the PMA is 16 or 20 bits wide, the byte ordering block can optionally search for 1 or 2 symbols.
Byte order pattern (hex)	User-specified 8-10 bit pattern	Specifies the search pattern for the byte ordering block.
Byte order pad value (hex)	User-specified 8-10 bit pattern	Specifies the pad pattern that is inserted by the byte ordering block. This value is inserted when the byte order pattern is recognized. The byte ordering pattern should occupy the least significant byte (LSB) of the parallel TX data. If the byte ordering block identifies the programmed byte ordering pattern in the most significant byte (MSB) of the byte-deserialized data, it inserts the appropriate number of user-specified pad bytes to push the byte ordering pattern to the LSB position, restoring proper byte ordering.
Enable rx_std_byteorder_ena port	**On/Off**	Enables the optional `rx_std_byte_order_ena` control input port. When this signal is asserted, the byte ordering block initiates a byte ordering operation if the **Byte ordering control mode** is set to **manual**. Once byte ordering has occurred, you must deassert and reassert this signal to perform another byte ordering operation. This signal is an synchronous input signal; however, it must be asserted for at least 1 cycle of `rx_std_clkout`.

Parameter	Range	Description
Enable rx_std_byteorder_flag port	**On/Off**	Enables the optional `rx_std_byteorder_flag` status output port. When asserted, indicates that the byte ordering block has performed a byte order operation. This signal is asserted on the clock cycle in which byte ordering occurred. This signal is synchronous to the `rx_std_clkout clock`.

Byte Serializer and Deserializer

The byte serializer and deserializer allow the PCS to operate at twice the data width of the PMA serializer. This feature allows the PCS to run at a lower frequency and accommodate a wider range of FPGA interface widths. The following table describes the byte serialization and deserialization options you can specify.

Table 15-13: Byte Serializer and Deserializer Parameters

Parameter	Range	Description
Enable TX byte serializer	**On/Off**	When you turn this option **On**, the PCS includes a TX byte serializer which allows the PCS to run at a lower clock frequency to accommodate a wider range of FPGA interface widths.
Enable RX byte deserializer	**On/Off**	When you turn this option **On**, the PCS includes an RX byte deserializer and deserializer which allows the PCS to run at a lower clock frequency to accommodate a wider range of FPGA interface widths.

8B/10B

The 8B/10B encoder generates 10-bit code groups from the 8-bit data and 1-bit control identifier. In 8-bit 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. The 8B/10B decoder decodes the data into an 8-bit data and 1-bit control identifier. The following table describes the 8B/10B encoder and decoder options.

Table 15-14: 8B/10B Encoder and Decoder Parameters

Parameter	Range	Description
Enable TX 8B/10B encoder	**On/Off**	When you turn this option **On**, the PCS includes the 8B/10B encoder.
Enable TX 8B/10B disparity control	**On/Off**	When you turn this option **On**, the PCS includes disparity control for the 8B/10B encoder. Your force the disparity of the 8B/10B encoder using the `tx_forcedisp` control signal.

Parameter	Range	Description
Enable RX 8B/10B decoder	**On/Off**	When you turn this option **On**, the PCS includes the 8B/10B decoder.

Rate Match FIFO

The rate match FIFO compensates for the very small frequency differences between the local system clock and the RX recovered clock. The following table describes the rate match FIFO parameters.

Table 15-15: Rate Match FIFO Parameters

Parameter	Range	Description
Enable RX rate match FIFO	**On/Off**	When you turn this option On , the PCS includes a FIFO to compensate for the very small frequency differences between the local system clock and the RX recovered clock.
RX rate match insert/delete +ve pattern (hex)	User-specified 20 bit pattern	Specifies the +ve (positive) disparity value for the RX rate match FIFO as a hexadecimal string.
RX rate match insert/delete -ve pattern (hex)	User-specified 20 bit pattern	Specifies the -ve (negative) disparity value for the RX rate match FIFO as a hexadecimal string.
Enable rx_std_rm_fifo_empty port	**On/Off**	When you turn this option **On**, the rate match FIFO outputs a FIFO empty status flag. The rate match FIFO compensates for small clock frequency differences between the upstream transmitter and the local receiver clocks by inserting or removing skip (SKP) symbols or ordered sets from the inter-packet gap (IPG) or idle stream. This port is only used for XAUI, GigE, and Serial RapidIO in double width mode. In double width mode, the FPGA data width is twice the PCS data width to allow the fabric to run at half the PCS frequency.
Enable rx_std_rm_fifo_full port	**On/Off**	When you turn this option **On**, the rate match FIFO outputs a FIFO full status flag. This port is only used for XAUI, GigE, and Serial RapidIO in double width mode.

When you enable the simplified data interface and enable the rate match FIFO status ports, the rate match FIFO bits map to the high-order bits of the data bus as listed in the following table. This table uses the following definitions:

- Basic double width: The **Standard PCS protocol mode** GUI option is set to **basic**. The FPGA data width is twice the PCS data width to allow the fabric to run at half the PCS frequency.
- Serial^{TM} RapidIO double width: You are implementing the Serial RapidIO protocol. The FPGA data width is twice the PCS data width to allow the fabric to run at half the PCS frequency.

Note: If you have the auto-negotiation state machine in your transceiver design, please note that the rate match FIFO is capable of inserting or deleting the first two bytes (K28.5//D2.2) of /C2/ ordered sets during auto-negotiation. However, the insertion or deletion of the first two bytes of /C2/ ordered sets can cause the auto-negotiation link to fail. For more information, visit **Altera Knowledge Base Support Solution**.

Table 15-16: Status Flag Mappings for Simplified Native PHY Interface

Status Condition	Protocol	Mapping of Status Flags to RX Data	Value
Full	PHY IP Core for PCI Express (PIPE) Basic double width	`RXD[62:62] = rx_rmfifostatus[1:0]`, or `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[30:29] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b11 = full`
	XAUI, GigE, Serial RapidIO double width	`rx_std_rm_fifo_full`	`1'b1 = full`
	All other protocols	Depending on the FPGA fabric to PCS interface width either: `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b11 = full`

Status Condition	Protocol	Mapping of Status Flags to RX Data	Value
Empty	PHY IP Core for PCI Express (PIPE) Basic double width	`RXD[62:62] = rx_rmfifostatus[1:0]`, or `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[30:29] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`(2'b10) AND (PAD or EDB)` PAD = K23.7 or 9'h1F7 EDB = K30.7 or 9'h1FE
	XAUI, GigE, Serial RapidIO double width	`rx_std_rm_fifo_empty`	`1'b1 = empty`
	All other protocols	Depending on the FPGA fabric to PCS interface width either: `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`(2'b10) AND (PAD or EDB)` (15) PAD = K23.7 or 9'h1F7 EDB = K30.7 or 9'h1FE
Insertion	Basic double width Serial RapidIO double width	`RXD[62:62] = rx_rmfifostatus[1:0]`, or `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[30:29] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b10`
	All other protocols	Depending on the FPGA fabric to PCS interface width either: `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b10`

(15) PAD and EBD are control characters. PAD character is typically used fo fill in the remaining lanes in a multi-lane link when one of the link goes to logical idle state. EDB indicates End Bad Packet.

Status Condition	Protocol	Mapping of Status Flags to RX Data	Value
Deletion	Basic double width Serial RapidIO double width	`RXD[62:62] = rx_rmfifostatus[1:0]`, or `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[30:29] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b01`
	All other protocols	Depending on the FPGA fabric to PCS interface width either: `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b01`

Word Aligner and Bit-Slip Parameters

The word aligner aligns the data coming from RX PMA deserializer to a given word boundary. When the word aligner operates in bit-slip mode, the word aligner slips a single bit for every rising edge of the bit slip control signal. The following table describes the word aligner and bit-slip parameters.

Table 15-17: Word Aligner and Bit-Slip Parameters

Parameter	Range	Description
Enable TX bit-slip	**On/Off**	When you turn this option **On**, the PCS includes the bit-slip function. The outgoing TX data can be slipped by the number of bits specified by the `tx_bitslipboundarysel` control signal.
Enable tx_std_bitslipboundarysel control input port	**On/Off**	When you turn this option On , the PCS includes the optional `tx_std_bitslipboundarysel` control input port.

Parameter	Range	Description
RX word aligner mode	**bit_slip** **sync_sm** **manual**	Specifies one of the following 3 modes for the word aligner: • **Bit_slip** : You can use bit slip mode to shift the word boundary. For every rising edge of the `rx_bitslip signal`, the word boundary is shifted by 1 bit. Each bit-slip removes the earliest received bit from the received data • **Sync_sm** : In synchronous state machine mode, a programmable state machine controls word alignment. You can only use this mode with 8B/10B encoding. The data width at the word aligner can be 10 or 20 bits. When you select this word aligner mode, the synchronous state machine has hysteresis that is compatible with XAUI. However, when you select **cpri** for the **Standard PCS Protocol Mode**, this option selects the deterministic latency word aligner mode. • **Manual** : This mode Enables word alignment by asserting the `rx_std_wa_patternalign`. This is an edge sensitive signal.
RX word aligner pattern length	**7**, **8**, **10** **16**, **20**, **32**	Specifies the length of the pattern the word aligner uses for alignment.
RX word aligner pattern (hex)	User-specified	Specifies the word aligner pattern in hex.
Number of word alignment patterns to achieve sync	`1-256`	Specifies the number of valid word alignment patterns that must be received before the word aligner achieves synchronization lock. The default is 3.
Number of invalid words to lose sync	`1-256`	Specifies the number of invalid data codes or disparity errors that must be received before the word aligner loses synchronization. The default is 3.
Number of valid data words to decrement error count	`1-256`	Specifies the number of valid data codes that must be received to decrement the error counter. If the word aligner receives enough valid data codes to decrement the error count to 0, the word aligner returns to synchronization lock.
Run length detector word count	`0-63`	Specifies the maximum number of contiguous 0s or 1s in the data stream before the word aligner reports a run length violation.

Parameter	Range	Description
Enable rx_std_wa_patternalign port	**On/Off**	Enables the optional `rx_std_wa_patternalign` control input port. A rising edge on this signal causes the word aligner to align the next incoming word alignment pattern when the word aligner is configured in **manual** mode.
Enable rx_std_wa_a1a2size port	**On/Off**	Enables the optional `rx_std_wa_a1a2size` control input port.
Enable rx_std_wa_bitslipboundarysel port	**On/Off**	Enables the optional `rx_std_wa_bitslipboundarysel` status output port.
Enable rx_std_wa_bitslip port	**On/Off**	Enables the optional `rx_std_wa_bitslip` control input port.
Enable rx_std_wa_runlength_err port	**On/Off**	Enables the optional `rx_std_wa_runlength_err` control input port.

Bit Reversal and Polarity Inversion

These functions allow you to reverse bit order, byte order, and polarity to correct errors and to accommodate different layouts of data. The following table describes these parameters.

Parameter	Range	Description
Enable TX bit reversal	**On/Off**	When you turn this option **On**, the word aligner reverses TX parallel data before transmitting it to the PMA for serialization. You can only change this static setting using the Transceiver Reconfiguration Controller.
Enable RX bit reversal	**On/Off**	When you turn this option **On**, the `rx_st_bitrev_ena` port controls bit reversal of the RX parallel data after it passes from the PMA to the PCS.
Enable RX byte reversal	**On/Off**	When you turn this option **On**, the word aligner reverses the byte order before transmitting data. This function allows you to reverse the order of bytes that were erroneously swapped. The PCS can swap the ordering of both 8 and10 bit words.
Enable TX polarity inversion	**On/Off**	When you turn this option **On**, the `tx_std_polinv` port controls polarity inversion of TX parallel data before transmitting the parallel data to the PMA.
Enable RX polarity inversion	**On/Off**	When you turn this option **On**, asserting `rx_std_polinv` controls polarity inversion of RX parallel data after PMA transmission.

Parameter	Range	Description
Enable rx_std_bitrev_ena port	**On/Off**	When you turn this option **On**, asserting `rx_std_bitrev_ena` control port causes the RX data order to be reversed from the normal order, LSB to MSB, to the opposite, MSB to LSB. This signal is an asynchronous input.
Enable rx_std_byterev_ena port	**On/Off**	When you turn this option **On**, asserting `rx_std_byterev_ena` input control port causes swaps the order of the individual 8- or 10-bit words received from the PMA.
Enable tx_std_polinv port	**On/Off**	When you turn this option **On**, the `tx_std_polinv` input is enabled. You can use this control port to swap the positive and negative signals of a serial differential link if they were erroneously swapped during board layout.
Enable rx_std_polinv port	**On/Off**	When you turn this option **On**, the `rx_std_polinv` input is enabled. You can use this control port to swap the positive and negative signals of a serial differential link if they were erroneously swapped during board layout.
Enable tx_std_elecidle port	**On/Off**	When you turn this option **On**, the `tx_std_elecidle` input port is enabled. When this signal is asserted, it forces the transmitter to electrical idle. This signal is required for the PCI Express protocol.
Enable rx_std_signaldetect port	**On/Off**	When you turn this option **On**, the optional `tx_std_signaldetect` output port is enabled. This signal is required for the PCI Express protocol. If enabled, the signal threshold detection circuitry senses whether the signal level present at the RX input buffer is above the signal detect threshold voltage that you specified. For SATA / SAS applications, enable this port and set the following QSF assignments to the transceiver receiver pin: • `set_instance_assignment -name XCVR_RX_SD_ENABLE ON` • `set_instance_assignment -name XCVR_RX_SD_THRESHOLD 7` • `set_instance_assignment -name XCVR_RX_COMMON_MODE_VOLTAGE VTT_OP55V` • `set_instance_assignment -name XCVR_RX_SD_OFF 1` • `set_instance_assignment -name XCVR_RX_SD_ON 2`

PRBS Verifier

You can use the PRBS pattern generators for verification or diagnostics. The pattern generator blocks support the following patterns:

- Pseudo-random binary sequence (PRBS)
- Square wave

Table 15-18: PRBS Parameters

Parameter	Range	Description
Enable rx_std_prbs ports	**On/Off**	When you turn this option **On**, the PCS includes the `rx_std_prbs_done` and `rx_std_prbs_err` signals to provide status on PRBS operation.

Related Information

Transceiver Architecture in Stratix V Devices

Standard PCS Pattern Generators

The Standard PCS includes a pattern generator that generates and verifies the PRBS patterns.

Table 15-19: Standard PCS PRBS Patterns

PATTERN	POLYNOMIAL
PRBS-7	$X^7 + X^6 + 1$
PRBS-8	$X^8 + X^7 + X^3 + X^2 + 1$
PRBS-10	$X^{10} + X^7 + 1$
PRBS-15	$X^{15} + X^{14} + 1$
PRBS-23	$X^{23} + X^{18} + 1$
PRBS-31	$X^{31} + X^{28} + 1$

The Standard PCS requires a specific word alignment for the PRBS pattern. You must specify a word alignment pattern in the verifier that matches the generator pattern specified. In the Standard PCS, PRBS patterns available depend upon the PCS-PMA width. The following table below illustrates the patterns are available based upon the PCS-PMA width.

Table 15-20: PRBS Patterns in the 8G PCS with PCS-PMA Widths

	PCS-PMA Width			
	8-Bit	10-Bit	16-Bit	20-Bit
PRBS-7	X		X	X
PRBS-8	X			
PRBS-10		X		
PRBS 15	X	X	X	X
PRBS 23	X		X	X
PRBS 31	X	X	X	X

Unlike the 10G PRBS verifier, the Standard PRBS verifier uses the Standard PCS word aligner. You must specify the word aligner size and pattern. The following table lists the encodings for the available choices.

Table 15-21: Word Aligner Size and Word Aligner Pattern

PCS-PMA Width	PRBS Patterns	PRBS Pattern Select	Word Aligner Size	Word Aligner Pattern
8-bit	PRBS 7	3'b010	3'b001	0x0000003040
	PRBS 8	3'b000	3'b001	0x000000FF5A
	PRBS 23	3'b100	3'b001	0x0000003040
	PRBS 15	3'b101	3'b001	0x0000007FFF
	PRBS 31	3'b110	3'b001	0x000000FFFF
10-bit	PRBS 10	3'b000	3'b010	0x00000003FF
	PRBS 15	3'b101	3'b000	0x0000000000
	PRBS 31	3'b110	3'b010	0x00000003FF
16-bit	PRBS 7	3'b000	3'b010	0x0000003040
	PRBS 23	3'b001	3'b101	0x00007FFFFF
	PRBS 15	3'b101	3'b011	0x0000007FFF
	PRBS 31	3'b110	3'b011	0x000000FFFF

PCS-PMA Width	PRBS Patterns	PRBS Pattern Select	Word Aligner Size	Word Aligner Pattern
20-bit	PRBS 7	3'b000	3'b100	0x0000043040
	PRBS 23	3'b001	3'b110	0x00007FFFFF
	PRBS 15	3'b101	3'b100	0x0000007FFF
	PRBS 31	3'b110	3'b110	0x007FFFFFFF

Registers and Values

The following table lists the offsets and registers for the Standard PCS pattern generator and verifier.

Note: All undefined register bits are reserved.

Table 15-22: Offsets for the Standard PCS Pattern Generator and Verifier

Offset	OffsetBits	R/W	Name	Description
0x97	[9]	R/W	`PRBS TX Enable`	When set to 1'b1, enables the PRBS generator.
	[8:6]	R/W	`PRBS Pattern Select`	Specifies the encoded PRBS pattern defined in the previous table.
0x99	[9]	R/W	`Clock Power Down TX`	When set to 1'b1, powers down the PRBS Clock in the transmitter. When set to 1'b0, enables the PRBS generator.
0x141	[0]	R/W	`PRBS TX Inversion`	Set to 1'b1 to invert the data leaving the PCS block.
0x16D	[2]	R/W	`PRBS RX Inversion`	Set to 1'b1 to invert the data entering the PCS block.
0xA0	[5]	R/W	`PRBS RX Enable`	When set to 1'b1, enables the PRBS verifier in the receiver.
	[4]	R/W	`PRBS Error Clear`	When set to 1'b1, deasserts `rx_prbs_` done and restarts the PRBS pattern.

Offset	OffsetBits	R/W	Name	Description
0xA1	[15:14]	R/W	Sync badcg	Must be set to 2'b00 to enable the PRBS verifier.
	[13]	R/W	Enable Comma Detect	Must be set to 1'b0 to enable the PRBS verifier.
	[11]	R/W	Enable Polarity	Must be set to 1'b0 to enable the PRBS verifier.
	[10:8]	R/W	Word Aligner Size	Specifies the word alignment size using the encodings defined in the previous table.
	[7:0]	R/W	Word Aligner Pattern [39:32]	Stores the high-order 8 bits of the word aligner pattern as specified in the previous table.
0xA2	[15:0]	R/W	Word Aligner Pattern [31:16]	Stores the middle 16 bits of the word aligner pattern as specified in the previous table.
0xA3	[15:0]	R/W	Word Aligner Pattern [15:0]	Stores the least significant 16 bits from the word aligner pattern as specified in the previous table.
0xA4	[15]	R/W	Sync State Machine Disable	Disables the synchronization state machine. When the PCS-PMA Width is 8 or 10, the value must be 1. When the PCS-PMA Width is 16 or 20, the value must be 0.
0xA6	[5]	R/W	Auto Byte Align Disable	Auto aligns the bytes. Must be set to 1'b0 to enable the PRBS verifier.
0xB8	[13]	R/W	DW Sync State Machine Enable	Enables the double width state machine. Must be set to 1'b0 to enable the PRBS verifier.
0xB9	[11]	R/W	Deterministic Latency State Machine Enable	Enables a deterministic latency state machine. Must be set to 1'b0 to enable the PRBS verifier.
0xBA	[11]	R/W	Clock Power Down RX	When set to 1'b0, powers down the PRBS clock in the receiver.

10G PCS Parameters for Arria V GZ Native PHY

This section shows the complete datapath and clocking for the 10G PCS and defines parameters available in the GUI to enable or disable the individual blocks in the 10G PCS.

Figure 15-4: The 10G PCS datapath

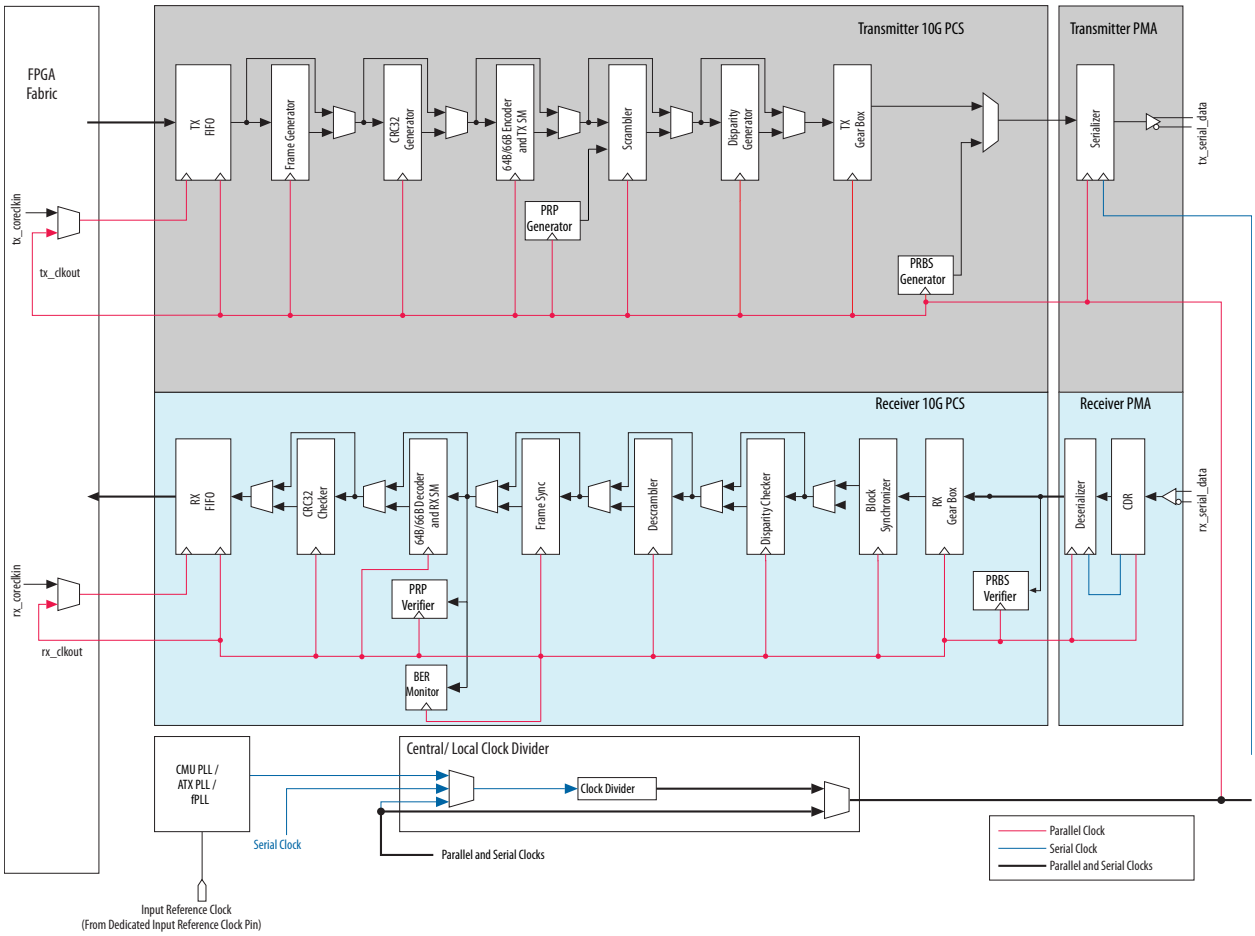


Table 15-23: General and Datapath Parameters

Parameter	Range	Description
10G PCS protocol mode	**basic** **interlaken** **sfi5** **teng_baser** **teng_sdi**	Specifies the protocol that you intend to implement with the Native PHY. The protocol mode selected guides the MegaWizard in identifying legal settings for the 10G PCS datapath. Use the following guidelines to select a protocol mode: • **basic** : Select this mode for when none of the other options are appropriate. You should also select this mode to enable diagnostics, such as loopback. • **interlaken**: Select this mode if you intend to implement Interlaken. • **sfi5** : Select this mode if you intend to implement the SERDES Framer Interface Level 5 protocol. • **teng_baser** : select this mode if you intend to implement the 10GBASE-R protocol. • **teng_sdi** : 10G SDI
10G PCS/PMA interface width	**32**, **40**, **64**	Specifies the width of the datapath that connects the FPGA fabric to the PMA.
FPGA fabric/10G PCS interface width	**32** **40** **50** **64** **66** **67**	Specifies the FPGA fabric to TX PCS interface width . The 66-bit FPGA fabric/PCS interface width is achieved using 64-bits from the TX and RX parallel data and the lower 2-bits from the control bus. The 67-bit FPGA fabric/PCS interface width is achieved using the 64-bits from the TX and RX parallel data and the lower 3-bits from the control bus.

10G TX FIFO

The TX FIFO is the interface between TX data from the FPGA fabric and the PCS. This FIFO is an asynchronous 73-bit wide, 32-deep memory buffer It also provides full, empty, partially full, and empty flags based on programmable thresholds. The following table describes the 10G TX FIFO parameters.

Table 15-24: 10G TX FIFO Parameters

Parameter	Range	Description
TX FIFO Mode	**Interlaken** **phase_comp** **register**	Specifies one of the following 3 modes: • **interlaken** : The TX FIFO acts as an elastic buffer. The FIFO write clock frequency (`coreclk`) can exceed that of the effective read clock, `tx_clkout`. You can control writes to the FIFO with `tx_data_valid`. By monitoring the FIFO flags, you can avoid the FIFO full and empty conditions. The Interlaken frame generator controls reads. • **phase_comp** : The TX FIFO compensates for the clock phase difference between the `coreclkin` and `tx_clkout` which is an internal PCS clock. • **register** : The TX FIFO is bypassed. `tx_data` and `tx_data_valid` are registered at the FIFO output. You must control `tx_data_valid` precisely based on gearbox ratio to avoid gearbox underflow or overflow conditions.
TX FIFO full threshold	`0-31`	Specifies the full threshold for the 10G PCS TX FIFO. The active high TX FIFO full flag is synchronous to `coreclk`. The default value is 31.
TX FIFO empty threshold	`0-31`	Specifies the empty threshold for the 10G PCS TX FIFO. The active high TX FIFO empty flag is synchronous to `coreclk`. The default value is 0.
TX FIFO partially full threshold	`0-31`	Specifies the partially full threshold for the 10G PCS TX FIFO. The active high TX FIFO partially full flag is synchronous to `coreclk`. The default value is 23.
TX FIFO partially empty threshold	`0-31`	Specifies the partially empty threshold for the 10G PCS TX FIFO. The active high TX FIFO partially empty flag is synchronous to `coreclk`.
Enable tx_10g_fifo_full port	**On/Off**	When you turn this option On , the 10G PCS includes the active high `tx_10g_fifo_full` port. `tx_10g_fifo_full` is synchronous to `coreclk`.

Parameter	Range	Description
Enable tx_10g_fifo_pfull port	**On/Off**	When you turn this option On , the 10G PCS includes the active high `tx_10g_fifo_pfull port`. `tx_10g_fifo_pfull` is synchronous to `coreclk`.
Enable tx_10g_fifo_empty port	**On/Off**	When you turn this option **On**, the 10G PCS includes the active high `tx_10g_fifo_empty` port. `tx_10g_fifo_empty` is pulse-stretched. It is asynchronous to `coreclk` and synchronous to `tx_clkout` which is the read clock.
Enable tx_10g_fifo_pempty port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `tx_10g_fifo_pempty` port.
Enable tx_10g_fifo_del port (10GBASE-R)	**On/Off**	When you turn this option **On**, the 10G PCS includes the active high `tx_10g_fifo_del` port. This signal is asserted when a word is deleted from the TX FIFO. This signal is only used for the 10GBASE-R protocol.
Enable tx_10g_fifo_insert port (10GBASE-R)	**On/Off**	When you turn this option **On**, the 10G PCS includes the active high `tx_10g_fifo_insert` port. This signal is asserted when a word is inserted into the TX FIFO. This signal is only used for the 10GBASE-R protocol.

10G RX FIFO

The RX FIFO is the interface between RX data from the FPGA fabric and the PCS. This FIFO is an asynchronous 73-bit wide, 32-deep memory buffer It also provides full, empty, partially full, and empty flags based on programmable thresholds. The following table describes the 10G RX FIFO parameters.

Table 15-25: 10G RX FIFO Parameters

Parameter	Range	Description
RX FIFO Mode	**Interlaken** **clk_comp** **phase_comp** **register**	Specifies one of the following 3 modes: • **interlaken** : Select this mode for the Interlaken protocol. To implement the deskew process. In this mode the FIFO acts as an elastic buffer. The FIFO write clock can exceed the read clock. Your implementation must control the FIFO write (`tx_datavalid`) by monitoring the FIFO flags. The read enable is controlled by the Interlaken Frame Generator. • **clk_comp** : This mode compensates for the clock difference between the PLD clock (`coreclkin`) and `rxclkout`. After block lock is achieved, idle ordered set insertions and deletions compensate for the clock difference between RX PMA clock and PLD clock up to ± 100 ppm. Use this mode for 10GBASE-R. • **phase_comp** : This mode compensates for the clock phase difference between the PLD clock (`coreclkin`) and `rxclkout`. • **register** : The TX FIFO is bypassed. `rx_data` and `rx_data_valid` are registered at the FIFO output.
RX FIFO full threshold	`0-31`	Specifies the full threshold for the 10G PCS RX FIFO. The default value is 31.
RX FIFO empty threshold	`0-31`	Specifies the empty threshold for the 10G PCS RX FIFO. The default value is 0.
RX FIFO partially full threshold	`0-31`	Specifies the partially full threshold for the 10G PCS RX FIFO. The default value is 23.
RX FIFO partially empty threshold	`0-31`	Specifies the partially empty threshold for the 10G PCS RX FIFO.
Enable RX FIFO deskew (interlaken)	**On/ Off**	When you turn this option **On**, the RX FIFO also performs deskew. This option is only available for the Interlaken protocol.

Parameter	Range	Description
Enable RX FIFO alignment word deletion (interlaken)	**On/Off**	When you turn this option **On**, all alignment words (sync words), including the first sync word, are removed after frame synchronization is achieved. If you enable this option, you must also enable control word deletion.
Enable RX FIFO control word deletion (interlaken)	**On/Off**	When you turn this option On , the `rx_control_del` parameter enables or disables writing the Interlaken control word to RX FIFO. When disabled, a value of 0 for `rx_control_del` writes all control words to RX FIFO. When enabled, a value of 1 deletes all control words and only writes the data to the RX FIFO.
Enable rx_10g_fifo_data_valid port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_data_valid` signal which Indicates when `rx_data` is valid. This option is available when you select the following parameters: • 10G PCS protocol mode is Interlaken • 10G PCS protocol mode is Basic and RX FIFO mode is phase_comp • 10G PCS protocol mode is Basic and RX FIFO mode is register
Enable rx_10g_fifo_full port	**On/Off**	When you turn this option **On**, the 10G PCS includes the active high `rx_10g_fifo_full` port. `rx_10g_fifo_full` is synchronous to `rx_clkout`.
Enable rx_10g_fifo_pfull port	**On/Off**	When you turn this option **On**, the 10G PCS includes the active high `rx_10g_fifo_pfull` port. `rx_10g_fifo_pfull` is synchronous to `rx_clkout`.
Enable rx_10g_fifo_empty port	**On/Off**	When you turn this option **On**, the 10G PCS includes the active high `rx_10g_fifo_empty` port.
Enable rx_10g_fifo_pempty port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_fifo_pempty` port.
Enable rx_10g_fifo_del port (10GBASE-R)	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_fifo_del` port. This signal is asserted when a word is deleted from the RX FIFO. This signal is only used for the 10GBASE-R protocol.

Parameter	Range	Description
Enable rx_10g_fifo_insert port (10GBASE-R)	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_fifo_insert` port. This signal is asserted when a word is inserted into the RX FIFO. This signal is only used for the 10GBASE-R protocol.
Enable rx_10g_fifo_rd_en port (Interlaken)	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_fifo_rd_en` input port. Asserting this signal reads a word from the RX FIFO. This signal is only available for the Interlaken protocol.
Enable rx_10g_fifo_align_val port (Interlaken)	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_fifo_align_val` output port. This signal is asserted when the word alignment pattern is found. This signal is only available for the Interlaken protocol.
enable rx10g_clk33out port	**On/Off**	When you turn this option **On**, the 10G PCS includes a divide by 33 clock output port. You typically need this option when the fabric to PCS interface width is 66.
Enable rx_10g_fifo_align_clr port (Interlaken)	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_fifo_align_clr` input port. When this signal is asserted, the FIFO resets and begins searching for a new alignment pattern. This signal is only available for the Interlaken protocol.
Enable rx_10g_fifo_align_en port (Interlaken)	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_fifo_align_en` input port. This signal is used for FIFO deskew for Interlaken. When asserted, the corresponding channel is enabled for alignment. This signal is only available for the Interlaken protocol.

Interlaken Frame Generator

TX Frame generator generates the metaframe. It encapsulates the payload from MAC with the framing layer control words, including sync, scrambler, skip and diagnostic words. The following table describes the Interlaken frame generator parameters.

Table 15-26: Interlaken Frame Generator Parameters

Parameter	Range	Description
teng_tx_framgen_enable	**On/Off**	When you turn this option **On**, the frame generator block of the 10G PCS is enabled.
teng_tx_framgen_user_length	0-8192	Specifies the metaframe length.
teng_tx_framgen_burst_enable	**On/Off**	When you turn this option **On**, the frame generator burst functionality is enabled.
Enable tx_10g_frame port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `tx_10g_frame` output port. When asserted, `tx_10g_frame` indicates the beginning of a new metaframe inside the frame generator.
Enable tx_10g_frame_diag_status port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `tx_10g_frame_diag_status` 2-bit input port. This port contains the lane Status Message from the framing layer Diagnostic Word, bits[33:32]. This message is inserted into the next Diagnostic Word generated by the frame generation block. The message must be held static for 5 cycles before and 5 cycles after the `tx_frame` pulse.
Enable tx_10g_frame_burst_en port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `tx_10g_frame_burst_en` input port. This port controls frame generator data reads from the TX FIFO. The value of this signal is latched once at the beginning of each Metaframe. It controls whether data is read from the TX FIFO or SKIP Words are inserted for the current Metaframe. It must be held static for 5 cycles before and 5 cycles after the `tx_frame` pulse. When `tx_10g_frame_burst_en` is 0, the frame generator does not read data from the TX FIFO for current Metaframe. It insert SKIPs. When `tx_10g_frame_burst_en` is 1, the frame generator reads data from the TX FIFO for current Metaframe.

Interlaken Frame Synchronizer

The Interlaken frame synchronizer block achieves lock by looking for four synchronization words in consecutive metaframes. After synchronization, the frame synchronizer monitors the scrambler word in the metaframe and deasserts the lock signal after three consecutive mismatches and starts the synchroni-

zation process again. Lock status is available to the FPGA fabric. The following table describes the Interlaken frame synchronizer parameters.

Table 15-27: Interlaken Frame Synchronizer Parameters

Parameter	Range	Description
teng_tx_framsync_enable	**On/Off**	When you turn this option **On**, the 10G PCS frame generator is enabled.
Enable rx_10g_frame port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame` output port. This signal is asserted to indicate the beginning of a new metaframe inside.
Enable rx_10g_frame_lock_port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame_lock` output port. This signal is asserted to indicate that the frame synchronization state machine has achieved frame lock.
Enable rx_10g_frame_mfrm_err port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame_mfrm_err` output port. This signal is asserted to indicate an metaframe error.
Enable rx_10g_frame_sync_err port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame_sync_err` output port. This signal is asserted to indicate synchronization control word errors. This signal remains asserted during the loss of block_lock and does not update until block_lock is recovered.
Enable rx_10g_frame_skip_ins port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame_skip_ins` output port. This signal is asserted to indicate a SKIP word was received by the frame sync in a non-SKIP word location within the metaframe.
Enable rx_10g_frame_pyld_ins port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame_pyld_ins` output port. This signal is asserted to indicate a SKIP word was not received by the frame sync in a SKIP word location within the metaframe.

Parameter	Range	Description
Enable rx_10g_frame_skip_err port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame_skip_err` output port. This signal is asserted to indicate the frame synchronization has received an erroneous word in a Skip control word location within the Metaframe. This signal remains asserted during the loss of block_lock and does update until block_lock is recovered.
Enable rx_10g_frame_diag_err port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame_diag_err` output port. This signal is asserted to indicate a diagnostic control word error. This signal remains asserted during the loss of block_lock and does update until block_lock is recovered.
Enable rx_10g_frame_diag_status port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_frame_diag_status` 2-bit output port per channel. This port contains the lane Status Message from the framing layer Diagnostic Word, bits[33:32]. This message is inserted into the next Diagnostic Word generated by the frame generation block.

Interlaken CRC32 Generator and Checker

CRC-32 provides a diagnostic tool on a per-lane basis. You can use CRC-32 to trace interface errors back to an individual lane. The CRC-32 calculation covers the whole metaframe including the Diagnostic Word itself. This CRC code value is stored in the CRC32 field of the Diagnostic Word. The following table describes the CRC-32 parameters.

Table 15-28: Interlaken CRC32 Generator and Checker Parameters

Parameter	Range	Description
Enable Interlaken TX CRC32 Generator	**On/Off**	When you turn this option **On**, the TX 10G PCS datapath includes the CRC32 function.
Enable Interlaken RX CRC32 Generator	**On/Off**	When you turn this option **On**, the RX 10G PCS datapath includes the CRC32 function.
Enable rx_10g_crc32_err port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_crc32_err` port. This signal is asserted to indicate that the CRC checker has found an error in the current metaframe.

10GBASE-R BER Checker

The BER monitor block conforms to the 10GBASE-R protocol specification as described in IEEE 802.3-2008 Clause-49. After block lock is achieved, the BER monitor starts to count the number of invalid synchronization headers within a 125-ms period. If more than 16 invalid synchronization headers are observed in a 125-ms period, the BER monitor provides the status signal to the FPGA fabric, indicating a high bit error. The following table describes the 10GBASE-R BER checker parameters.

Table 15-29: 10GBASE-R BER Checker Parameters

Parameter	Range	Description
Enable rx_10g_highber port (10GBASE-R)	**On/Off**	When you turn this option **On**, the TX 10G PCS datapath includes the `rx_10g_highber` output port. This signal is asserted to indicate a BER of $>10^{-4}$. A count of 16 errors in 125- m s period indicates a BER $> 10^{-4}$. This signal is only available for the 10GBASE-R protocol.
Enable rx_10g_highber_clr_cnt port (10GBASE-R)	**On/Off**	When you turn this option **On**, the TX 10G PCS datapath includes the `rx_10g_highber_clr_cnt` input port. When asserted, the BER counter resets to 0. This signal is only available for the 10GBASE-R protocol.
Enable rx_10g_clr_errblk_count port (10GBASE-R)	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_clr_errblk_count` input port. When asserted, error block counter that counts the number of RX errors resets to 0. This signal is only available for the 10GBASE-R protocol.

64b/66b Encoder and Decoder

The 64b/66b encoder and decoder conform to the 10GBASE-R protocol specification as described in IEEE 802.3-2008 Clause-49. The 64b/66b encoder sub-block receives data from the TX FIFO and encodes the 64-bit data and 8-bit control characters to the 66-bit data block required by the 10GBASE-R protocol. The transmit state machine in the 64b/66b encoder sub-block checks the validity of the 64-bit data from the MAC layer and ensures proper block sequencing.

The 64b/66b decoder sub-block converts the received data from the descrambler into 64-bit data and 8-bit control characters. The receiver state machine sub-block monitors the status signal from the BER monitor. The following table describes the 64/66 encoder and decoder parameters.

Table 15-30: 64b/66b Encoder and Decoder Parameters

Parameter	Range	Description
Enable TX sync header error insertion	**On/Off**	When you turn this option **On**, the 10G PCS records. This parameter is valid for the Interlaken and 10GBASE-R protocols.

Parameter	Range	Description
Enable TX 64b/66b encoder	**On/Off**	When you turn this option **On**, the 10G PCS includes the TX 64b/66b encoder.
Enable TX 64b/66b encoder	**On/Off**	When you turn this option **On**, the 10G PCS includes the RX 64b/66b decoder.

Scrambler and Descrambler Parameters

TX scrambler randomizes data to create transitions to create DC-balance and facilitate CDR circuits based on the $x^{58} + x^{39} + 1$ polynomial. The scrambler operates in the following two modes:

- Synchronous—The Interlaken protocol requires synchronous mode.
- Asynchronous (also called self-synchronized)—The 10GBASE-R protocol requires this mode as specified in IEEE 802.3-2008 Clause-49.

The descrambler block descrambles received data to regenerate unscrambled data using the x58+x39+1 polynomial. The following table describes the scrambler and descrambler parameters.

Table 15-31: Scrambler and Descrambler Parameters

Parameter	Range	Description
Enable TX scrambler	**On/Off**	When you turn this option **On**, the TX 10G PCS datapath includes the scrambler function. This option is available for the Interlaken and 10GBASE-R protocols.
TX scrambler seed	User-specified 15-bit value	You must provide a different seed for each lane. This parameter is only required for the Interlaken protocol.
Enable RX scrambler	**On/Off**	When you turn this option **On**, the RX 10G PCS datapath includes the scrambler function. This option is available for the Interlaken and 10GBASE-R protocols.
Enable rx_10g_descram_err port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_descram_err` port.

Interlaken Disparity Generator and Checker

The Disparity Generator monitors the data transmitted to ensure that the running disparity remains within a ±96-bit bound. It adds the 67th bit to indicate whether or not the data is inverted. The Disparity Checker monitors the status of the 67th bit of the incoming word to determine whether or not to invert bits[63:0] of the received word. The following table describes Interlaken disparity generator and checker parameters.

Table 15-32: Interlaken Disparity Generator and Checker Parameters

Parameter	Range	Description
Enable Interlaken TX disparity generator	**On/Off**	When you turn this option **On**, the 10G PCS includes the disparity generator. This option is available for the Interlaken protocol.
Enable Interlaken RX disparity generator	**On/Off**	When you turn this option **On**, the 10G PCS includes the disparity checker. This option is available for the Interlaken protocol.

Block Synchronization

The block synchronizer determines the block boundary of a 66-bit word for the 10GBASE-R protocol or a 67-bit word for the Interlaken protocol. The incoming data stream is slipped one bit at a time until a valid synchronization header (bits 65 and 66) is detected in the received data stream. After the predefined number of synchronization headers is detected, the block synchronizer asserts `rx_10g_blk_lock` to other receiver PCS blocks down the receiver datapath and to the FPGA fabric. The block synchronizer is designed in accordance with both the Interlaken protocol specification and the 10GBASE-R protocol specification as described in IEEE 802.3-2008 Clause-49.

Table 15-33: Bit Reversal and Polarity Inversion Parameters

Parameter	Range	Description
Enable RX block synchronizer	**On/Off**	When you turn this option **On**, the 10G PCS includes the RX block synchronizer. This option is available for the Interlaken and 10GBASE-R protocols.
Enable rx_10g_blk_lock port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10G_blk_lock` output port. This signal is asserted to indicate the receiver has achieved block synchronization. This option is available for the Interlaken, 10GBASE-R, and other protocols that user the PCS lock state machine to achieve and monitor block synchronization.
Enable rx_10g_blk_sh_err port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10G_blk_sh_err` output port. This signal is asserted to indicate that an invalid sync header has been received. This signal is active after block lock is achieved. This option is available for the Interlaken, 10GBASE-R, and other protocols that user the PCS lock state machine to achieve and monitor block synchroniza-tion.

Gearbox

The gearbox adapts the PMA data width to a wider PCS data width when the PCS is not two or four times the PMA width.

Table 15-34: Gearbox Parameters

Parameter	Range	Description
Enable TX data polarity inversion	**On/Off**	When you turn this option **On**, the gearbox inverts the polarity of TX data allowing you to correct incorrect placement and routing on the PCB.
Enable TX data bitslip	**On/Off**	When you turn this option **On**, the TX gearbox operates in bitslip mode.
Enable RX data polarity inversion	**On/Off**	When you turn this option **On**, the gearbox inverts the polarity of RX data allowing you to correct incorrect placement and routing on the PCB.
Enable RX data bitslip	**On/Off**	When you turn this option **On**, the 10G PCS RX block synchronizer operates in bitslip mode.
Enable tx_10g_bitslip port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `tx_10g_bitslip` input port. The data slips 1 bit for every positive edge of the `tx_10g_bitslip` input. The maximum shift is < *pcswidth*> -1 bits, so that if the PCS is 64 bits wide, you can shift 0-63 bits.
Enable rx_10g_bitslip port	**On/Off**	When you turn this option **On**, the 10G PCS includes the `rx_10g_bitslip` input port. The data slips 1 bit for every positive edge of the `rx_10g_bitslip` input. he maximum shift is < *pcswidth*> -1 bits, so that if the PCS is 64 bits wide, you can shift 0-63 bits.

PRBS Verifier

You can use the PRBS pattern generators for verification or diagnostics. The pattern generator blocks support the following patterns:

- Pseudo-random binary sequence (PRBS)
- Pseudo-random pattern
- Square wave

Table 15-35: PRBS Parameters

Parameter	Range	Description
Enable rx_10g_prbs ports	**On/Off**	When you turn this option **On**, the PCS includes the `rx_10g_prbs_done` , `rx_10g_prbs_err` and `rx_10g_prbs_err_clr`signals to provide status on PRBS operation.

Related Information

Transceiver Archictecture in Arria V GZ Devices

10G PCS Pattern Generators

The 10G PCS supports the PRBS, pseudo-random pattern, and square wave pattern generators. You enable the pattern generator or verifiers in the 10G PCS, by writing a 1 to the `TX Test Enable` and `RX Test Enable` bits. The following table lists the offsets and registers of the pattern generators and verifiers in the 10G PCS.

Note:
- The 10G PRBS generator inverts its pattern before transmission. The 10G PRBS verifier inverts the received pattern before verification. You may need to invert the patterns if you connect to third-party PRBS pattern generators and checkers.
- All undefined register bits are reserved.

Table 15-36: Pattern Generator Registers

Offset	Bits	R/W	Name	Description
0x12D	[15:0]	R/W	`Seed A for PRP`	Bits [15:0] of seed A for the pseudo-random pattern.
0x12E	[15:0]			Bits [31:16] of seed A for the pseudo-random pattern.
0x12F	[15:0]			Bits [47:21] of seed A for the pseudo-random pattern.
0x130	[9:0]			Bits [57:48] of seed A for the pseudo-random pattern.
0x131	[15:0]	R/W	`Seed B for PRP`	Bits [15:0] of seed B for the pseudo-random pattern.
0x132	[15:0]			Bits [31:16] of seed B for the pseudo-random pattern.
0x133	[15:0]			Bits [47:32] of seed B for the pseudo-random pattern.
0x134	[9:0]			Bits [57:48] of seed B for the pseudo-random pattern.

Offset	Bits	R/W	Name	Description
0x135	[15:12]	R/W	Square Wave Pattern	Specifies the number of consecutive 1s and 0s. The following values are available: 1, 4, 5, 6, 8, and 10.
	[10]	R/W	TX PRBS 7 Enable	Enables the PRBS-7 polynomial in the transmitter.
	[8]	R/W	TX PRBS 23 Enable	Enables the PRBS-23 polynomial in the transmitter.
	[6]	R/W	TX PRBS 9 Enable	Enables the PRBS-9 polynomial in the transmitter.
	[4]	R/W	TX PRBS 31 Enable	Enables the PRBS-31 Polynomial in the transmitter.
	[3]	R/W	TX Test Enable	Enables the pattern generator in the transmitter.
	[1]	R/W	TX Test Pattern Select	Selects between the square wave or pseudo-random pattern generator. The following encodings are defined: • 1'b1: Square wave • 1'b0: Pseudo-random pattern or PRBS
	[0]	R/W	Data Pattern Select	Selects the data pattern for the pseudo-random pattern. The following encodings are defined: • 1'b1: Two Local Faults. Two, 32-bit ordered sets are XORd with the pseudo-random pattern. • 1'b0: All 0's
0x137	[2]	R/W	TX PRBS Clock Enable	Enables the transmitter PRBS clock.
	[1]	R/W	TX Square Wave Clock Enable	Enables the square wave clock.

Offset	Bits	R/W	Name	Description
0x15E	[14]	R/W	RX PRBS 7 Enable	Enables the PRBS-7 polynomial in the receiver.
	[13]	R/W	RX PRBS 23 Enable	Enables the PRBS-23 polynomial in the receiver.
	[12]	R/W	RX PRBS 9 Enable	Enables the PRBS-9 polynomial in the receiver.
	[11]	R/W	RX PRBS 31 Enable	Enables the PRBS-31 polynomial in the receiver.
	[10]	R/W	RX Test Enable	Enables the PRBS pattern verifier in the receiver.
0x164	[10]	R/W	RX PRBS Clock Enable	Enables the receiver PRBS Clock.
0x169	[0]	R/W	RX Test Pattern Select	Selects between a square wave or pseudo-random pattern. The following encodings are defined: • 1'b1: Square wave • 1'b0: Pseudo-random pattern or PRBS

PRBS Pattern Generator

To enable the PRBS pattern generator, write 1'b1 to the `RX PRBS Clock Enable` and `TX PRBS Clock Enable` bits.

The following table shows the available PRBS patterns:

Table 15-37: 10G PCS PRBS Patterns

Pattern	Polynomial
PRBS-31	$X^{31}+x^{28}+1$
PRBS-9	$X^{9}+x^{5}+1$
PRBS-23	$X^{23}+x^{18}+1$
PRBS-7	$X^{7}+x^{6}+1$

Pseudo-Random Pattern Generator

The pseudo-random pattern generator is specifically designed for the 10GBASE-R and 1588 protocols. To enable this pattern generator, write the following bits:

- Write 1'b0 to the `TX Test Pattern Select` bit.
- Write 1'b1 to the `TX Test Enable` bit.

In addition you have the following options:

- You can toggle the `Data Pattern Select` bit switch between two data patterns.
- You can change the value of `Seed A` and `Seed B`.

Unlike the PRBS pattern generator, the pseudo-random pattern generator does not require a configurable clock.

Square Wave Generator

To enable the square wave, write the following bits:

- Write 1'b1 to the `TX Test Enable` bit.
- Write 1'b1 to the `Square Wave Clock Enable` bit.
- Write 1'b1 to the `TX Test Select` bit.
- Write the `Square Wave Pattern` to 1, 4, 5, 6, 8 or 10 consecutive 1s or 0s.

The RX datapath does not include a verifier for the square wave and does drive a clock.

Interfaces for Arria V GZ Native PHY

This section describes the interfaces available for the Arria V GZ native PHY.

The Native PHY includes several interfaces that are common to all parameterizations. It also has separate interfaces for the Standard and 10G PCS datapaths. If you use dynamic reconfiguration to change between the Standard and 10G PCS datapaths, your top-level HDL file includes the port for both the Standard and 10G PCS datapaths. In addition, the Native PHY allows you to enable ports, even for disabled blocks to facilitate dynamic reconfiguration.

The Native PHY uses the following prefixes for port names:

- Standard PCS ports—tx_std_, rx_std_
- 10G PCS ports—tx_10g_, rx_10g_
- PMA ports—tx_pma_, rx_pma_

The port descriptions use the following variables to represent parameters:

- <*n*>—The number of lanes
- <*p*>—The number of PLLs
- <*r*>—the number of CDR references clocks selected

Common Interface Ports for Arria V GZ Native PHY

This section describes the interface ports for the Arria V GZ native PHY.

Common interface consists of reset, clock signals, serial interface ports, control and status ports, parallel data ports, PMA ports and reconfig interface ports. The following figure illustrates these ports.

Figure 15-5: Arria V GZ Native PHY Common Interfaces

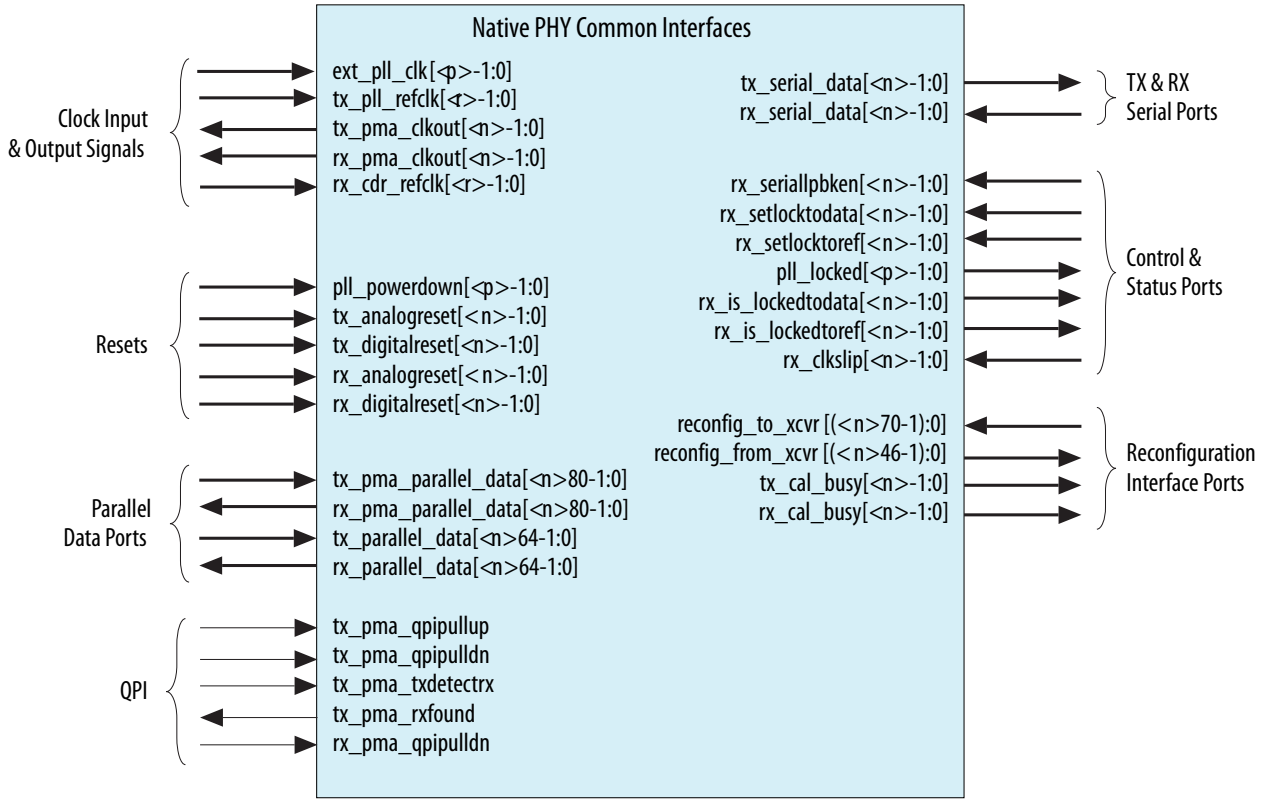


Table 15-38: Native PHY Common Interfaces

Name	Direction	Description
Clock Inputs and Output Signals		
`tx_pll_refclk [<r> -1:0]`	Input	The reference clock input to the TX PLL.
`tx_pma_clkout [<n> -1:0]`	Output	TX parallel clock output from PMA
`rx_pma_clkout [<n> -1:0]`	Output	RX parallel clock (recovered clock) output from PMA
`rx_cdr_refclk [<n> -1:0]`	Input	Input reference clock for the RX PFD circuit.
`ext_pll_clk[ <p> -1:0]`	Input	This optional signal is created when you select the **Use external TX PLL** option. If you instantiate a fractional PLL which is external to the Native PHY IP, then connect the output clock of this PLL to `ext_pll_clk`.

Name	Direction	Description
Resets		
pll_powerdown [<*n*> -1:0]	Input	When asserted, resets the TX PLL. Active high, edge sensitive reset signal. By default, the Arria V GZ Native Transceiver PHY IP Core creates a separate pll_powerdown signal for each logical PLL. However, the Fitter may merge the PLLs if they are in the same transceiver bank. PLLs can only be merged if their pll_powerdown signals are driven from the same source. If the PLLs are in separate transceiver banks, you can choose to drive the pll_powerdown signals separately.
tx_analogreset [<*n*> -1:0]	Input	When asserted, resets for TX PMA, TX clock generation block, and serializer. Active high, edge sensitive reset signal.
tx_digitalreset [<*n*> -1:0]	Input	When asserted, resets the digital components of the TX datapath. Active high, edge sensitive reset signal.If your design includes bonded TX PCS channels, refer to *Timing Constraints for Reset Signals when Using Bonded PCS Channels* for a SDC constraint you must include in your design.
rx_analogreset [<*n*> -1:0]	Input	When asserted, resets the RX CDR, deserializer, Active high, edge sensitive reset signal.
rx_digitalreset [<*n*> -1:0]	Input	When asserted, resets the digital components of the RX datapath. Active high, edge sensitive reset signal.
Parallel Data Ports		
tx_pma_parallel_data [<*n*> 80-1:0]	Input	TX parallel data for the PMA Direct datapath. Driven directly from the FPGA fabric to the PMA. Not used when you enable either the Standard or 10G PCS datapath.
rx_pma_parallel_data [<*n*> 80-1:0]	Output	RX PMA parallel data driven from the PMA to the FPGA fabric. Not used when you enable either the Standard or 10G PCS datapath.

Name	Direction	Description
tx_parallel_data [<*n*> 64-1:0]	Input	PCS TX parallel data. Used when you enable either the Standard or 10G datapath. For the Standard datapath, if you turn on Enable simplified data interface , tx_parallel_data includes only the data and control signals necessary for the current configuration. Dynamic reconfiguration of the interface is not supported. For the 10G PCS, if the parallel data interface is less than 64 bits wide, the low-order bits of tx_parallel_data are valid. For the 10G PCS operating in 66:40 Basic mode, the 66 bus is formed as follows: { tx_parallel_data[63:0],tx_10g_control[0], tx_10g_control[1]}. For the Standard PCS, refer to **Table 15-39** for bit definitions. Refer to **Table 15-40** for various parameterizations.
rx_parallel_data [<*n*> 64-1:0]	Output	PCS RX parallel data. Used when you enable either the Standard or 10G datapath. For the Standard datapath, if you turn on Enable simplified data interface , rx_parallel_data includes only the data and control signals necessary for the current configuration. Dynamic reconfiguration of the interface is not supported. For the 10G PCS, if the parallel data interface is less than 64 bits wide, the low-order bits of rx_parallel_data are valid. For the 10G PCS operating in 66:40 mode, the 66 bus is formed as follows: { rx_parallel_data[63:0],rx_10g_control[0], rx_10g_control[1]}. For the Standard PCS, refer to **Table 15-41** for bit definitions. Refer to **Table 15-42** for various parameterizations.
QPI		
tx_pma_qpipullup	Input	Control input port for Quick Path Interconnect (QPI) applications. When asserted, the transmitted pulls the output signal to high state. Use this port only for QPI applications.
tx_pma_qpipulldn	Input	Control input port for Quick Path Interconnect (QPI) applications. This is an active low signal. When asserted, the transmitter pulls the output signal to low state. Use this port only for QPI applications.

Name	Direction	Description
tx_pma_txdetectrx	Input	When asserted, the RX detect block in the TX PMA detects the presence of a receiver at the other end of the channel. After receiving a tx_pma_txdetectrx request, the receiver detect block initiates the detection process. Only for QPI applications.
tx_pma_rxfound	Output	Indicates the status of an RX detection in the TX PMA. Only for QPI applications.
rx_pma_qpipulldn	Input	Control input port for Quick Path Interconnect (QPI) applications. This is an active low signal. When asserted, the receiver pulls the input signal to low state. Use this port only for QPI applications.
TX and RX Serial Ports		
tx_serial_data [<*n*> -1:0]	Output	TX differential serial output data.
rx_serial_data [<*n*> -1:0]	Input	RX differential serial output data.
Control and Status Ports		
rx_seriallpbken [<*n*> -1:0]	Input	When asserted, the transceiver enters loopback mode. Loopback drives TX data to the RX interface.
rx_set_locktodata [<*n*> -1:0]	Input	When asserted, programs the RX CDR to manual lock to data mode in which you control the reset sequence using the rx_setlocktoref and rx_setlocktodata. Refer to *Reset Sequence for CDR in Manual Lock Mode* in *Transceiver Reset Control in Arria V GZ Devices* for more information about manual control of the reset sequence.
rx_set_locktoref [<*n*> -1:0]	Input	When asserted, programs the RX CDR to manual lock to reference mode in which you control the reset sequence using the rx_setlocktoref and rx_setlocktodata. Refer to *Reset Sequence for CDR in Manual Lock Mode* in *Transceiver Reset Control in Arria V GZ Devices* for more information about manual control of the reset sequence.
pll_locked [<*p*> -1:0]	Output	When asserted, indicates that the PLL is locked to the input reference clock.
rx_is_lockedtodata [<*n*> -1:0]	Output	When asserted, the CDR is locked to the incoming data.

Name	Direction	Description
rx_is_lockedtoref [<n> -1:0]	Output	When asserted, the CDR is locked to the incoming reference clock.
rx_clkslip [<n> -1:0]	Input	When you turn this signal on, deserializer performs a clock slip operation to achieve word alignment. The clock slip operation alternates between skipping 1 serial bit and pausing the serial clock for 2 cycles to achieve word alignment. As a result, the period of the parallel clock could be extended by 2 unit intervals (UI) during the clock slip operation. This is an optional control input signal.
Reconfig Interface Ports		
reconfig_to_xcvr [(<n> 70-1):0]	Input	Reconfiguration signals from the Transceiver Reconfiguration Controller. <*n*> grows linearly with the number of reconfiguration interfaces.
reconfig_from_xcvr [(<n> 46-1):0]	Output	Reconfiguration signals to the Transceiver Reconfiguration Controller. <*n*> grows linearly with the number of reconfiguration interfaces.
tx_cal_busy [<n> -1:0]	Output	When asserted, indicates that the initial TX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP. You must hold the channel in reset until calibration completes.
rx_cal_busy [<n> -1:0]	Output	When asserted, indicates that the initial RX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP.

Table 15-39: Signal Definitions for tx_parallel_data with and without 8B/10B Encoding

The following table shows the signals within tx_parallel_data that correspond to data, control, and status signals. The tx_parallel_data bus is always 64 bits to enable reconfigurations between the Standard and 10G PCS datapaths. If you only enable the Standard datapath, the 20, high-order bits are not used.

TX Data Word	Description
Signal Definitions with 8B/10B Enabled	
tx_parallel_data[7:0]	TX data bus
tx_parallel_data[8]	TX data control character
tx_parallel_data[9]	Force disparity, validates disparity field.

TX Data Word	Description
tx_parallel_data[10]	Specifies the current disparity as follows: • 1'b0 = positive • 1'b1 = negative
Signal Definitions with 8B/10B Disabled	
tx_parallel_data[9:0]	TX data bus
tx_parallel_data[10]	Unused

Table 15-40: Location of Valid Data Words for tx_parallel_data for Various FPGA Fabric to PCS Parameterizations

The following table shows the valid 11-bit data words with and without the byte deserializer for single- and double-word FPGA fabric to PCS interface widths.

Configuration	Bus Used Bits
Single word data bus, byte deserializer disabled	[10:0] (word 0)
Single word data bus, byte serializer enabled	[32:22], [10:0] (words 0 and 2)
Double word data bus, byte serializer disabled	[21:0] (words 0 and 1)
Double word data bus, byte serializer enabled	[43:0] (words 0-3)

Table 15-41: Signal Definitions for rx_parallel_data with and without 8B/10B Encoding

This table shows the signals within rx_parallel_data that correspond to data, control, and status signals.

RX Data Word	Description
Signal Definitions with 8B/10B Enabled	
rx_parallel_data[7:0]	RX data bus
rx_parallel_data[8]	RX data control character
rx_parallel_data[9]	Error Detect
rx_parallel_data[10]	Word Aligner / synchronization status
rx_parallel_data[11]	Disparity error
rx_parallel_data[12]	Pattern detect
rx_parallel_data[14:13]	The following encodings are defined: • 2'b00: Normal data • 2'b01: Deletion • 2'b10: Insertion • 2'b11: Underflow
rx_parallel_data[15]	Running disparity value
Signal Definitions with 8B/10B Disabled	
rx_parallel_data[9:0]	RX data bus

RX Data Word	Description
rx_parallel_data[10]	Word Aligner / synchronization status
rx_parallel_data[11]	Disparity error
rx_parallel_data[12]	Pattern detect
rx_parallel_data[14:13]	The following encodings are defined: • 2'b00: Normal data • 2'b01: Deletion • 2'b10: Insertion (or Underflow with 9'h1FE or 9'h1F7) • 2'b11: Overflow
rx_parallel_data[15]	Running disparity value

Table 15-42: Location of Valid Data Words for rx_parallel_data for Various FPGA Fabric to PCS Parameterizations

The following table shows the valid 16-bit data words with and without the byte deserializer for single- and double-word FPGA fabric to PCS interface widths.

Configuration	Bus Used Bits
Single word data bus, byte deserializer disabled	[15:0] (word 0)
Single word data bus, byte serializer enabled	[47:32], [15:0] (words 0 and 2)
Double word data bus, byte serializer disabled	[31:0] (words 0 and 1)
Double word data bus, byte serializer enabled	[63:0] (words 0-3)

Related Information

Timing Constraints for Bonded PCS and PMA Channels on page 18-11

Standard PCS Interface Ports

This section describes the PCS interface.

Figure 15-6: Standard PCS Interfaces

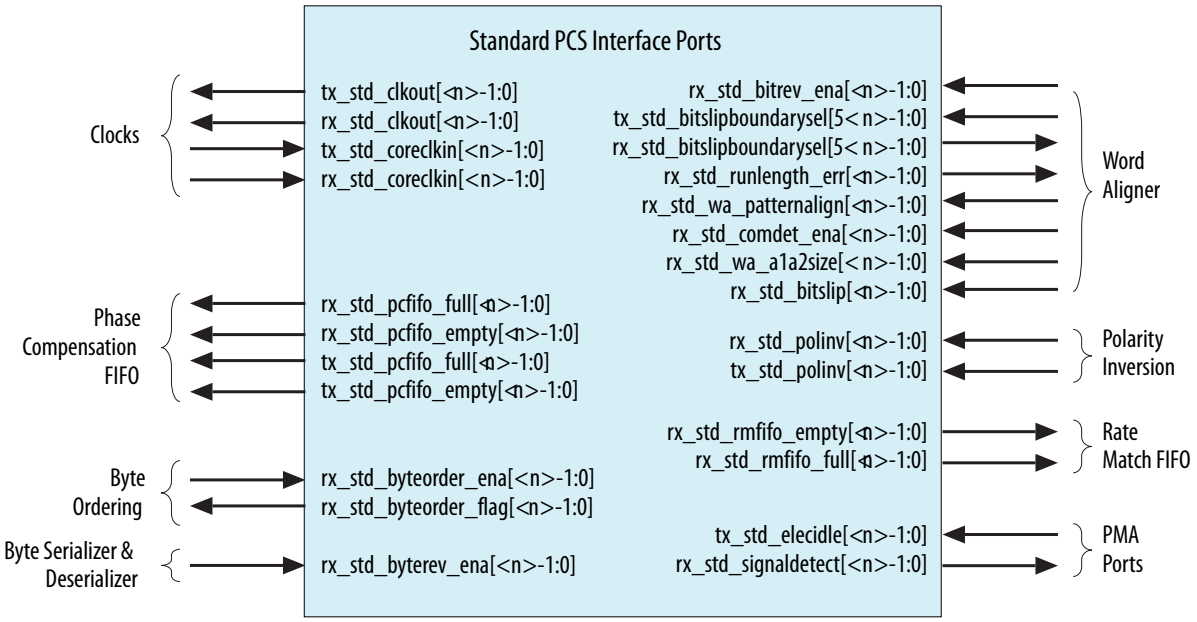


Table 15-43: Standard PCS Interface Ports

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
Clocks			
tx_std_clkout[<n>-1:0]	Output	—	TX Parallel clock output.
rx_std_clkout[<n>-1:0]	Output	—	RX parallel clock output. The CDR circuitry recovers RX parallel clock from the RX data stream.
tx_std_coreclkin[<n>-1:0]	Input	—	TX parallel clock input from the FPGA fabric that drives the write side of the TX phase compensation FIFO.
rx_std_coreclkin[<n>-1:0]	Input	—	RX parallel clock that drives the read side of the RX phase compensation FIFO.
Phase Compensation FIFO			
rx_std_pcfifo_full[<n>-1:0]	Output	Yes	RX phase compensation FIFO full status flag.

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
rx_std_pcfifo_empty[<n>-1:0]	Output	Yes	RX phase compensation FIFO status empty flag.
tx_std_pcfifo_full[<n>-1:0]	Output	Yes	TX phase compensation FIFO status full flag.
tx_std_pcfifo_empty[<n>-1:0]	Output	Yes	TX phase compensation FIFO status empty flag.
Byte Ordering			
rx_std_byteorder_ena[<n>-1:0]	Input	No	Byte ordering enable. When this signal is asserted, the byte ordering block initiates a byte ordering operation if the **Byte ordering control mode** is set to **manual**. Once byte ordering has occurred, you must deassert and reassert this signal to perform another byte ordering operation. This signal is an synchronous input signal; however, it must be asserted for at least 1 cycle of rx_std_clkout.
rx_std_byteorder_flag[<n>-1:0]	Output	Yes	Byte ordering status flag. When asserted, indicates that the byte ordering block has performed a byte order operation. This signal is asserted on the clock cycle in which byte ordering occurred. This signal is synchronous to the rx_std_clkout clock. You must a synchronizer this signal.
Byte Serializer and Deserializer			
rx_std_byterev_ena[<n>-1:0]	Input	No	This control signal is available in when the PMA width is 16 or 20 bits. When asserted, enables byte reversal on the RX interface.
8B/10B			

Name	Dir	Synchronous to tx_std_coreclkin/rx_std_coreclkin	Description
rx_std_polinv[<n>-1:0]	Input	No	Polarity inversion for the 8B/10B decoder, When set, the RX channels invert the polarity of the received data. You can use this signal to correct the polarity of differential pairs if the transmission circuitry or board layout mistakenly swapped the positive and negative signals. The polarity inversion function operates on the word aligner input.
tx_std_polinv[<n>-1:0]	Input	No	Polarity inversion, part of 8B10B encoder, When set, the TX interface inverts the polarity of the TX data.
Rate Match FIFO			
rx_std_rmfifo_empty[<n>-1:0]	Output	No	Rate match FIFO empty flag. When asserted, the rate match FIFO is empty. This port is only used for XAUI, GigE, and Serial RapidIO in double width mode. In double width mode, the FPGA data width is twice the PCS data width to allow the fabric to run at half the PCS frequency
rx_std_rmfifo_full[<n>-1:0]	Output	No	Rate match FIFO full flag. When asserted the rate match FIFO is full. You must synchronize this signal. This port is only used for XAUI, GigE, and Serial RapidIO in double width mode.
Word Aligner			
rx_std_bitrev_ena[<n>-1:0]	Input	No	When asserted, enables bit reversal on the RX interface. Bit order may be reversed if external transmission circuitry transmits the most significant bit first. When enabled, the receive circuitry receives all words in the reverse order. The bit reversal circuitry operates on the output of the word aligner.

Name	Dir	Synchronous to tx_std_coreclkin/rx_std_coreclkin	Description
tx_std_bitslipboundarysel[5<n>-1:0]	Input	No	BitSlip boundary selection signal. Specifies the number of bits that the TX bit slipper must slip.
rx_std_bitslipboundarysel[5<n>-1:0]	Output	No	This signal operates when the word aligner is in bitslip word alignment mode. It reports the number of bits that the RX block slipped to achieve deterministic latency.
rx_std_runlength_err[<n>-1:0]	Output	No	When asserted, indicates a run length violation. Asserted if the number of consecutive 1s or 0s exceeds the number specified in the parameter editor GUI.
rx_st_wa_patternalign	Input	No	Active when you place the word aligner in manual mode. In manual mode, you align words by asserting rx_st_wa_patternalign. rx_st_wa_patternalign is edge sensitive. For more information refer to the *Word Aligner* section in the *Transceiver Architecture in Arria V Devices*.
rx_std_wa_a1a2size[<n>-1:0]	Input	No	Used for the SONET protocol. Assert when the A1 and A2 framing bytes must be detected. A1 and A2 are SONET backplane bytes and are only used when the PMA data width is 8 bits.
rx_std_bitslip[<n>-1:0]	Input	No	Used when word aligner mode is bitslip mode. For every rising edge of the rx_std_bitslip signal, the word boundary is shifted by 1 bit. Each bitslip removes the earliest received bit from the received data. This is an asynchronous input signal and inside there is a synchronizer to synchronize it with rx_pma_clk/rx_clkout.
Miscellaneous			

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
tx_std_elecidle[<n>-1:0]	Input		When asserted, enables a circuit to detect a downstream receiver. This signal must be driven low when not in use because it causes the TX PMA to enter electrical idle mode with the TX serial data signals in tristate mode.
rx_std_signaldetect[<n>-1:0]	Output	No	Signal threshold detect indicator. When asserted, it indicates that the signal present at the receiver input buffer is above the programmed signal detection threshold value. You must synchronize this signal.

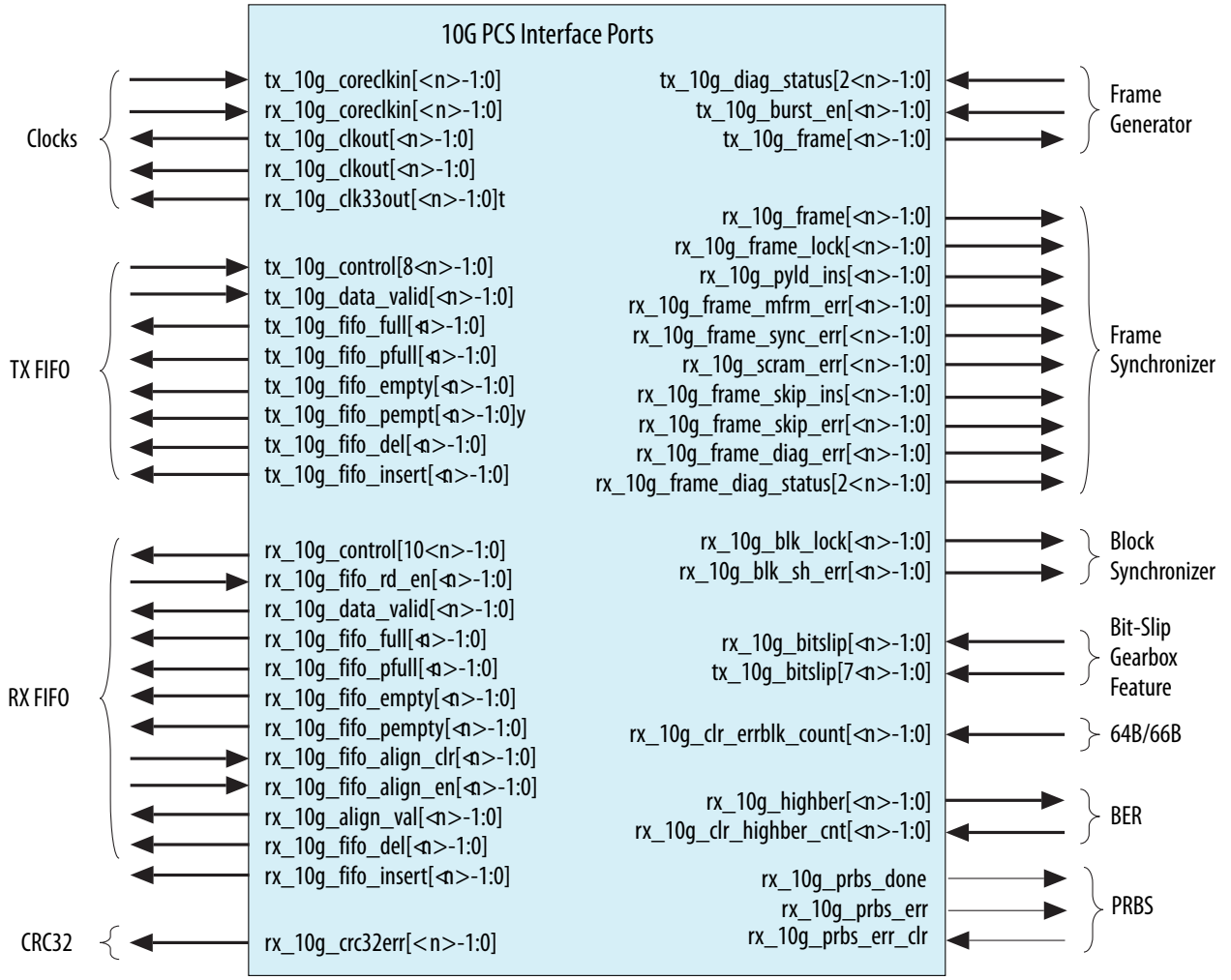
Related Information

Transceiver Architecture in Arria V Devices

10G PCS Interface

The following figure illustrates the top-level signals of the 10G PCS. If you enable both the 10G PCS and Standard PCS your top-level HDL file includes all the interfaces for both.

Figure 15-7: Arria V Native PHY 10G PCS Interfaces



The following table describes the signals available for the 10G PCS datapath. When you enable both the 10G and Standard datapaths, both sets of signals are included in the top-level HDL file for the Native PHY.

Note: In the following table, the column labeled *"Synchronous to tx_10_coreclkin/rx_10g_coreclkin"* refers to cases where the phase compensation FIFO is not in register mode.

Table 15-44: Name Dir Synchronous to tx_10g_coreclkin/rx_10g_coreclkin Description

Name	Dir	Synchronous to tx_10g_coreclkin/rx_10g_coreclkin	Description
Clocks			

Name	Dir	Synchronous to tx_10g_coreclkin/ rx_10g_coreclkin	Description
tx_10g_coreclkin [<*n*>-1:0]	Input	—	TX parallel clock input that drive the write side of the TX FIFO.
rx_10g_coreclkin [<*n*>-1:0]	Input	—	RX parallel clock input that drives the read side of the RX FIFO.
tx_10g_clkout [<*n*>-1:0]	Output	—	TX parallel clock output for the TX PCS.
rx_10g_clkout [<*n*>-1:0]	Output	—	RX parallel clock output which is recovered from the RX data stream.
rx_10g_clk33out [<*n*>-1:0]	Output	—	This clock is driven by the RX deserializer. Its frequency is RX CDR PLL clock frequency divided by 33 or equivalently the RX PMA data rate divided by 66. It is typically used for ethernet applications that use 66b/64b decoding.
TX FIFO			
tx_10g_control [9<*n*>-1:0]	Input	Yes	TX control signals for the Interlaken, 10GBASE-R, and Basic protocols. Synchronous to tx_10g_coreclk_in. The following signals are defined: Interlaken mode: • [8]: Active-high synchronous error insertion control bit • [7:3]: Not Used

Name	Dir	Synchronous to tx_10g_coreclkin/rx_10g_coreclkin	Description
`tx_10g_control [9<n>-1:0] (continued)`	Input	Yes	• [2]: Inversion signal, must always be set to 1'b0. • [1]: Sync Header, 1 indicates a control word • [0]: Sync Header, 1 indicates a data word 10G BaseR mode: • [8]: Active-high synchronous error insertion control signal • [7]: MII control signal for `tx_data[63:56]` • [6]: MII control signal for `tx_data[55:48]` • [5]: MII control signal for `tx_data[47:40]` • [4]: MII control signal for `tx_data[39:32]` • [3]: MII control signal for `tx_data[31:24]` • [2]: MII control signal for `tx_data[23:16]` • [1]: MII control signal for `tx_data[15:8]` • [0]: MII control signal for `tx_data[7:0]` Basic mode: 67-bit word width: • [8:3]: Not used • [2]: Inversion Bit - must always be set to 1'b0. • [1]: Sync Header, 1 indicates a control word) • [0]: Sync Header, 1 indicates a data word) Basic mode: 66-bit word width: • [8:2]: Not used • [1]: Sync Header, 1 indicates a control word) • [0]: Sync Header, 1 indicates 1 data word) Basic mode: 64-bit, 50-bit, 40-bit, 32-bit word widths: [8:0]: Not used

Name	Dir	Synchronous to tx_10g_coreclkin/ rx_10g_coreclkin	Description
tx_10g_data_valid [<*n*>-1:0]	Input	Yes	When asserted, indicates if tx_data is valid Use of this signal depends upon the protocol you are implementing, as follows: • 10G BASE-R: Tie to 1'b1 • Interlaken: Acts as control for FIFO write enable. You should tie this signal to tx_10g_fifo_pempty. • Basic with phase compensation FIFO: Tie to 1'b1 as long as tx_coreclkin = data_rate/ pld_pcs interface width. Otherwise, tie this signal to tx_10g_fifo_pempty. • Basic with phase compensation FIFO in register mode. This mode only allows a 1:1 gear box ratio such as 32:32 and 64:64; consequently, you can tie tx_10g_data_valid to 1'b1.
tx_10g_fifo_full [<*n*>-1:0]	Output	Yes	When asserted, indicates that the TX FIFO is full. Synchronous to tx_std_clkout,
tx_10g_fifo_pfull [<*n*>-1:0]	Output	Yes	When asserted, indicates that the TX FIFO is partially full.
tx_10g_fifo_empty [<*n*>-1:0]	Output	No	TX FIFO empty flag. Synchronous to tx_std_clkout. This signal is pulse-stretched; you must use a synchronizer.
tx_10g_fifo_pempty [<*n*>-1:0]	Output	No	TX FIFO partially empty flag. Synchronous to tx_std_clkout. This signal is pulse-stretched; you must use a synchronizer.
tx_10g_fifo_del [<*n*>-1:0]	Output	Yes	When asserted, indicates that a word has been deleted from the rate match FIFO. This signal is used for the 10GBASE-R protocol.
tx_10g_fifo_insert [<*n*>-1:0]	Output	No	When asserted, indicates that a word has been inserted into the rate match FIFO. This signal is used for the 10GBASE-R protocol. This signal is pulse-stretched, you must use a synchronizer.
RX FIFO			

Name	Dir	Synchronous to tx_10g_coreclkin/ rx_10g_coreclkin	Description
`rx_10g_control [10<n>-1:0]`	Output	Yes	RX control signals for the Interlaken, 10GBASE-R, and Basic protocols. The following signals are defined: Interlaken mode: • [9]: Active-high synchronous status signal that indicates when block lock and frame lock are achieved. • [8]: Active-high synchronous status signal that indicates a synchronization header, metaframe or CRC32 error. • [7]: Active-high synchronous status signal that indicates the Diagnostic Word location within a metaframe. • [6]: Active-high synchronous status signal that indicates the SKIP Word location within a metaframe. • [5]: Active-high synchronous status signal that indicates the Scrambler State Word location within a metaframe. • [4]: Active-high synchronous status signal that indicates the Synchronization Word location within a metaframe. • [3]: Active-high synchronous status signal that indicates the Payload Word location within a metaframe. • [2]: Inversion signal, when asserted indicates that the polarity of the signal has been inverted. • [1]: Synchronization header, 1 indicates control word. • [0]: Synchronization header, 1 indicates data word. 10GBASE-R mode: • [9]: Active-high synchronous status signal indicating when Block Lock is achieved • [8]: Active-high status signal that indicates a Idle/OS deletion • [7]: MII control signal for `tx_data[63:56]` • [6]: MII control signal for `tx_data[55:48]` • [5]: MII control signal for `tx_data[47:40]` • [4]: MII control signal for `tx_data[39:32]` • [3]: MII control signal for `tx_data[31:24]` • [2]: MII control signal for `tx_data[23:16]` • [1]: MII control signal for `tx_data[15:8]` • [0]: MII control signal for `tx_data[7:0]`

Name	Dir	Synchronous to tx_10g_coreclkin/ rx_10g_coreclkin	Description
rx_10g_control [10<*n*>-1:0] (continued)			Basic mode: 67-bit mode with Block Sync: • [9]: Active-high synchronous status signal that indicates when Block Lock is achieved. • [8]: Active-high synchronous status signal that indicates a sync header error • [7:3]: Not used [2]: Used • [1]: Synchronization header, a 1 indicates control word • [0]: Synchronization header, a 1 indicates data word Basic mode: 66-bit mode with Block Sync: [9]: Active-high synchronous status signal that indicates when Block Lock is achieved. [8]: Active-high synchronous status signal that indicates a sync header error. [7:2]: Not used • [1]: Synchronization header, a 1 indicates control word • [0]: Synchronization header, a 1 indicates data word Basic mode: 67-bit mode without Block Sync: [9:3]: Not used 66-bit mode without Block Sync: [9:2]: Not used • [1]: Synchronization header, a 1 indicates control word • [0]: Synchronization header, a 1 indicates data word Basic mode: 64-bit, 50-bit, 40-bit and 32-bit modes: [9:0]: Not used
rx_10g_fifo_rd_en [<*n*>-1:0]	Input	Yes	Active high read enable signal for RX FIFO. Asserting this signal reads 1 word from the RX FIFO.

Name	Dir	Synchronous to tx_10g_coreclkin/ rx_10g_coreclkin	Description
rx_10g_data_valid [<*n*>-1:0]	Output	Yes	Active valid data signal with the following use: • 10GBASE-R: Always high • Interlaken: Toggles indicating when rx_data is valid. • Basic - Phase compensation: Toggles indicating when rx_data is valid. • Basic - Register: Toggles indicating when rx_data is valid.
rx_10g_fifo_full [<*n*>-1:0]	Output	No	Active high RX FIFO full flag. Synchronous to rx_10g_clkout. This signal is pulse-stretched; you must use a synchronizer.
rx_10g_fifo_pfull [<*n*>-1:0]	Output	No	RX FIFO partially full flag. Synchronous to rx_10g_clkout. This signal is pulse-stretched; you must use a synchronizer.
rx_10g_fifo_empty [<*n*>-1:0]	Output	Yes	Active high RX FIFO empty flag,
rx_10g_fifo_pempty [<*n*>-1:0]	Output	Yes	Active high. RX FIFO partially empty flag,
rx_10g_fifo_align_clr [<*n*>-1:0]	Input	Yes	For the Interlaken protocol, this signal clears the current word alignment when the RX FIFO acts as a deskew FIFO. When it is asserted, the RX FIFO is reset and searches for a new alignment pattern.
rx_10g_fifo_align_en [<*n*>-1:0]	Input	Yes	For the Interlaken protocol, you must assert this signal to enable the RX FIFO for alignment.
rx_10g_align_val [<*n*>-1:0]	Output	Yes	For the Interlaken protocol, an active high indication that the alignment pattern has been found
Rx_10g_fifo_del [<*n*>-1:0]	Output	No	When asserted, indicates that a word has been deleted from the TX FIFO. This signal is used for the 10GBASE-R protocol. This signal is pulse-stretched; you must use a synchronizer.

Name	Dir	Synchronous to tx_10g_coreclkin/ rx_10g_coreclkin	Description
Rx_10g_fifo_insert [<*n*>-1:0]	Output	Yes	Active-high 10G BaseR RX FIFO insertion flag When asserted, indicates that a word has been inserted into the TX FIFO. This signal is used for the 10GBASE-R protocol.
CRC32			
rx_10g_crc32err [<*n*>-1:0]	Output	No	For the Interlaken protocol, asserted to indicate that the CRC32 Checker has found a CRC32 error in the current metaframe. Is is asserted at the end of current metaframe. This signal is pulse-stretched; you must use a synchronizer.
Frame Generator			
tx_10g_diag_status [*2*<*n*>-1:0]	Input	No	For the Interlaken protocol, provides diagnostic status information reflecting the lane status message contained in the Framing Layer Diagnostic Word (bits[33:32]). This message is inserted into the next Diagnostic Word generated by the Frame Generation Block. The message must be held static for 5 cycles before and 5 cycles after the tx_frame pulse.
tx_10g_burst_en [<*n*>-1:0]	Input	No	For the Interlaken protocol, controls frame generator reads from the TX FIFO. Latched once at the beginning of each metaframe.When 0, the frame generator inserts SKIPs. When 1, the frame generator reads data from the TX FIFO. Must be held static for 5 cycles before and 5 cycles after the tx_frame pulse.
tx_10g_frame [<*n*>-1:0]	Output	No	For the Interlaken protocol, asserted to indicate the beginning of a new metaframe inside the frame generator. This signal is pulse-stretched; you must use a synchronizer.
Frame Synchronizer			
rx_10g_frame [<*n*>-1:0]	Output	No	For the Interlaken protocol, asserted to indicate the beginning of a new metaframe inside the frame synchronizer. This signal is pulse-stretched, you must use a synchronizer. This signal is pulse-stretched; you must use a synchronizer.

Name	Dir	Synchronous to tx_10g_coreclkin/rx_10g_coreclkin	Description
rx_10g_frame_lock [<*n*>-1:0]	Output	No	For the Interlaken protocol, asserted to indicate that the frame synchronizer state machine has achieved frame lock. This signal is pulse-stretched, you must use a synchronizer. This signal is pulse-stretched; you must use a synchronizer.
Rx_10g_pyld_ins [<*n*>-1:0]	Output	No	For the Interlaken protocol, asserted to indicate a SKIP Word was not received by the frame synchronizer in a SKIP Word location within the metaframe. This signal is pulse-stretched, you must use a synchronizer. This signal is pulse-stretched; you must use a synchronizer.
rx_10g_frame_mfrm_err [<*n*>-1:0]	Output	No	For the Interlaken protocol, asserted to indicate an error has occurred in the metaframe. This signal is pulse-stretched, you must use a synchronizer. This signal is pulse-stretched; you must use a synchronizer.
rx_10g_frame_sync_err [<*n*>-1:0]	Output	No	For the Interlaken protocol, asserted to indicate a synchronization Control Word error was received in a synchronization Control Word location within the metaframe. This signal is sticky if block lock is lost and does not update until block lock is re-established.This signal is pulse-stretched; you must use a synchronizer.
rx_10g_scram_err [<*n*>-1:0]	Output	No	For the Interlaken protocol, asserted to indicate, Scrambler Control Word errors in a Scrambler Control Word location within the metaframe. This signal is sticky during the loss of block lock and does not update until block lock is re-established. This signal is pulse-stretched; you must use a synchronizer.
rx_10g_frame_skip_ins [<*n*>-1:0]	Output	No	For the Interlaken protocol, asserted to indicate to a SKIP Word was received by the frame synchronizer in a non-SKIP Word location within the metaframe. This signal is pulse-stretched; you must use a synchronizer.

Name	Dir	Synchronous to tx_10g_coreclkin/ rx_10g_coreclkin	Description
rx_10g_frame_skip_err [<*n*>-1:0]	Output	No	For the Interlaken protocol, asserted to indicate a Skip Control Word error was received in a Skip Control Word location within the metaframe. This signal is sticky during the loss of block lock and does not update until block lock is re-established. This signal is pulse-stretched; you must use a synchronizer.
rx_10g_frame_diag_err[<*n*>-1:0]	Output	No	For the Interlaken protocol, asserted to indicate a Diagnostic Control Word error was received in a Diagnostic Control Word location within the metaframe. This signal is sticky during the loss of block lock and does not update until block lock is re-established. This signal is pulse-stretched; you must use a synchronizer.
rx_10g_frame_diag_status [*2*<*n*>-1:0]	Output	No	For the Interlaken protocol, reflects the lane status message contained in the framing layer Diagnostic Word (bits[33:32]). This information is latched when a valid Diagnostic Word is received in a Diagnostic Word Metaframe location. This signal is pulse-stretched; you must use a synchronizer.
Block Synchronizer			
rx_10g_blk_lock [<*n*>-1:0]	Output	No	Active-high status signal that is asserted when block synchronizer acquires block lock. Valid for the 10GBASE-R and Interlaken protocols, and any basic mode that uses the lock state machine to achieve and monitor block synchronization for word alignment. Once the block synchronizer acquires block lock, it takes at least 16 errors for rx_10g_blk_lock to be deasserted.
rx_10g_blk_sh_err [<*n*>-1:0]	Output	No	Error status signal from block synchronizer indicating an invalid synchronization header has been received. Valid for the 10GBASE-R and Interlaken protocols, and any legal basic mode that uses the lock state machine to achieve and monitor block synchronization for word alignment. Active only after block lock is achieved. This signal is generated by rx_pma_clk and is pulse-stretched by 3 clock cycles. You must use a synchronizer.

Name	Dir	Synchronous to tx_10g_coreclkin/ rx_10g_coreclkin	Description
Bit-Slip Gearbox Feature Synchronizer			
rx_10g_bitslip [<*n*>-1:0]	Input	No	User control bit-slip in the RX Gearbox. Slips one bit per rising edge pulse.
tx_10g_bitslip [7<*n*>-1:0]	Input	No	TX bit-slip is controlled by tx_bitslip port. Shifts the number of bit location specified by tx_bitslip. The maximum shift is <*pcswidth-1*>.
64b/66b			
rx_10g_clr_errblk_count [<*n*>-1:0]	Input	No	For the 10GBASE-R protocol, asserted to clear the error block counter which counts the number of times the RX state machine enters the RX error state.
BER			
rx_10g_highber [<*n*>-1:0]	Output	No	For the 10GBASE-R protocol, status signal asserted to indicate a bit error ratio of $>10^{-4}$. A count of 16 in 125us indicates a bit error ratio of $>10^{-4}$. Once asserted, it remains high for at least 125 us.
rx_10g_clr_highber_cnt [<*n*>-1:0]	Input	No	For the 10GBASE-R protocol, status signal asserted to clear the BER counter which counts the number of times the BER state machine enters the BER_BAD_SH state. This signal has no effect on the operation of the BER state machine.
PRBS			
rx_10g_prbs_done	Output		When asserted, indicates the verifier has aligned and captured consecutive PRBS patterns and the first pass through a polynomial is complete.
rx_10g_prbs_err	Output		When asserted, indicates an error only after the rx_10g_prbs_done signal has been asserted. This signal pulses for every error that occurs. An error can only occur once per word. This signal indicates errors for both the PRBS and pseudo-random patterns.
rx_10g_prbs_err_clr	Input		When asserted, clears the PRBS pattern and de-asserts the rx_10g_prbs_done signal.

SDC Timing Constraints of Arria V GZ Native PHY

This section describes SDC examples and approaches to identify false timing paths.

The Intel Quartus Prime software reports timing violations for asynchronous inputs to the Standard PCS and 10G PCS. Because many violations are for asynchronous paths, they do not represent actual timing failures. You may choose one of the following three approaches to identify these false timing paths to the Intel Quartus Prime or TimeQuest software.

In all of these examples, you must substitute you actual signal names for the signal names shown.

Example 15-1: Using the set_false_path Constraint to Identify Asynchronous Inputs

You can cut these paths in your Synopsys Design Constraints (.sdc) file by using the set_false_path command as shown in following example.

```
set_false_path -through {*10gtxbursten*} -to [get_registers
*10g_tx_pcs*SYNC_DATA_REG*]

set_false_path -through {*10gtxdiagstatus*} -to [get_registers
*10g_tx_pcs*SYNC_DATA_REG*]

set_false_path -through {*10gtxwordslip*} -to [get_registers
*10g_tx_pcs*SYNC_DATA_REG*]

set_false_path -through {*10gtxbitslip*} -to [get_registers
*10g_tx_pcs*SYNC_DATA_REG*]

set_false_path -through {*10grxbitslip*} -to [get_registers
*10g_rx_pcs*SYNC_DATA_REG*]

set_false_path -through {*10grxclrbercount*} -to [get_registers
*10g_rx_pcs*SYNC_DATA_REG*]

set_false_path -through {*10grxclrerrblkcnt*} -to [get_registers
*10g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*10grxprbserrclr*} -to [get_registers
*10g_rx_pcs*SYNC_DATA_REG*]

set_false_path -through {*8gbitslip*} -to  [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*8gbytordpld*} -to [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*8gcmpfifoburst*} -to [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*8gphfifoburstrx*} -to [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]

set_false_path -through {*8gsyncsmen*} -to [get_registers
*8g*pcs*SYNC_DATA_REG*]
set_false_path -through {*8gwrdisablerx*} -to [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*rxpolarity*} -to [get_registers *SYNC_DATA_REG*]
set_false_path -through {*pldeidleinfersel*} -to [get_registers
*SYNC_DATA_REG*]
```

Example 15-2: Using the max_delay Constraint to Identify Asynchronous Inputs

You can use the set_max_delay constraint on a given path to create a constraint for asynchronous signals that do not have a specific clock relationship but require a maximum path delay. The following example illustrates this approach.

```
# Example: Apply 10ns max delay

set_max_delay -from *tx_from_fifo* -to *8g*pcs*SYNC_DATA_REG1 10
```

Example 15-3: Using the set_false TimeQuest Constraint to Identify Asynchronous Inputs

You can use the set_false path command only during Timequest timing analysis. The following example illustrates this approach.

```
#if {$::TimeQuestInfo(nameofexecutable) eq "quartus_fit"} {

#} else {

#set_false_path -from [get_registers {*tx_from_fifo*}] -through
{*txbursten*} -to [get_registers *8g_*_pcs*SYNC_DATA_REG
```

Dynamic Reconfiguration for Arria V GZ Native PHY

Dynamic reconfiguration calibrates each channel to compensate for variations due to process, voltage, and temperature (PVT).

As silicon progresses towards smaller process nodes, circuit performance is affected more by variations due to process, voltage, and temperature (PVT). These process variations result in analog voltages that can be offset from required ranges. The calibration performed by the dynamic reconfiguration interface compensates for variations due to PVT.

For more information about transceiver reconfiguration refer to Chapter 16, Transceiver Reconfiguration Controller IP Core.

Example 15-4: Informational Messages for the Transceiver Reconfiguration Interface

For non-bonded clocks, each channel and each TX PLL has a separate dynamic reconfiguration interfaces. The MegaWizard Plug-In Manager provides informational messages on the connectivity of these interfaces. The following example shows the messages for the Arria V GZ Native PHY with four duplex channels, four TX PLLs, in a non-bonded configuration.

```
PHY IP will require 8 reconfiguration interfaces for connection to the
external reconfiguration controller.
Reconfiguration interface offsets 0-3 are connected to the transceiver
channels.
Reconfiguration interface offsets 4–7 are connected to the transmit PLLs.
```

Example 15-5: Overriding Logical Channel 0 Channel Assignment Restrictions in Arria V GZ Device for ×6 or ×N Bonding

If you are using ×6 or ×N bonding, transceiver dynamic reconfiguration requires that you assign the starting channel number. Logical channel 0 should be assigned to either physical transceiver channel 1 or channel 4 of a transceiver bank. However, if you have already created a PCB with a different lane assignment for logical lane 0, you can use the workaound shown in the following example to remove this restriction. The following example redefines the pma_bonding_master parameter using the Intel Quartus Prime Assignment Editor. In this example, the pma_bonding_master was originally assigned to physical channel 1. (The original assignment could also have been to physical channel 4.) The to parameter reassigns the pma_bonding_master to the Deterministic Latency PHY instance name. You must substitute the instance name from your design for the instance name shown in quotation marks

```
set_parameter -name pma_bonding_master "\"1\"" -to "<PHY IP instance name>"
```

Simulation Support

The Intel Quartus Prime release provides simulation and compilation support for the Arria V GZ Native PHY IP Core. Refer to Running a Simulation Testbench for a description of the directories and files that the Intel Quartus Prime software creates automatically when you generate your Arria V GZ Transceiver Native PHY IP Core.

Slew Rate Settings

The following transceiver slew rate settings are allowed in Intel Quartus Prime software.

Table 15-45: Slew Rate Settings for Arria V GZ devices

Protocol / Datarate	Allowed Intel Quartus Prime Settings	IBIS-AMI Settings
PCI Express Gen3, Gen2, CEI	4	*_30ps
PCI Express Gen1, XAUI	3	*_50ps
Gigabit Ethernet	1	*_160ps
SATA_1500 sub-protocol	3	*_50ps
SATA_3000 sub-protocol	3	*_50ps
SATA_6000 sub-protocol	4	*_30ps
=<1 Gbps	1, 2, 3	*_160ps, *_90ps, *_50ps
1 Gbps - 3 Gbps	2, 3	*_90ps, *_50ps
3 Gbps - 6 Gbps	3, 4	*_50ps, *_30ps
>6 Gbps	5	*_15ps

Assigning an invalid slew rate will result in an error message similar to the one below:

Error (15001): Assignment XCVR_TX_SLEW_RATE_CTRL of value "4" conflicts with the valid parameter values for pm_tx_slew_rate_ctrl

- Protocol declarations take priority over datarate. For example, XAUI has a per-lane datarate of 3.125 Gbps, but only a setting of "3" is allowed. A setting of "4" is not allowed for XAUI.
- For protocols not listed in the table, you should use the slew settings associated with your datarate.
- The IBIS-AMI slew rate figure is defined as the approximate transmitter 20% - 80% rise time. The "ps" figure should not be considered quantitative and is an approximate label only.
- The IBIS-AMI models will allow you to simulate any slew rate setting for any datarate or protocol.

Cyclone V Transceiver Native PHY IP Core Overview 16

2020.06.02

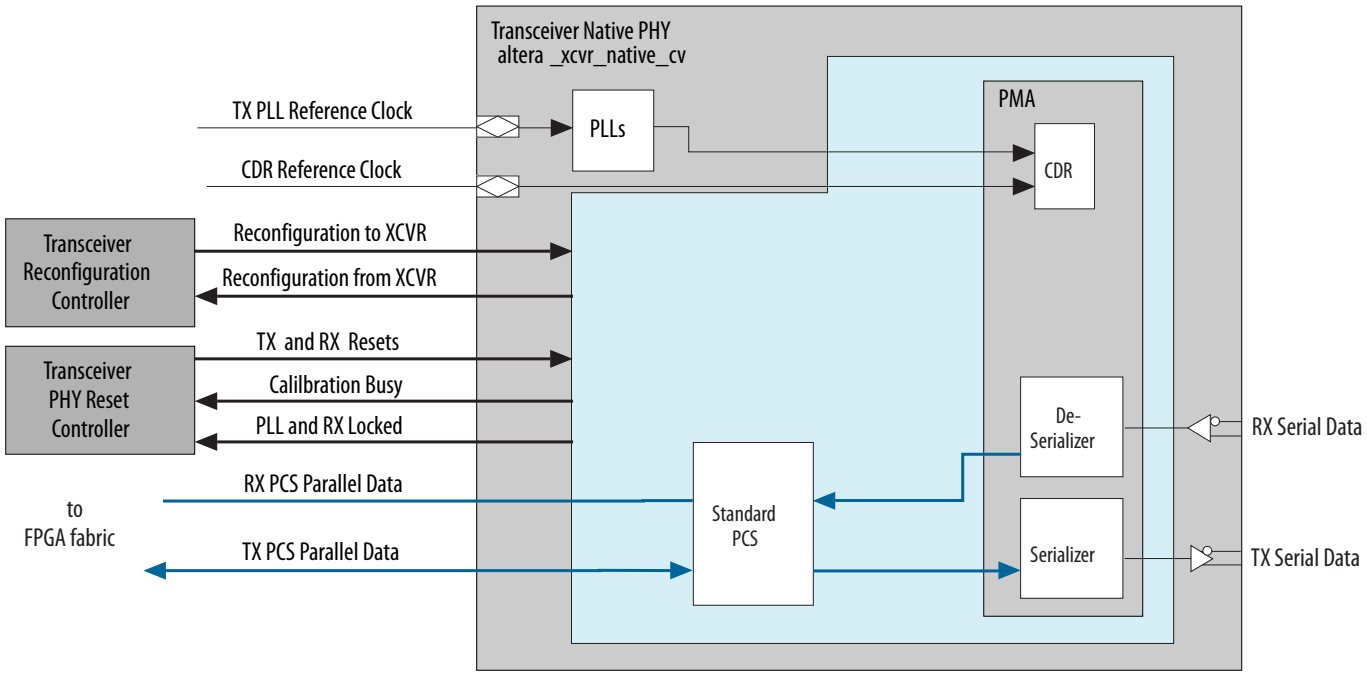
UG-01080 Subscribe Send Feedback

The Cyclone V Transceiver Native PHY IP Core provides direct access to all control and status signals of the transceiver channels. Unlike other PHY IP Cores, the Native PHY IP Core does not include an Avalon Memory-Mapped (Avalon-MM) interface. Instead, it exposes all signals directly as ports. The Cyclone V Transceiver Native PHY IP Core includes the Standard PCS. You can select the PCS functions and control and status port that your transceiver PHY requires.

The Native Transceiver PHY does not include an embedded reset controller. You can either design custom reset logic or incorporate Altera's "Transceiver PHY Reset Controller IP Core" to implement reset functionality.

As the following figure illustrates, TX PLL and clock data recovery (CDR) reference clocks from the pins of the device are input to the PLL module and CDR logic. The Standard PCS drives TX parallel data and receives RX parallel data.

Figure 16-1: Cyclone Native Transceiver PHY IP Core



In a typical design, the separately instantiated Transceiver PHY Reset Controller drives reset signals to Native PHY and receives calibration and locked status signal from the Native PHY. The Native PHY reconfiguration buses connect the external Transceiver Reconfiguration Controller for calibration and dynamic reconfiguration of the channel and PLLs.

You specify the initial configuration when you parameterize the IP core. The Transceiver Native PHY IP Core connects to the "Transceiver Reconfiguration Controller IP Core" to dynamically change reference clocks, PLL connectivity, and the channel configurations at runtime.

Related Information

- **Transceiver PHY Reset Controller IP Core** on page 18-1
- **Transceiver Reconfiguration Controller IP Core Overview** on page 17-1

Cyclone Device Family Support

IP cores provide either final or preliminary support for target Altera device families. These terms have the following definitions:

- *Final support*—Verified with final timing models for this device.
- *Preliminary support*—Verified with preliminary timing models for this device.

Table 16-1: Device Family Support

Device Family	Support
Cyclone V devices	Final
Other device families	No support

Cyclone V Native PHY Performance and Resource Utilization

Because the Standard PCS and PMA are implemented in hard logic, the Cyclone V Native PHY IP Core requires minimal resources.

Parameterizing the Cyclone V Native PHY

Complete the following steps to configure the Cyclone V Native PHY IP Core in:

1. Under **Tools** > **IP Catalog**, select **Arria V** as the device family.
2. Under **Tools** > **IP Catalog** > **Interface Protocols** > **Transceiver PHY**, select **Cyclone V Native PHY**.
3. Use the tabs on the MegaWizard Plug-In Manager to select the options required for the protocol.
4. Click **Finish** to generate your customized Cyclone V Native PHY IP Core.

Note: The Cyclone V Transceiver Native PHY provides presets for CPRI, GIGE, and the Low Latency Standard PCS. The presets specify the parameters required to the protocol specified.

General Parameters

This section lists the parameters available on the **General Options** tab.

Table 16-2: General and Datapath Options

Name	Range	Description
Device speed grade	**fastest**	Specifies the speed grade.
Message level for rule violations	**error** **warning**	Allows you to specify the message level, as follows: • **error**: Intel Quartus Prime checker will not create an instance with invalid parameters. You must change incompatible parameter selections to proceed. • **warning**: Intel Quartus Prime checker will allow instance creation with invalid parameters, but the instance will not compile successfully.
Datapath Options		
Enable TX datapath	**On/Off**	When you turn this option **On**, the core includes the TX datapath.
Enable RX datapath	**On/Off**	When you turn this option **On**, the core includes the RX datapath.
Initial PCS datapath selection	**Standard**	The Cyclone V Native PHY only supports the **Standard** datapath.
Number of data channels	1-36	Specifies the total number of data channels in each direction.

Name	Range	Description
Bonding mode	**Non-bonded or x1** **Bonded or xN**	In **Non-bonded or x1** mode, each channel is assigned a PLL. If one PLL drives multiple channels, PLL merging is required. During compilation, the Intel Quartus Prime Fitter, merges all the PLLs that meet PLL merging requirements. Refer to **Merging TX PLLs In Multiple Transceiver PHY Instances** on page 17-58 to observe PLL merging rules. Select **Bonded or xN** to use the same clock source for up to 6 channels in a single transceiver bank or the same clock source for all the transceivers on one side of the device. ×N bonding results in reduced clock skew. You must use contiguous channels when you select ×N bonding. For more information about the clock architecture of bonding, refer to "Transmitter Clock Network" in *Transceiver Clocking in CycloneV Devices* chapter of the *Cyclone V Device Handbook*.
Enable simplified data interface	**On/Off**	When you turn this option **On**, the data interface provides only the relevant interface to the FPGA fabric for the selected configuration. You can only use this option for static configurations. When you turn this option **Off**, the data interface provides the full physical interface to the fabric. Select this option if you plan to use dynamic reconfiguration that includes changing the interface to the FPGA fabric. Refer to "Active Bits for Each Fabric Interface Width" for guidance.

Related Information

Transceiver Clocking in Cyclone V Devices

PMA Parameters

This section describes the options available for the PMA.

For more information about the PMA, refer to the *PMA Architecture* section in the *Transceiver Architecture in Cyclone V Devices*. Some parameters have ranges where the value is specified as Device Dependent. For such parameters, the possible range of frequencies and bandwidths depends on the device, speed

grade, and other design characteristics. Refer to *Device Datasheet for Cyclone V Devices* for specific data for Cyclone V devices.

Table 16-3: PMA Options

Parameter	Range	Description
Data rate	Device Dependent	Specifies the data rate. The maximum data rate is 6.144 Gbps.
TX local clock division factor	**1, 2, 4, 8**	Specifies the value of the divider available in the transceiver channels to divide the input clock to generate the correct frequencies for the parallel and serial clocks. This divisor divides the fast clock from the PLL in nonbonded configurations.
PLL base data rate	Device Dependent	Shows the base data rate of the clock input to the TX PLL.The **PLL base data rate** is computed from the **TX local clock division factor** multiplied by the **data rate**. Select a **PLL base data rate** that minimizes the number of PLLs required to generate all the clocks for data transmission. By selecting an appropriate **PLL base data rate**, you can change data rates by changing the **TX local clock division factor** used by the clock generation block.

Related Information

- **Transceiver Architecture in Cyclone V Devices**
- **Device Datasheet for Cyclone V Devices**

TX PMA Parameters

Note: For more information about PLLs in Cyclone V devices, refer to the *Cyclone V PLLs* section in *Clock Networks and PLLs in Cyclone V Devices.*

Table 16-4: TX PMA Parameters

Parameter	Range	Description
Enable TX PLL dynamic reconfiguration	**On/Off**	When you turn this option **On**, you can dynamically reconfigure the PLL. This option is also required to simulate TX PLL reconfiguration. If you turn this option **On**, the Intel Quartus Prime Fitter prevents PLL merging by default; however, you can specify merging using the `XCVR_TX_PLL_RECONFIG_GROUP` QSF assignment.
Use external TX PLL	**On/Off**	When you turn this option **On**, the Native PHY does not include TX PLLs. Instead, the Native PHY includes a input clock port for connection to the fast clock from an external PLL, `ext_pll_clk[<p>-1:0]` that you can connect to external PLLs. Use feature when need to perform TX PLL switching between fractional PLL and a CMU PLL.
Number of TX PLLs	**1–4**	Specifies the number of TX PLLs that can be used to dynamically reconfigure channels to run at multiple data rates. If your design does not require transceiver TX PLL dynamic reconfiguration, set this value to 1. The number of actual physical PLLs that are implemented depends on the selected clock network. Each channel can dynamically select between n PLLs, where n is the number of PLLs specified for this parameter. **Note:** Refer to *Transceiver Clocking in Cyclone V Devices* chapter for more details.
Main TX PLL logical index	**0–3**	Specifies the index of the TX PLL used in the initial configuration.
Number of TX PLL reference clocks	**1–5**	Specifies the total number of reference clocks that are used by all of the PLLs.

Related Information

Cyclone V Device Handbook Volume 2: Transceivers

TX PLL Parameters

This section allows you to define multiple TX PLLs for your Native PHY. The Native PHY GUI provides a separate tab for each TX PLL.

Table 16-5: TX PLL Parameters

Parameter	Range	Description
PLL type	**CMU**	This is the only PLL type available.
PLL base data rate	Device Dependent	Shows the base data rate of the clock input to the TX PLL.The **PLL base data rate** is computed from the **TX local clock division factor** multiplied by the **Data rate**. Select a **PLL base data rate** that minimizes the number of PLLs required to generate all the clocks for data transmission. By selecting an appropriate **PLL base data rate**, you can change data rates by changing the **TX local clock division factor** used by the clock generation block.
Reference clock frequency	Device Dependent	Specifies the frequency of the reference clock for the **Selected reference clock source** index you specify. You can define a single frequency for each PLL. You can use the Transceiver Reconfiguration Controller to dynamically change the reference clock input to the PLL. Note that the list of frequencies updates dynamically when you change the **Data rate**. The Input clock frequency drop down menu is populated with all valid frequencies derived as a function of the **Data rate** and **Base data rate**.
Selected reference clock source	**0–4**	You can define up to 5 reference clock sources for the PLLs in your core. The **Reference clock frequency** selected for index **0**, is assigned to TX PLL<0>. The **Reference clock frequency** selected for index **1**, is assigned to TX PLL<1>, and so on.
Selected clock network	**x1 ×N**	Selects the clock network for the TX PLL. In non-bonded mode, each channel is assigned to one PLL. PLL merging is required when multiple channels are assigned to one PLL. During compilation, the Intel Quartus Prime Fitter, merges all the PLLs that meet PLL merging requirements. Refer to **Merging TX PLLs In Multiple Transceiver PHY Instances** on page 17-58 for more details.

RX PMA Parameters

This section describes the RX PMA options you can specify.

Note: For more information about the CDR circuitry, refer to the Receiver PMA Datapath section in the *Transceiver Architecture in Cyclone V Devices Cyclone V Devices.*

Table 16-6: RX PMA Parameters

Parameter	Range	Description
Enable CDR dynamic reconfiguration	**On/Off**	When you turn this option **On**, you can dynamically change the data rate of the CDR circuit.
Number of CDR reference clocks	**1–5**	Specifies the number of reference clocks for the CDRs.
Selected CDR reference clock	**0–4**	Specifies the index of the selected CDR reference clock.
Selected CDR reference clock frequency	Device Dependent	Specifies the frequency of the clock input to the CDR.
PPM detector threshold	**+/- 1000 PPM**	Specifies the maximum PPM difference the CDR can tolerate between the input reference clock and the recovered clock.
Enable rx_pma_clkout port	**On/Off**	When you turn this option **On**, the RX parallel clock which is recovered from the serial received data is an output of the PMA.
Enable rx_is_lockedtodata port	**On/Off**	When you turn this option **On**, the `rx_is_lockedtodata` port is an output of the PMA.
Enable rx_is_lockedtoref port	**On/Off**	When you turn this option **On**, the `rx_is_lockedtoref` port is an output of the PMA.
Enable rx_set_lockedtodata and rx_set_locktoref ports	**On/Off**	When you turn this option **On**, the `rx_set_lockedtdata` and `rx_set_lockedtoref` ports are outputs of the PMA.
Enable rx_pma_bitslip_port	**On/Off**	When you turn this option **On**, the `rx_pma_bitslip` is an input to the core. The deserializer slips one clock edge each time this signal is asserted. You can use this feature to minimize uncertainty in the serialization process as required by protocols that require a datapath with deterministic latency such as CPRI.

Parameter	Range	Description
Enable rx_seriallpbken port	**On/Off**	When you turn this option **On**, the `rx_seriallpbken` is an input to the core. When your drive a 1 on this input port, the PMA operates in serial loopback mode with TX data looped back to the RX channel.

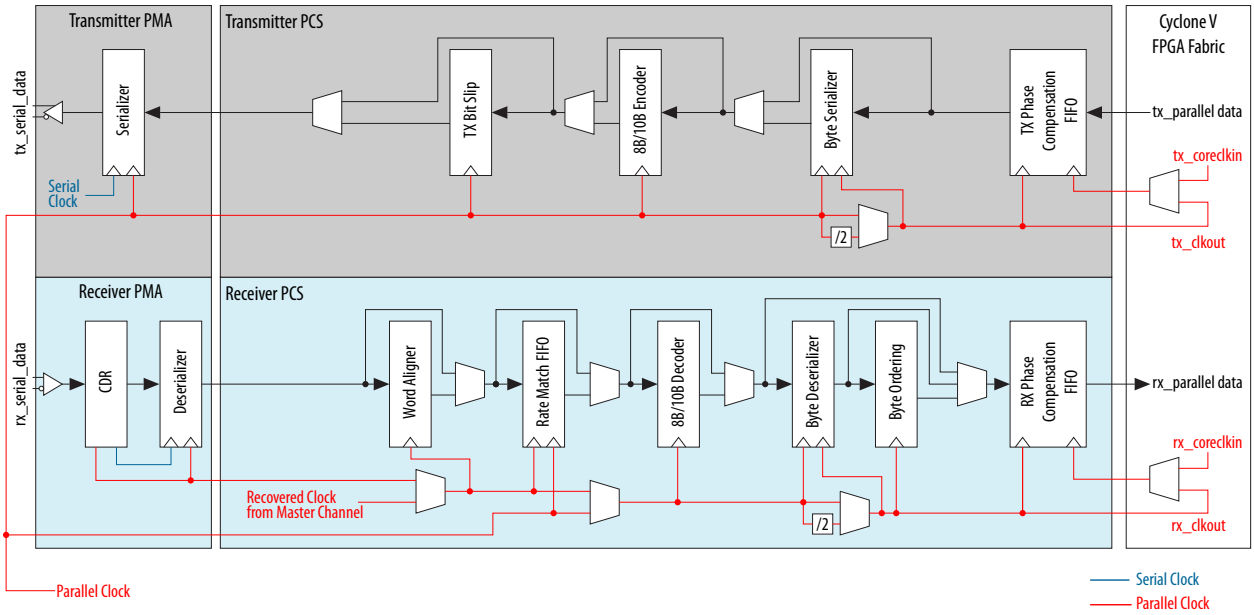
Related Information

Transceiver Architecture in Cyclone V Devices Cyclone V Devices

Standard PCS Parameters

This section illustrates the complete datapath and clocking for the Standard PCS and defines the parameters available to enable or disable the individual blocks in the Standard PCS.

Figure 16-2: The Standard PCS Datapath



Note: For more information about the Standard PCS, refer to the *PCS Architecture* section in the *Transceiver Architecture in Cyclone V Devices*.

The following table describes the general and datapath options for the Standard PCS.

Table 16-7: General and Datapath Parameters

Parameter	Range	Description
Standard PCS protocol mode	**basic** **cpri** **gige**	Specifies the protocol that you intend to implement with the Native PHY. The protocol mode selected guides the MegaWizard in identifying legal settings for the Standard PCS datapath. Use the following guidelines to select a protocol mode: • **basic**–select this mode for when none of the other options are appropriate. You should also select this mode to enable diagnostics, such as loopback. • **cpri**–select this mode if you intend to implement CPRI or another protocol that requires deterministic latency. Altera recommends that you select the appropriate CPRI preset for the CPRI protocol. • **gige**–select this mode if you intend to implement either the 1.25 Gbps or 2.5 Gbps Ethernet protocol. Altera recommends that you select the appropriate preset for the Ethernet protocol.
Standard PCS/PMA interface width	**8, 10, 16, 20**	Specifies the width of the datapath that connects the FPGA fabric to the PMA. The transceiver interface width depends upon whether you enable 8B/10B. To simplify connectivity between the FPGA fabric and PMA, the bus bits used are not contiguous for 16 and 32bit buses. Refer to Active Bits for Each Fabric Interface Width for the bits used.
FPGA fabric/Standard TX PCS interface width	**8, 10, 16, 20, 32, 40**	Shows the FPGA fabric to TX PCS interface width which is calculated from the **Standard PCS/PMA interface width .**
FPGA fabric/Standard RX PCS interface width	**8, 10, 16, 20, 32, 40**	Shows the FPGA fabric to RX PCS interface width which is calculated from the **Standard PCS/PMA interface width .**

Parameter	Range	Description
Enable Standard PCS low latency mode	**On/Off**	When you turn this option **On**, all PCS functions are disabled except for the phase compensation FIFO, byte serializer and byte deserializer. This option creates the lowest latency Native PHY that allows dynamic reconfigure between multiple PCS datapaths.

Phase Compensation FIFO

The phase compensation FIFO assures clean data transfer to and from the FPGA fabric by compensating for the clock phase difference between the low speed parallel clock and FPGA fabric interface clock.

Note: For more information refer to the *Receiver Phase Compensation FIFO* and *Transmitter Phase Compensation FIFO* sections in the *Transceiver Architecture in Cyclone V Devices*.

Table 16-8: Phase Compensation FIFO Parameters

Parameter	Range	Description
TX FIFO mode	**low_latency** **register_fifo**	The following 2 modes are possible: • **low_latency**: This mode adds 3–4 cycles of latency to the TX datapath. • **register_fifo**: In this mode the FIFO is replaced by registers to reduce the latency through the PCS. Use this mode for protocols that require deterministic latency, such as CPRI.
RX FIFO mode	**low_latency** **register_fifo**	The following 2 modes are possible: • **low_latency**: This mode adds 2–3 cycles of latency to the TX datapath. • **register_fifo**: In this mode the FIFO is replaced by registers to reduce the latency through the PCS. Use this mode for protocols that require deterministic latency, such as CPRI.
Enable tx_std_pcfifo_full port	**On/Off**	When you turn this option **On**, the TX Phase compensation FIFO outputs a FIFO full status flag.
Enable tx_std_pcfifo_empty port	**On/Off**	When you turn this option **On**, the TX Phase compensation FIFO outputs a FIFO empty status flag.

Parameter	Range	Description
Enable rx_std_pcfifo_full port	**On/Off**	When you turn this option **On**, the RX Phase compensation FIFO outputs a FIFO full status flag.
Enable rx_std_pcfifo_empty port	**On/Off**	When you turn this option **On**, the RX Phase compensation FIFO outputs a FIFO empty status flag.
Enable rx_std_rmfifo_empty port	**On/Off**	When you turn this option **On**, the rate match FIFO outputs a FIFO empty status flag. The rate match FIFO compensates for small clock frequency differences between the upstream transmitter and the local receiver clocks by inserting or removing skip (SKP) symbols or ordered sets from the interpacket gap (IPG) or idle stream.
Enable rx_std_rmfifo_full port	**On/Off**	When you turn this option **On**, the rate match FIFO outputs a FIFO full status flag.

Related Information

Transceiver Architecture in Cyclone V Devices

Byte Ordering Block Parameters

The RX byte ordering block realigns the data coming from the byte deserializer. This block is necessary when the PCS to FPGA fabric interface width is greater than the PCS datapath.

Because the timing of the RX PCS reset logic is indeterminate, the byte ordering at the output of the byte deserializer may or may not match the original byte ordering of the transmitted data.

Note: For more information refer to the Byte Ordering section in the *Transceiver Architecture in Cyclone V Devices*.

Table 16-9: Byte Ordering Block Parameters

Parameter	Range	Description
Enable RX byte ordering	**On/Off**	When you turn this option **On**, the PCS includes the byte ordering block.

<table>
<tr><th>Parameter</th><th>Range</th><th>Description</th></tr>
<tr><td>Byte ordering control mode</td><td>Manual
Auto</td><td>Specifies the control mode for the byte ordering block. The following modes are available:
Manual: Allows you to control the byte ordering block
Auto: The word aligner automatically controls the byte ordering block once word alignment is achieved.</td></tr>
<tr><td>Byte ordering pattern width</td><td>8–10</td><td>Shows width of the pattern that you must specify. This width depends upon the PCS width and whether or not 8B/10B encoding is used as follows:<table><tr><th>Width</th><th>8B/10B</th><th>Pad Pattern</th></tr><tr><td>8, 16,32</td><td>No</td><td>8 bits</td></tr><tr><td>10, 20, 40</td><td>No</td><td>10 bits</td></tr><tr><td>8, 16, 32</td><td>Yes</td><td>9 bits</td></tr></table></td></tr>
<tr><td>Byte ordering symbol count</td><td>1–2</td><td>Specifies the number of symbols the word aligner should search for. When the PMA is 16 or 20 bits wide, the byte ordering block can optionally search for 1 or 2 symbols.</td></tr>
<tr><td>Byte order pattern (hex)</td><td>User-specified 8-10 bit pattern</td><td>Specifies the search pattern for the byte ordering block.</td></tr>
<tr><td>Byte order pad value (hex)</td><td>User–specified 8-10 bit pattern</td><td>Specifies the pad pattern that is inserted by the byte ordering block. This value is inserted when the byte order pattern is recognized.
The byte ordering pattern should occupy the least significant byte (LSB) of the parallel TX data. If the byte ordering block identifies the programmed byte ordering pattern in the most significant byte (MSB) of the byte-deserialized data, it inserts the appropriate number of user-specified pad bytes to push the byte ordering pattern to the LSB position, restoring proper byte ordering.</td></tr>
<tr><td>Enable rx_std_byteorder_ena port</td><td>On/Off</td><td>Enables the optional <code>rx_std_byte_order_ena</code> control input port. When this signal is asserted, the byte ordering block initiates a byte ordering operation if the Byte ordering control mode is set to manual. Once byte ordering has occurred, you must deassert and reassert this signal to perform another byte ordering operation. This signal is an synchronous input signal; however, it must be asserted for at least 1 cycle of <code>rx_std_clkout</code>.</td></tr>
</table>

Parameter	Range	Description
Enable rx_std_byteorder_flag port	**On/Off**	Enables the optional `rx_std_byteorder_flag` status output port. When asserted, indicates that the byte ordering block has performed a byte order operation. This signal is asserted on the clock cycle in which byte ordering occurred. This signal is synchronous to the `rx_std_clkout` clock.

Related Information

Transceiver Architecture in Cyclone V Devices

Byte Serializer and Deserializer

The byte serializer and deserializer allow the PCS to operate at twice the data width of the PMA serializer. This feature allows the PCS to run at a lower frequency and accommodate a wider range of FPGA interface widths.

Note: For more information refer to the Byte Serializer and Byte Deserializer sections in the *Transceiver Architecture in Cyclone V Devices*.

Table 16-10: Byte Serializer and Deserializer Parameters

Parameter	Range	Description
Enable TX byte serializer	**On/Off**	When you turn this option **On**, the PCS includes a TX byte serializer which allows the PCS to run at a lower clock frequency to accommodate a wider range of FPGA interface widths.
Enable RX byte deserializer	**On/Off**	When you turn this option **On**, the PCS includes an RX byte deserializer which allows the PCS to run at a lower clock frequency to accommodate a wider range of FPGA interface widths.

Related Information

Transceiver Architecture in Cyclone V Devices

8B/10B

The 8B/10B encoder generates 10-bit code groups from the 8-bit data and 1-bit control identifier. The 8B/10B decoder decodes the data into an 8-bit data and 1-bit control identifier.

In 8-bit 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.

Note: For more information refer to the *8B/10B Encoder* and *8B/10B Decoder* sections in the *Transceiver Architecture in Cyclone V Devices*.

Table 16-11: 8B/10B Encoder and Decoder Parameters

Parameter	Range	Description
Enable TX 8B/10B encoder	**On/Off**	When you turn this option **On**, the PCS includes the 8B/10B encoder.
Enable TX 8B/10B disparity control	**On/Off**	When you turn this option **On**, the PCS includes disparity control for the 8B/10B encoder. You force the disparity of the 8B/10B encoder using the `tx_forcedisp` and `tx_dispval` control signal.
Enable RX 8B/10B decoder	**On/Off**	When you turn this option **On**, the PCS includes the 8B/10B decoder.

Related Information

Transceiver Architecture in Cyclone V Devices

Rate Match FIFO

The rate match FIFO compensates for the very small frequency differences between the local system clock and the RX recovered clock.

For more information refer to the Rate Match FIFO sections in the *Transceiver Architecture in Cyclone V Devices*.

Table 16-12: Rate Match FIFO Parameters

Parameter	Range	Description
Enable RX rate match FIFO	**On/Off**	When you turn this option **On**, the PCS includes a FIFO to compensate for the very small frequency differences between the local system clock and the RX recovered clock.
RX rate match insert/delete +ve pattern (hex)	User-specified 20 bit pattern	Specifies the +ve (positive) disparity value for the RX rate match FIFO as a hexadecimal string.
RX rate match insert/delete -ve pattern (hex)	User-specified 20 bit pattern	Specifies the -ve (negative) disparity value for the RX rate match FIFO as a hexadecimal string.

When you enable the simplified data interface and enable the rate match FIFO status ports, the rate match FIFO bits map to the high-order bits of the data bus as listed in the following table. This table uses the following definitions:

- Basic double width: The **Standard PCS protocol mode** GUI option is set to **basic**. The FPGA data width is twice the PCS data width to allow the fabric to run at half the PCS frequency.
- Serial^{TM} RapidIO double width: You are implementing the Serial RapidIO protocol. The FPGA data width is twice the PCS data width to allow the fabric to run at half the PCS frequency.

Note: If you have the auto-negotiation state machine in your transceiver design, please note that the rate match FIFO is capable of inserting or deleting the first two bytes (K28.5//D2.2) of /C2/ ordered sets during auto-negotiation. However, the insertion or deletion of the first two bytes of /C2/ ordered sets can cause the auto-negotiation link to fail. For more information, visit **Altera Knowledge Base Support Solution**.

Table 16-13: Status Flag Mappings for Simplified Native PHY Interface

Status Condition	Protocol	Mapping of Status Flags to RX Data	Value
Full	PHY IP Core for PCI Express (PIPE) Basic double width	`RXD[62:62] = rx_rmfifostatus[1:0]`, or `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[30:29] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b11 = full`
	XAUI, GigE, Serial RapidIO double width	`rx_std_rm_fifo_full`	`1'b1 = full`
	All other protocols	Depending on the FPGA fabric to PCS interface width either: `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b11 = full`

Status Condition	Protocol	Mapping of Status Flags to RX Data	Value
Empty	PHY IP Core for PCI Express (PIPE) Basic double width	`RXD[62:62] = rx_rmfifostatus[1:0]`, or `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[30:29] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`(2'b10) AND (PAD or EDB)` PAD = K23.7 or 9'h1F7 EDB = K30.7 or 9'h1FE
	XAUI, GigE, Serial RapidIO double width	`rx_std_rm_fifo_empty`	`1'b1 = empty`
	All other protocols	Depending on the FPGA fabric to PCS interface width either: `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`(2'b10) AND (PAD or EDB)` (16) PAD = K23.7 or 9'h1F7 EDB = K30.7 or 9'h1FE
Insertion	Basic double width Serial RapidIO double width	`RXD[62:62] = rx_rmfifostatus[1:0]`, or `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[30:29] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b10`
	All other protocols	Depending on the FPGA fabric to PCS interface width either: `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b10`

(16) PAD and EBD are control characters. PAD character is typically used fo fill in the remaining lanes in a multi-lane link when one of the link goes to logical idle state. EDB indicates End Bad Packet.

Status Condition	Protocol	Mapping of Status Flags to RX Data	Value
Deletion	Basic double width Serial RapidIO double width	`RXD[62:62] = rx_rmfifostatus[1:0]`, or `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[30:29] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b01`
	All other protocols	Depending on the FPGA fabric to PCS interface width either: `RXD[46:45] = rx_rmfifostatus[1:0]`, or `RXD[14:13] = rx_rmfifostatus[1:0]`	`2'b01`

Related Information

Transceiver Architecture in Cyclone V Devices

Word Aligner and BitSlip Parameters

The word aligner aligns the data coming from RX PMA deserializer to a given word boundary. When the word aligner operates in bitslip mode, the word aligner slips a single bit for every rising edge of the bit slip control signal.

Note: For more information refer to the Word Aligner section in the *Transceiver Architecture inCycloneV Devices*.

Table 16-14: Word Aligner and BitSlip Parameters

Parameter	Range	Description
Enable TX bit slip	**On/Off**	When you turn this option **On**, the PCS includes the bitslip function. The outgoing TX data can be slipped by the number of bits specified by the `tx_bitslipboundarysel` control signal.
Enable tx_std_bitslipboundarysel control input port.	**On/Off**	When you turn this option **On**, the PCS includes the optional `tx_std_bitslipboundarysel` control input port.

Parameter	Range	Description
RX word aligner mode	**bit_slip** **sync_sm** **Manual**	Specifies one of the following 3 modes for the word aligner: • **bit_slip**: You can use bit slip mode to shift the word boundary. For every rising edge of the `rx_bitslip` signal, the word boundary is shifted by 1 bit. Each bitslip removes the earliest received bit from the received data. • **sync_sm**: In synchronous state machine mode, a programmable state machine controls word alignment. You can only use this mode with 8B/10B encoding. The data width at the word aligner can be 10 or 20 bits. When you select this word aligner mode, the synchronous state machine has hysteresis that is compatible with XAUI. However, when you select **cpri** for the **Standard PCS Protocol Mode**, this option selects the deterministic latency word aligner mode. • **Manual**: This mode enables word alignment by asserting the `rx_std_wa_pattern`. This is an edge sensitive signal.
RX word aligner pattern length	**7, 8, 10, 16, 20, 32, 40**	Specifies the length of the pattern the word aligner uses for alignment. The pattern is specified in LSBtoMSB order.
RX word aligner pattern (hex)	User-specified	Specifies the word aligner pattern in hex.
Number of word alignment patterns to achieve sync	`1–256`	Specifies the number of valid word alignment patterns that must be received before the word aligner achieves synchronization lock. The default is 3.
Number of invalid words to lose sync	`1–256`	Specifies the number of invalid data codes or disparity errors that must be received before the word aligner loses synchronization. The default is 3.
Number of valid data words to decrement error count	`1–256`	Specifies the number of valid data codes that must be received to decrement the error counter. If the word aligner receives enough valid data codes to decrement the error count to 0, the word aligner returns to synchronization lock.
Run length detector word count	`0–63`	Specifies the maximum number of contiguous 0s or 1s in the data stream before the word aligner reports a run length violation.

Parameter	Range	Description
Enable rx_std_wa_patternalign port	**On/Off**	Enables the optional `rx_std_wa_patternalign` control input port.
Enable rx_std_wa_a1a2size port	**On/Off**	Enables the optional `rx_std_wa_a1a2size` control input port.
Enable rx_std_bitslipboundarysel port	**On/Off**	Enables the optional `rx_std_wa_bitslipboundarysel` status output port.
Enable rx_std_bitslip port	**On/Off**	Enables the optional `rx_std_wa_bitslip` control input port.
Enable rx_std_runlength_err port	**On/Off**	Enables the optional `rx_std_wa_runlength_err` control input port.

Related Information

Transceiver Architecture inCycloneV Devices

Bit Reversal and Polarity Inversion

The bit reversal and polarity inversion functions allow you to reverse bit order, byte order, and polarity to correct errors and to accommodate different layouts of data.

Table 16-15: Bit Reversal and Polarity Inversion Parameters

Parameter	Range	Description
Enable TX bit reversal	**On/Off**	When you turn this option **On**, the word aligner reverses TX parallel data before transmitting it to the PMA for serialization. You can only change this static setting using the Transceiver Reconfiguration Controller.
Enable RX bit reversal	**On/Off**	When you turn this option **On**, the `rx_std_bitrev_ena` port controls bit reversal of the RX parallel data after it passes from the PMA to the PCS.
Enable RX byte reversal	**On/Off**	When you turn this option **On**, the word aligner reverses the byte order before transmitting data. This function allows you to reverse the order of bytes that were erroneously swapped. The PCS can swap the ordering of both 8 and10 bit words.

Parameter	Range	Description
Enable TX polarity inversion	**On/Off**	When you turn this option **On**, the `tx_std_polinv` port controls polarity inversion of TX parallel data before transmitting the parallel data to the PMA.
Enable RX polarity inversion	**On/Off**	When you turn this option **On**, asserting `rx_std_polinv` controls polarity inversion of RX parallel data after PMA transmission.
Enable rx_std_bitrev_ena port	**On/Off**	When you turn this option **On**, asserting `rx_std_bitrev_ena` control port causes the RX data order to be reversed from the normal order, LSB to MSB, to the opposite, MSB to LSB. This signal is an asynchronous input.
Enable rx_std_byterev_ena port	**On/Off**	When you turn this option **On**, asserting `rx_std_byterev_ena` input control port swaps the order of the individual 8 or 10bit words received from the PMA.
Enable tx_std_polinv port	**On/Off**	When you turn this option **On**, the `tx_std_polinv` input is enabled. You can use this control port to swap the positive and negative signals of a serial differential link if they were erroneously swapped during board layout.
Enable rx_std_polinv port	**On/Off**	When you turn this option **On**, the `rx_std_polinv` input is enabled. You can use this control port to swap the positive and negative signals of a serial differential link if they were erroneously swapped during board layout.
Enable tx_std_elecidle port	**On/Off**	When you turn this option **On**, the `tx_std_elecidle` input port is enabled. When this signal is asserted, it forces the transmitter to electrical idle.

Parameter	Range	Description
Enable rx_std_signaldetect port	**On/Off**	When you turn this option **On**, the optional `rx_std_signaldetect` output port is enabled. This signal is required for the PCI Express protocol. If enabled, the signal threshold detection circuitry senses whether the signal level present at the RX input buffer is above the signal detect threshold voltage that you specified. For SATA / SAS applications, enable this port and set the following QSF assignments to the transceiver receiver pin: • `set_instance_assignment -name XCVR_RX_SD_ENABLE ON` • `set_instance_assignment -name XCVR_RX_SD_THRESHOLD 7` • `set_instance_assignment -name XCVR_RX_COMMON_MODE_VOLTAGE VTT_OP55V` • `set_instance_assignment -name XCVR_RX_SD_OFF 1` • `set_instance_assignment -name XCVR_RX_SD_ON 2` **Note:** The above QSF assignments provide you an example, but the actual setting may vary as per the recommendations from the characterization report. You are also required to set up the platform device level pairing.

Interfaces

The Native PHY includes several interfaces that are common to all parameterizations.

The Native PHY allows you to enable ports, even for disabled blocks to facilitate dynamic reconfiguration.

The Native PHY uses the following prefixes for port names:

- Standard PCS ports—`tx_std`, `rx_std`

The port descriptions use the following variables to represent parameters:

- <*n*>—The number of lanes
- <*p*>—The number of PLLs
- <*r*>—The number of CDR references clocks selected

Common Interface Ports

Common interface consists of reset, clock signals, serial interface ports, control and status ports, parallel data ports, and reconfig interface ports.

Figure 16-3: Common Interface Ports

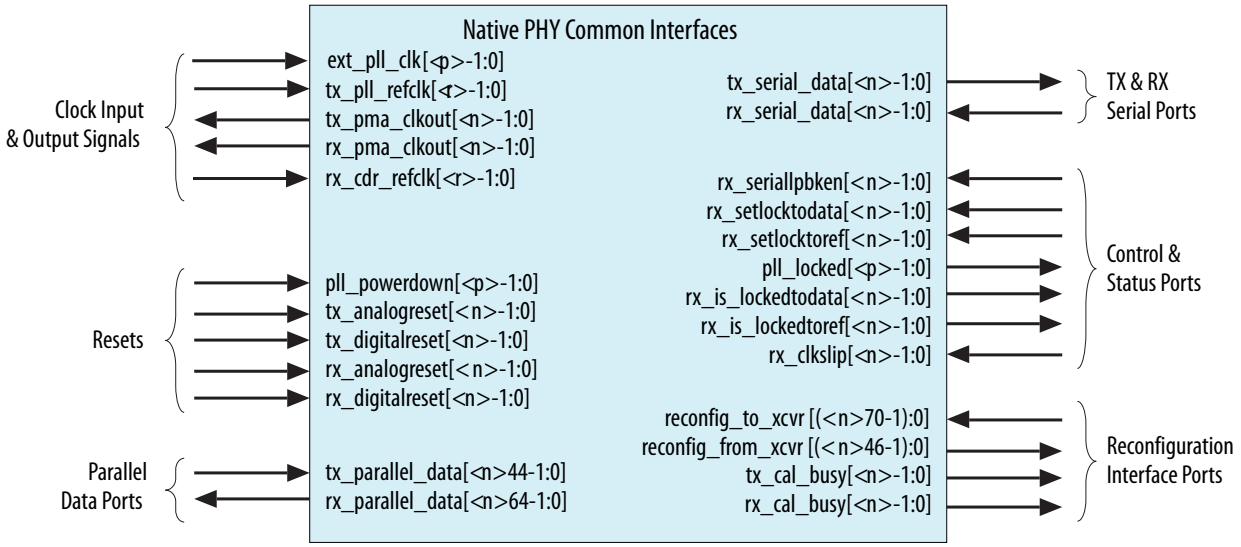


Table 16-16: Native PHY Common Interfaces

Name	Direction	Description
Clock Inputs and Output Signals		
`tx_pll_refclk[<r>-1:0]`	Input	The reference clock input to the TX PLL.
`rx_pma_clkout[<n>-1:0]`	Output	RX parallel clock (recovered clock) output from PMA
`rx_cdr_refclk[<n>-1:0]`	Input	Input reference clock for the RX PFD circuit.
`ext_pll_clk[ <p> -1:0]`	Input	This optional signal is created when you select the **Use external TX PLL** option. If you instantiate a fractional PLL which is external to the Native PHY IP, then connect the output clock of this PLL to `ext_pll_clk`.
Resets		

Name	Direction	Description
pll_powerdown[<p>-1:0]	Input	When asserted, resets the TX PLL. Active high, edge sensitive reset signal. By default, the Cyclone Native Transceiver PHY IP Core create a separate pll_powerdown signal for each logical PLL. However, the Fitter may merge the PLLs if they are in the same transceiver bank. PLLs can only be merged if their pll_powerdown signals are driven from the same source. If the PLLs are in separate transceiver banks, you can choose to drive the pll_powerdown signals separately.
tx_analogreset[<n>-1:0]	Input	When asserted, resets for TX PMA, TX clock generation block, and serializer. Active high, edge sensitive reset signal.
tx_digitalreset[<n>-1:0]	Input	When asserted, resets the digital components of the TX datapath. Active high, edge sensitive, asynchronous reset signal. If your design includes bonded TX PCS channels, refer to *Timing Constraints for Reset Signals* when Using Bonded PCS Channels for a SDC constraint you must include in your design.
rx_analogreset[<n>-1:0]	Input	When asserted, resets the RX CDR, deserializer. Active high, edge sensitive, asynchronous reset signal.
rx_digitalreset[<n>-1:0]	Input	When asserted, resets the digital components of the RX datapath. Active high, edge sensitive, asynchronous reset signal.
Parallel data ports		
tx_parallel_data[43:0]	Input	PCS TX parallel data, consisting of 4, 11-bit words. Refer to *Table 15-16* for bit definitions. Refer to *Table 15-17* for the locations of valid words in each parameter.
rx_parallel_data[63:0]	Output	PCS RX parallel data, consisting of 4, 16-bit words. Refer to *Table 15-18* for bit definitions. Refer to *Table 15-19* for the locations of valid words in each parameter .

Name	Direction	Description
TX and RX serial ports		
tx_serial_data[<n>-1:0]	Output	TX differential serial output data.
rx_serial_data[<n>-1:0]	Input	RX differential serial output data.
Control and Status ports		
rx_seriallpbken[<n>-1:0]	Input	When asserted, the transceiver enters serial loopback mode. Loopback drives serial TX data to the RX interface.
rx_set_locktodata[<n>-1:0]	Input	When asserted, programs the RX CDR to manual lock to data mode in which you control the reset sequence using the rx_set_locktoref and rx_set_locktodata. Refer to "Transceiver Reset Sequence" in *Transceiver Reset Control in Cyclone V Devices* for more information about manual control of the reset sequence.
rx_set_locktoref[<n>-1:0]	Input	When asserted, programs the RX CDR to manual lock to reference mode in which you control the reset sequence using the rx_set_locktoref and rx_set_locktodata. Refer to Refer to "Transceiver Reset Sequence" in *Transceiver Reset Control in CycloneV Devices* for more information about manual control of the reset sequence.
pll_locked[<p>-1:0]	Output	When asserted, indicates that the PLL is locked to the input reference clock.
rx_is_lockedtodata[<n>-1:0]	Output	When asserted, the CDR is locked to the incoming data.
rx_is_lockedtoref[<n>-1:0]	Output	When asserted, the CDR is locked to the incoming reference clock.

Name	Direction	Description
rx_clkslip[<n>-1:0]	Input	When you turn this signal on, the deserializer performs a clock slip operation to achieve word alignment. The clock slip operation alternates between skipping 1 serial bit and pausing the serial clock for 2 cycles to achieve word alignment. As a result, the period of the parallel clock can be extended by 2 unit intervals (UI) during the clock slip operation. This is an optional control input signal.
Reconfig Interface Ports		
reconfig_to_xcvr [(<n>70-1):0]	Input	Reconfiguration signals from the Transceiver Reconfiguration Controller. <n> grows linearly with the number of reconfiguration interfaces.
reconfig_from_xcvr [(<n>46-1):0]	Output	Reconfiguration signals to the Transceiver Reconfiguration Controller. <n> grows linearly with the number of reconfiguration interfaces.
tx_cal_busy[<n>-1:0]	Output	When asserted, indicates that the initial TX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP. You must hold the channel in reset until calibration completes.
rx_cal_busy[<n>-1:0]	Output	When asserted, indicates that the initial RX calibration is in progress. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP.

Table 16-17: Signal Definitions for `tx_parallel_data` with and without 8B/10B Encoding

The following table shows the signals within `tx_parallel_data` that correspond to data, control, and status signals.

TX Data Word	Description
Signal Definitions with 8B/10B Enabled	
tx_parallel_data[7:0]	TX data bus
tx_parallel_data[8]	TX data control character
tx_parallel_data[9]	Force disparity, validates disparity field.

TX Data Word	Description
tx_parallel_data[10]	Specifies the current disparity as follows: • 1'b0 = positive • 1'b1 = negative
Signal Definitions with 8B/10B Disabled	
tx_parallel_data[9:0]	TX data bus
tx_parallel_data[10]	Unused

Table 16-18: Location of Valid Data Words for tx_parallel_data for Various FPGA Fabric to PCS Parameterizations

The following table shows the valid 11-bit data words with and without the byte deserializer for single- and double-word FPGA fabric to PCS interface widths.

Configuration	Bus Used Bits
Single word data bus, byte deserializer disabled	[10:0] (word 0)
Single word data bus, byte serializer enabled	[32:22], [10:0] (words 0 and 2)
Double word data bus, byte serializer disabled	[21:0] (words 0 and 1)
Double word data bus, byte serializer enabled	[43:0] (words 0-3)

Table 16-19: Signal Definitions for rx_parallel_data with and without 8B/10B Encoding

This table shows the signals within rx_parallel_data that correspond to data, control, and status signals.

RX Data Word	Description
Signal Definitions with 8B/10B Enabled	
rx_parallel_data[7:0]	RX data bus
rx_parallel_data[8]	RX data control character
rx_parallel_data[9]	Error Detect
rx_parallel_data[10]	Word Aligner / synchronization status
rx_parallel_data[11]	Disparity error
rx_parallel_data[12]	Pattern detect
rx_parallel_data[14:13]	The following encodings are defined: • 2'b00: Normal data • 2'b01: Deletion • 2'b10: Insertion • 2'b11: Underflow
rx_parallel_data[15]	Running disparity value
Signal Definitions with 8B/10B Disabled	
rx_parallel_data[9:0]	RX data bus

RX Data Word	Description
rx_parallel_data[10]	Word Aligner / synchronization status
rx_parallel_data[11]	Disparity error
rx_parallel_data[12]	Pattern detect
rx_parallel_data[14:13]	The following encodings are defined: • 2'b00: Normal data • 2'b01: Deletion • 2'b10: Insertion (or Underflow with 9'h1FE or 9'h1F7) • 2'b11: Overflow
rx_parallel_data[15]	Running disparity value

Table 16-20: Location of Valid Data Words for rx_parallel_data for Various FPGA Fabric to PCS Parameterizations

The following table shows the valid 16-bit data words with and without the byte deserializer for single- and double-word FPGA fabric to PCS interface widths.

Configuration	Bus Used Bits
Single word data bus, byte deserializer disabled	[15:0] (word 0)
Single word data bus, byte serializer enabled	[47:32], [15:0] (words 0 and 2)
Double word data bus, byte serializer disabled	[31:0] (words 0 and 1)
Double word data bus, byte serializer enabled	[63:0] (words 0-3)

Related Information

- **Timing Constraints for Bonded PCS and PMA Channels** on page 18-11
- **Transceiver Architecture in Cyclone V Devices**
- **Transceiver Architecture in Cyclone V Devices**

Cyclone V Standard PCS Interface Ports

This section describes the signals that comprise the Standard PCS interface.

Figure 16-4: Standard PCS Interfaces

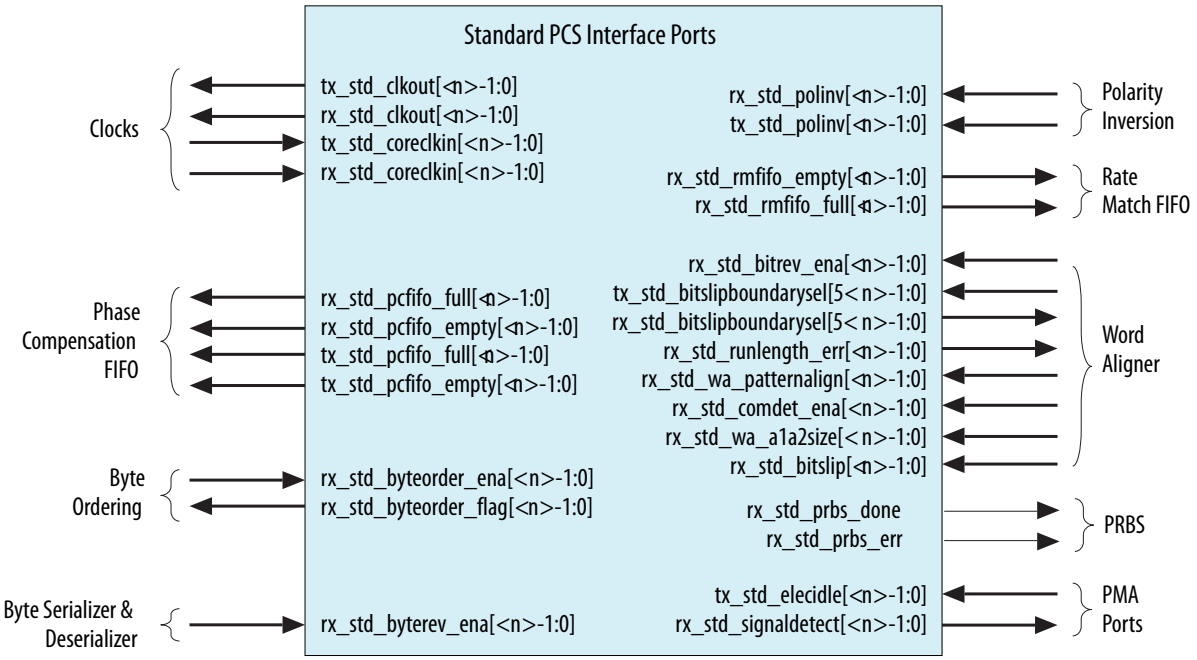


Table 16-21: Standard PCS Interface Ports

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
Clocks			
tx_std_clkout[<n>-1:0]	Output	—	TX Parallel clock output.
rx_std_clkout[<n>-1:0]	Output	—	RX parallel clock output. The CDR circuitry recovers RX parallel clock from the RX data stream.
tx_std_coreclkin[<n>-1:0]	Input	—	TX parallel clock input from the FPGA fabric that drives the write side of the TX phase compensation FIFO.
rx_std_coreclkin[<n>-1:0]	Input	—	RX parallel clock that drives the read side of the RX phase compensation FIFO.
Phase Compensation FIFO			
rx_std_pcfifo_full[<n>-1:0]	Output	Yes	RX phase compensation FIFO full status flag.

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
rx_std_pcfifo_empty[<n>-1:0]	Output	Yes	RX phase compensation FIFO status empty flag.
tx_std_pcfifo_full[<n>-1:0]	Output	Yes	TX phase compensation FIFO status full flag.
tx_std_pcfifo_empty[<n>-1:0]	Output	Yes	TX phase compensation FIFO status empty flag.
Byte Ordering			
rx_std_byteorder_ena[<n>-1:0]	Input	No	Byte ordering enable. When this signal is asserted, the byte ordering block initiates a byte ordering operation if the **Byte ordering control mode** is set to **manual**. Once byte ordering has occurred, you must deassert and reassert this signal to perform another byte ordering operation. This signal is an synchronous input signal; however, it must be asserted for at least 1 cycle of rx_std_clkout.
rx_std_byteorder_flag[<n>-1:0]	Output	Yes	Byte ordering status flag. When asserted, indicates that the byte ordering block has performed a byte order operation. This signal is asserted on the clock cycle in which byte ordering occurred. This signal is synchronous to the rx_std_clkout clock. You must a synchronizer this signal.
Byte Serializer and Deserializer			
rx_std_byterev_ena[<n>-1:0]	Input	No	This control signal is available in when the PMA width is 16 or 20 bits. When asserted, enables byte reversal on the RX interface.
8B/10B			

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
rx_std_polinv[<n>-1:0]	Input	No	Polarity inversion for the 8B/10B decoder, When set, the RX channels invert the polarity of the received data. You can use this signal to correct the polarity of differential pairs if the transmission circuitry or board layout mistakenly swapped the positive and negative signals. The polarity inversion function operates on the word aligner input.
tx_std_polinv[<n>-1:0]	Input	No	Polarity inversion, part of 8B10B encoder, When set, the TX interface inverts the polarity of the TX data.
Rate Match FIFO			
rx_std_rmfifo_empty[<n>-1:0]	Output	No	Rate match FIFO empty flag. When asserted, the rate match FIFO is empty.
rx_std_rmfifo_full[<n>-1:0]	Output	No	Rate match FIFO full flag. When asserted the rate match FIFO is full. You must synchronize this signal.
Word Aligner			
rx_std_bitrev_ena[<n>-1:0]	Input	No	When asserted, enables bit reversal on the RX interface. Bit order may be reversed if external transmission circuitry transmits the most significant bit first. When enabled, the receive circuitry receives all words in the reverse order. The bit reversal circuitry operates on the output of the word aligner.
tx_std_bitslipboundarysel[5<n>-1:0]	Input	No	BitSlip boundary selection signal. Specifies the number of bits that the TX bit slipper must slip.
rx_std_bitslipboundarysel[5<n>-1:0]	Output	No	This signal operates when the word aligner is in bitslip word alignment mode. It reports the number of bits that the RX block slipped to achieve deterministic latency.

Name	Dir	Synchronous to tx_std_coreclkin/ rx_std_coreclkin	Description
rx_std_runlength_err[<n>-1:0]	Output	No	When asserted, indicates a run length violation. Asserted if the number of consecutive 1s or 0s exceeds the number specified in the parameter editor GUI.
rx_st_wa_patternalign	Input	No	Active when you place the word aligner in manual mode. In manual mode, you align words by asserting rx_st_wa_patternalign. rx_st_wa_patternalign is edge sensitive. For more information refer to the *Word Aligner* section in the *Transceiver Architecture in Cyclone V Devices*.
rx_std_wa_a1a2size[<n>-1:0]	Input	No	Used for the SONET protocol. Assert when the A1 and A2 framing bytes must be detected. A1 and A2 are SONET backplane bytes and are only used when the PMA data width is 8 bits.
rx_std_bitslip[<n>-1:0]	Input	No	Used when word aligner mode is bitslip mode. For every rising edge of the rx_std_bitslip signal, the word boundary is shifted by 1 bit. Each bitslip removes the earliest received bit from the received data. You must synchronize this signal.
Miscellaneous			
tx_std_elecidle[<n>-1:0]	Input		When asserted, enables a circuit to detect a downstream receiver. This signal must be driven low when not in use because it causes the TX PMA to enter electrical idle mode with the TX serial data signals in tristate mode.
rx_std_signaldetect[<n>-1:0]	Output	No	Signal threshold detect indicator. When asserted, it indicates that the signal present at the receiver input buffer is above the programmed signal detection threshold value. You must synchronize this signal.

Related Information

Transceiver Architecture in Cyclone V Devices

SDC Timing Constraints

This section describes SDC timing constraints for the Cyclone V Native PHY.

The Intel Quartus Prime software reports timing violations for asynchronous inputs to the Standard PCS. Because many violations are for asynchronous paths, they do not represent actual timing failures. You may choose one of the following three approaches to identify these false timing paths to the Intel Quartus Prime or TimeQuest software.

- You can cut these paths in your Synopsys Design Constraints (**.sdc**) file by using the `set_false_path` command as shown in the following example.

Example 16-1: Using the set_false_path Constraint to Identify Asynchronous Inputs

```
set_false_path -through {*8gbitslip*} -to  [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*8gbytordpld*} -to [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*8gcmpfifoburst*} -to [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*8gphfifoburstrx*} -to [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]

set_false_path -through {*8gsyncsmen*} -to [get_registers
*8g*pcs*SYNC_DATA_REG*]
set_false_path -through {*8gwrdisablerx*} -to [get_registers
*8g_rx_pcs*SYNC_DATA_REG*]
set_false_path -through {*rxpolarity*} -to [get_registers *SYNC_DATA_REG*]
set_false_path -through {*pldeidleinfersel*} -to [get_registers
*SYNC_DATA_REG*]
```

- You can use the `set_max_delay` constraint on a given path to create a constraint for asynchronous signals that do not have a specific clock relationship but require a maximum path delay.

Example 16-2: Using the max_delay Constraint to Identify Asynchronous Inputs

```
# Example: Apply 10ns max delay
set_max_delay -from *tx_from_fifo* -to *8g*pcs*SYNC_DATA_REG1 10
```

- You can use the `set_false` path command only during Timequest timing analysis.

Example 16-3: Using the set_false TimeQuest Constraint to Identify Asynchronous Inputs

```
#if {$::TimeQuestInfo(nameofexecutable) eq "quartus_fit"} {
#} else {
#set_false_path -from [get_registers {*tx_from_fifo*}] -through
{*txbursten*} -to [get_registers *8g_*_pcs*SYNC_DATA_REG
```

Note: In all of these examples, you must substitute your actual signal names for the signal names shown.

Dynamic Reconfiguration

Dynamic reconfiguration calibrates each channel to compensate for variations due to process, voltage, and temperature (PVT).

These process variations result in analog voltages that can be offset from required ranges. The calibration performed by the dynamic reconfiguration interface compensates for variations due to PVT.

For non-bonded clocks, each channel and each TX PLL has a separate dynamic reconfiguration interfaces. The MegaWizard Plug-In Manager provides informational messages on the connectivity of these interfaces. The following example shows the messages for the Cyclone V Native PHY with four duplex channels, four TX PLLs, in a nonbonded configuration.

For more information about transceiver reconfiguration refer to Transceiver Reconfiguration Controller IP Core.

Example 16-4: Informational Messages for the Transceiver Reconfiguration Interface

```
PHY IP will require 8 reconfiguration interfaces for connection to the
external reconfiguration controller.
Reconfiguration interface offsets 0-3 are connected to the transceiver
channels.
Reconfiguration interface offsets 4–7 are connected to the transmit PLLs.
```

Related Information

Transceiver Architecture in Cyclone V Devices

Simulation Support

The Intel Quartus Prime release provides simulation and compilation support for the Native PHY IP Core. Refer to Running a Simulation Testbench for a description of the directories and files that the Intel Quartus Prime software creates automatically when you generate your Native PHY IP Core.

Related Information

Running a Simulation Testbench on page 1-6

Slew Rate Settings

The following transceiver slew rate settings are allowed in Intel Quartus Prime software.

Table 16-22: Slew Rate Settings for Cyclone V devices

Protocol / Datarate	Allowed Intel Quartus Prime Settings	IBIS-AMI Settings
PCI Express Gen 2	4	Protocol - (1) PCIe
PCI Express Gen1, XAUI	3	Protocol - (4) XAUI
Gigabit Ethernet	2	Protocol - (3) Gigabit Ethernet

Protocol / Datarate	Allowed Intel Quartus Prime Settings	IBIS-AMI Settings
<1 Gbps	2	Protocol - (0) Basic, Slew - (0)
1 Gbps - 3 Gbps	2, 3	Protocol - (0) Basic, Slew - (0 or 1)
3 Gbps - 5 Gbps	3, 4	Protocol - (0) Basic, Slew - (1 or 2)
>5 Gbps	5	Protocol - (0) Basic, Slew - (2)

Note: For protocols not mentioned in the above table, the Intel Quartus Prime and IBIS-AMI settings will be in accordance to the specified data rates. For example, for SATA 3Gbps; allowed Intel Quartus Prime and IBIS-AMI settings are 3 and Protocol - (4) XAUI respectively.

Transceiver Reconfiguration Controller IP Core Overview 17

2020.06.02

UG-01080

Subscribe Send Feedback

The Altera Transceiver Reconfiguration Controller dynamically reconfigures analog settings in Arria V, Arria V GZ, Cyclone V, and Stratix V devices. Dynamic reconfiguration allows you to compensate for variations due to process, voltage, and temperature (PVT) in 28-nm devices.

Dynamic reconfiguration is required for Arria V, Arria V GZ, Cyclone V, and Stratix V devices that include transceivers. The reconfiguration functionality available in Arria V and Cyclone V devices is a subset of the functionality available for Stratix V devices.

Note: Some of the reconfiguration features not available for Arria V and Cyclone V devices in the current release, may be available in subsequent releases. Arria V and Cyclone V devices do not include ATX PLLs. Stratix V and Arria V GZ devices include ATX PLLs.

Table 17-1: Device Support for Dynamic Reconfiguration

Area	Feature	Stratix V	Arria V	Arria V GZ	Cyclone V
Calibration Functions	Offset cancellation	Yes	Yes	Yes	Yes
	Duty cycle distortion calibration	—	Yes	—	Yes
	ATX PLL calibration	Yes	—	Yes	—
Analog Features	On-chip signal quality monitoring	Yes	—	Yes	—
	Decision feedback equalization (DFE)	Yes	—	Yes	—
	Adaptive equalization	Yes	—	Yes	—
Loopback modes	Pre-CDR reverse serial loopback	Yes	Yes	Yes	Yes
	Post-CDR reverse serial loopback	Yes	Yes	Yes	Yes
PLL reconfiguration	Reference clock switching (CDR, ATX PLLs, and TX PLLs)	Yes	Yes	Yes	Yes
	TX PLL connected to a transceiver channel reconfiguration	Yes	Yes	Yes	Yes

Area	Feature	Stratix V	Arria V	Arria V GZ	Cyclone V
Transceiver Channel/PLL Reconfiguration	RX CDR reconfiguration	Yes	Yes	Yes	Yes
	Reconfiguration of PCS blocks	Yes	Yes	Yes	Yes
	TX PLL switching	Yes	Yes	Yes	Yes
	ATX PLL switching	Yes	—	Yes	—
	TX local clock divider reconfiguration (1,2,4,8)	Yes	Yes	Yes	Yes
	FPGA fabric-transceiver channel data width reconfiguration	Yes	Yes	Yes	Yes

For more information about the features that are available for each device refer to the following device documentation: *Dynamic Reconfiguration in Stratix V Devices*, *Dynamic Reconfiguration in Arria V Devices*, and *Dynamic Reconfiguration in Cyclone V Devices*. These chapters are included in the Stratix V, Arria V, and Cyclone V device handbooks, respectively.

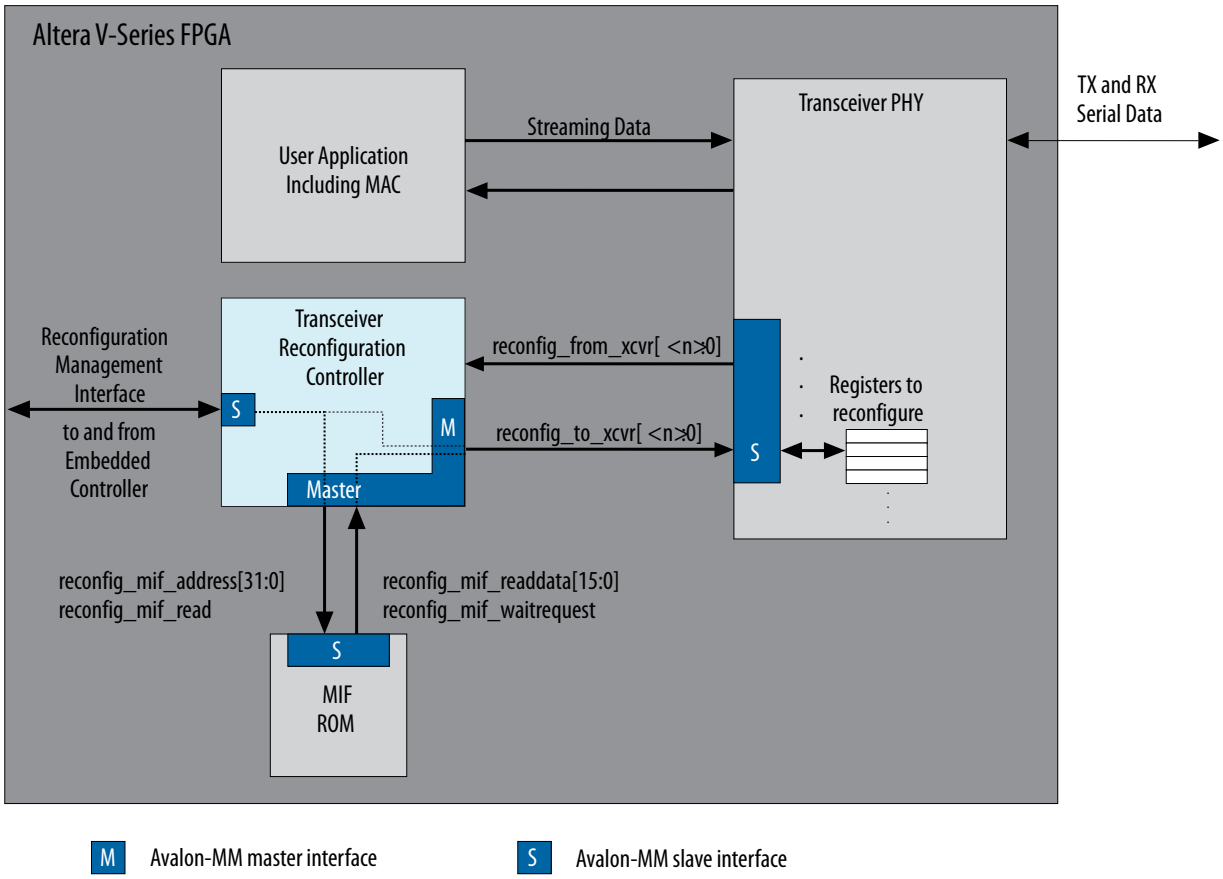
Related Information

- **Dynamic Reconfiguration in Stratix V Devices**
- **Dynamic Reconfiguration in Arria V Devices**
- **Dynamic Reconfiguration in Cyclone V Devices**

Transceiver Reconfiguration Controller System Overview

This section describes the Transceiver Reconfiguration Controller's role. You can include the embedded controller that initiates reconfiguration in your FPGA or use an embedded processor on the PCB.

Figure 17-1: Transceiver Reconfiguration Controller



An embedded controller programs the Transceiver Reconfiguration Controller using its Avalon-MM slave interface. The `reconfig_to_xcvr` and `reconfig_from_xcvr` buses include the Avalon-MM address, read, write, readdata, writedata, and signals that connect to features related to calibration and signal integrity.

The Transceiver Reconfiguration Controller provides two modes to dynamically reconfigure transceiver settings:

- Register Based—In this access mode you can directly reconfigure a transceiver PHY IP core using the Transceiver Reconfiguration Controller's reconfiguration management interface. You initiate reconfiguration using a series of Avalon-MM reads and writes to the appropriate registers of the Transceiver Reconfiguration Controller. The Transceiver Reconfiguration Controller translates the device independent commands received on the reconfiguration management interface to device dependent commands on the transceiver reconfiguration interface. For more information, refer to *Changing Transceiver Settings Using Register-Based Reconfiguration*.

  For more information about Avalon-MM interfaces including timing diagrams, refer to the Avalon Interface Specifications.
- Streamer Based —This access mode allows you to either stream a MIF that contains the reconfiguration data or perform direct writes to perform reconfiguration. The streaming mode uses a memory initialization file (**.mif**) to stream an update to the transceiver PHY IP core. The **.mif** file can contain changes for many settings. For example, a single **.mif** file might contain changes to the PCS datapath settings, clock settings, and PLL parameters. You specify the **.mif** using write commands on the Avalon-MM PHY management interface. After the streaming operation is specified, the update proceeds in a single step. For more information, refer to *Changing Transceiver Settings Using Streamer-Based Reconfiguration*. In the direct write mode, you perform Avalon-MM reads and writes to initiate a reconfiguration of the PHY IP. For more information, refer to *Direct Write Reconfiguration*.

The following table shows the features that you can reconfigure or control using register-based and MIF-based access modes for Stratix V devices.

Table 17-2: Reconfiguration Feature Access Modes

Feature	Register-Based	Streamer-Based
PMA settings, including V_{OD}, pre-emphasis, RX equalization DC gain, RX equalization control	Yes	Yes
Pre-CDR and post-CDR loopback modes	Yes	—
DFE post taps and polarity	Yes	—
AEQ mode	Yes	—
Eye Monitor	Yes	—
ATX Tuning	Yes	Yes
Reference clock	Yes	Yes
TX PLL clock switching	—	Yes
Channel interface	—	Yes

Related Information

- **Changing Transceiver Settings Using Register-Based Reconfiguration** on page 17-43
- **Changing Transceiver Settings Using Streamer-Based Reconfiguration** on page 17-44
- **Direct Write Reconfiguration** on page 17-44

- Avalon Interface Specifications

Transceiver Reconfiguration Controller Performance and Resource Utilization

This section describes the approximate device resource utilization for a the Transceiver Reconfiguration Controller for Stratix V devices. The numbers of combinational ALUTs and logic registers are rounded to the nearest 50.

Note: To close timing, you may need to instantiate multiple instances of the Transceiver Reconfiguration Controller IP Core to the multiple transceiver PHYs in your design to reduce routing delays. However, you cannot connect multiple Transceiver Reconfiguration Controllers to a single transceiver PHY.

Table 17-3: Resource Utilization for Stratix V Devices

Component	ALUTs	Registers	Memory Blocks	M20Ks	Run Time
Transceiver Calibration Functions					
Offset Cancellation	500	400	0	0	100 us/channel
Duty cycle calibration	350	400	0	0	70 us/channel
ATX PLL calibration	650	450	0	4	60 us/channel
Analog Features					
EyeQ	300	200	0	0	-
AEQ	700	500	0	0	40 us/channel
Reconfiguration Features					
Channel and PLL reconfiguration	400	500	0	0	- (17)
PLL reconfiguration (only)	250	350	0	0	

Parameterizing the Transceiver Reconfiguration Controller IP Core

Complete the following steps to configure the Transceiver Reconfiguration Controller IP Core in the MegaWizard Plug-In Manager:

1. Under **Tools** > **IP Catalog**, select the device family of your choice.
2. Under **Tools** > **Interfaces** > **Transceiver PHY** > **Transceiver Reconfiguration Controller**
3. Select the options required for your design.
4. Click **Finish** to generate your parameterized Reconfiguration Controller IP Core.

(17) The time to complete these functions depends upon the complexity of the reconfiguration operation.

Parameterizing the Transceiver Reconfiguration Controller IP Core in Qsys

Complete the following steps to configure the Transceiver Reconfiguration Controller IP Core in Qsys:

1. On the **Project Settings** tab, select **Arria V**, **Arria V GZ**, **Cyclone V**, or **Stratix V** from the list.
2. On the Component Library tab, type the following text string in the search box: `reconfig`. Qsys filters the component library and shows all components matching the text string you entered.
3. Click **Transceiver Reconfiguration Controller** and then click +**Add**.
4. Select the options required for your design. For a description of these options, refer to the **General Options Parameters**.
5. Click **Finish** to generate your customized Transceiver Reconfiguration Controller PHY IP Core.

General Options Parameters

This section lists the available options.

Table 17-4: General Options

Name	Value	Description
Device family	Arria V Arria V GZ Cyclone V Stratix V	Specifies the device family. The reconfiguration functions available for Arria V and Cyclone V devices are a subset of those available for Stratix V devices. Refer to Device Support for Dynamic Reconfiguration for more information about available functions.
Interface Bundles		
Number of reconfiguration interfaces	`<IF>`	Specifies the total number of reconfiguration interfaces that connect to the Transceiver Reconfiguration Controller. There is one interface for each channel and TX PLL. When you specify the parameters for a transceiver PHY, the message window displays the number of interfaces required.
Optional interface grouping	$\texttt{<Grp}_1\texttt{>,<Grp}_2\texttt{>,}$ $\texttt{<Grp}_3\texttt{>}$	Specifies the grouping of reconfiguration interfaces as a comma-separated list with each integer indicating the total number of reconfiguration interfaces that are connected to a transceiver PHY instance. Leave this entry blank if all reconfiguration interfaces connect to the same transceiver PHY instance. Refer to Understanding Logical Channel Numbering for more information about grouping interfaces.

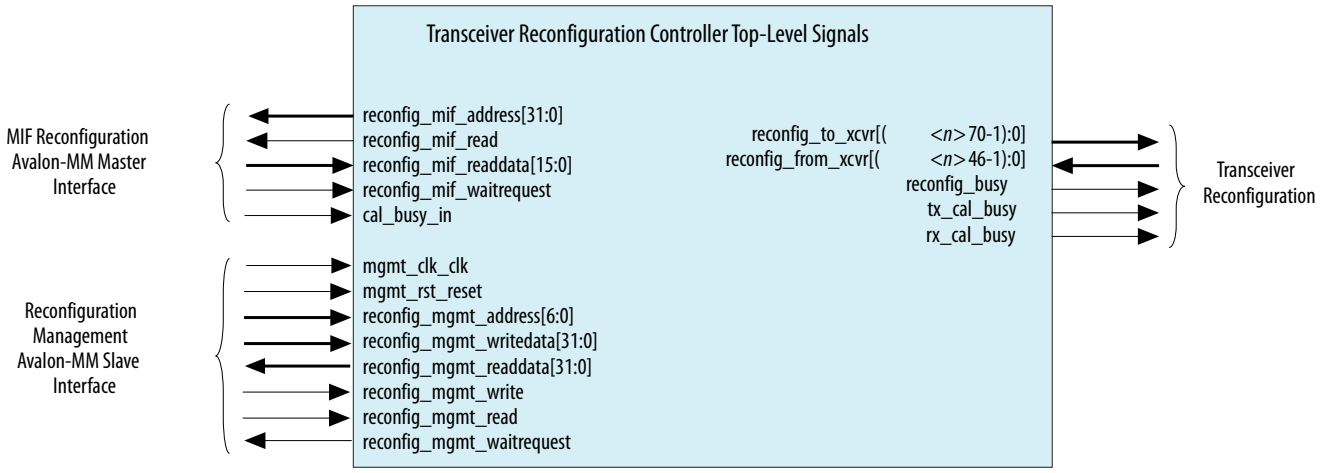
Name	Value	Description
Transceiver Calibration Functions		
Enable offset cancellation	**On**	When enabled, the Transceiver Reconfiguration Controller includes the offset cancellation functionality. This option is always on. Offset cancellation occurs automatically at power-up and runs only once.
Enable duty cycle calibration	**On/Off**	For Arria V devices, when enable, DCD calibrates for duty cycle distortion caused by clock network skew. DCD calibration runs once during power up. You should enable this option for protocols running at greater than 4.9152 Gbps.
Enable PLL calibration	**On/Off**	When enabled, an algorithm that improves the signal integrity of the PLLs is included in the Transceiver Reconfiguration Controller IP Core. This feature is only available for Stratix V devices.
Create optional calibration status ports	**On/Off**	When you turn this option **On**, the core includes `tx_cal_busy` and `rx_cal_busy` ports. These signals are asserted when calibration is active.
Analog Features		
Enable Analog controls	**On/Off**	When enabled, TX and RX signal conditioning features are enabled.
Enable EyeQ block	**On/Off**	When enabled, you can use the EyeQ, the on-chip signal quality monitoring circuitry, to estimate the actual eye opening at the receiver. This feature is only available for Stratix V devices.
Enable decision feedback equalizer (DFE) block	**On/Off**	When you turn this option On, the Transceiver Reconfiguration Controller includes logic to perform DFE
Enable adaptive equalization (AEQ) block	**On/Off**	When enabled, the Transceiver Reconfiguration Controller includes logic to perform AEQ. This feature is only available for Stratix V devices.
Reconfiguration Features		
Enable channel/PLL reconfiguration	**On/Off**	When enabled, the Transceiver Reconfiguration Controller includes logic to include both channel and PLL reconfiguration.

Name	Value	Description
Enable PLL reconfiguration support block	**On/Off**	When enabled, the Transceiver Reconfiguration Controller includes logic to perform PLL reconfiguration.

Transceiver Reconfiguration Controller Interfaces

This section describes the top-level signals of the Transceiver Reconfiguration Controller.

Figure 17-2: Top-Level Signals of the Transceiver Reconfiguration Controller



Note: By default, the **Block Diagram** shown in the MegaWizard Plug-In Manager labels the external pins with the *interface type* and places the *interface name* inside the box. The interface type and name are used in the Hardware Component Description File (**_hw.tcl**). If you click **Show signals**, the block diagram expands to show all of the signals of the component given the options currently selected in the MegaWizard Plug-In Manager.

For more information about **_hw.tcl** files refer to the *Component Interface Tcl Reference* in volume 1 of the *Intel Quartus Prime Handbook*.

Related Information

Component Interface Tcl Reference

MIF Reconfiguration Management Avalon-MM Master Interface

This section describes the signals that comprise of the MIF Reconfiguration Management Interface. The Transceiver Reconfiguration Controller communicates to an on-chip ROM or any other memory used to store the MIF using this interface.

Table 17-5: MIF Reconfiguration Management Avalon-MM Master Interface

Signal Name	Direction	Description
`reconfig_mif_address[31:0]`	Output	This is the Avalon-MM address. This is a byte address.
`reconfig_mif_read`	Output	When asserted, signals an Avalon-MM read request.
`reconfig_mif_readdata[15:0]`	Input	The read data.
`reconfig_mif_waitrequest`	Input	When asserted, indicates that the MIF Avalon-MM slave is not ready to respond to a read request.
`cal_busy_in`	Input	In Arria V and Cyclone V devices, acts as a status port for DCD calibration to prevent simultaneous DCD calibration for multiple channels on the same side of the device. This signal is only available when you select **Create optional calibration status ports**. If your design includes more than 1 Transceiver Reconfiguration Controller on the same side of the FPGA, you must daisy chain the `tx_cal_busy` output ports to the `cal_busy_in` input ports on the same side of the FPGA. Arria V devices require DCD calibration for channels with data rates equal to or greater than 4.9152 Gbps.

Transceiver Reconfiguration Interface

This section describes the signals that comprise the dynamic reconfiguration interface. The Transceiver Reconfiguration Controller communicates with the PHY IP cores using this interface. In the following table, <n> is the number of reconfiguration interfaces connected to the Transceiver Reconfiguration Controller.

Table 17-6: Transceiver Reconfiguration Interface

Signal Name	Direction	Description
`reconfig_to_xcvr[(<n>×70)-1:0]`	Output	Parallel reconfiguration bus from the Transceiver Reconfiguration Controller to the PHY IP Core.
`reconfig_from_xcvr[(<n>×46)-1:0]`	Input	Parallel reconfiguration bus from the PHY IP core to the Transceiver Reconfiguration Controller.

Signal Name	Direction	Description
reconfig_busy	Output	When asserted, indicates that a reconfiguration operation is in progress and no further reconfiguration operations should be performed. You can monitor this signal to determine the status of the Transceiver Reconfiguration Controller. Alternatively, you can monitor the busy bit of the control and status registers of any reconfiguration feature to determine the status of the Transceiver Reconfiguration Controller.
tx_cal_busy	Output	This optional signal is asserted while initial TX calibration is in progress and no further reconfiguration operations should be performed. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP. You can monitor this signal to determine the status of the Transceiver Reconfiguration Controller. Arria V devices require DCD calibration for channels with data rates equal to or greater than 4.9152 Gbps. In Arria V devices, you cannot run DCD calibration for multiple channels on the same side of a device simultaneously. If your design includes more than 1 Transceiver Reconfiguration Controller on a single side of the FPGA, you must daisy chain the tx_cal_busy output port to the next cal_busy_in input port on the same side of the FPGA.
rx_cal_busy	Output	This optional signal is asserted while initial RX calibration is in progress and no further reconfiguration operations should be performed. It is also asserted if reconfiguration controller is reset. It will not be asserted if you manually re-trigger the calibration IP. You can monitor this signal to determine the status of the Transceiver Reconfiguration Controller.

Reconfiguration Management Interface

This section describes the reconfiguration management interface.

The reconfiguration management interface is an Avalon-MM slave interface. You can use an embedded controller to drive this interface. Alternatively, you can use a finite state machine to control all Avalon-MM reads and writes to the Transceiver Reconfiguration Controller. This interface provides access to the Transceiver Reconfiguration Controller's Avalon-MM registers.

For more information about the Avalon-MM protocol, including timing diagrams, refer to the *Avalon Interface Specifications*.

Table 17-7: Reconfiguration Management Interface

Signal Name	Direction	Description
mgmt_clk_clk	Input	Avalon-MM clock input. The frequency range for the mgmt_clk_clk is 100-125 MHz for Stratix V and Arria V GZ devices. It is 75-125 MHz for Arria V devices. For Cyclone V devices, the frequency range is 75-125MHz if the Cyclone V Hard IP for PCI Express IP Core is not enabled. When the Hard IP for PCI Express is enabled, the frequency range is 75-100 MHz. Falling outside of the required frequency range may reduce the accuracy of the calibration functions. If your design includes the following components: • The Stratix V Hard IP for PCI Express with CvP enabled • Any additional transceiver PHY connected to the same Transceiver Reconfiguration Controller then you must connect the PLL reference clock which is called *refclk* in the Stratix V Hard IP for PCI Express IP Core to the *mgmt_clk_clk* signal of the Transceiver Reconfiguration Controller and the additional transceiver PHY. In addition, if your design includes more than one Transceiver Reconfiguration Controllers on the same side of the FPGA, they all must share the *mgmt_clk_clk* signal. **Note:** The frequency range depends on the device speed grade. Slower speed grade variants of Stratix V and Arria V GZ devices may require a 100 MHz reconfiguration clock to close timing.
mgmt_rst_reset	Input	This signal resets the Transceiver Reconfiguration Controller. This signal is active high and level sensitive. If the Transceiver Reconfiguration Controller IP Core connects to an Interlaken PHY IP Core, the Reconfiguration Controller IP Core mgmt_rst_reset must be simultaneously asserted with phy_mgmt_clk_reset to bring the Frame Generators in the link into alignment. Failure to meet to this requirement will result in excessive transmit lane-to-lane skew in the Interlaken link.
reconfig_mgmt_address[6:0]	Input	Avalon-MM address.
reconfig_mgmt_writedata[31:0]	Input	Input data.
reconfig_mgmt_readdata[31:0]	Output	Output data.
reconfig_mgmt_write	Input	Write signal. Active high.

Signal Name	Direction	Description
reconfig_mgmt_read	Input	Read signal. Active high.

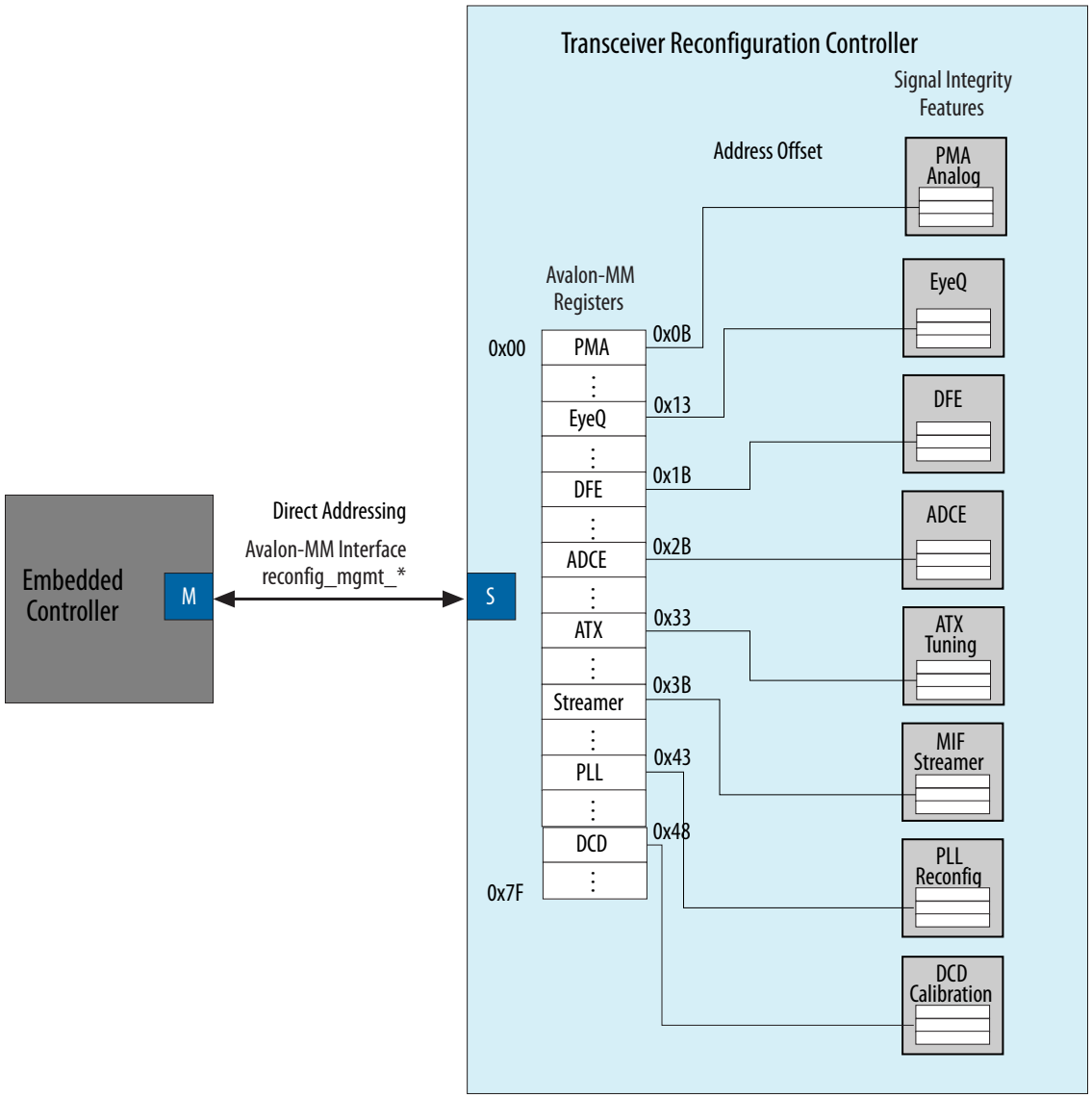
Related Information

Avalon Interface Specifications

Transceiver Reconfiguration Controller Memory Map

Each register-based feature has its own Avalon-MM address space within the Transceiver Reconfiguration Controller.

Figure 17-3: Memory Map of the Transceiver Reconfiguration Controller Registers



The following table lists the address range for the Transceiver Reconfiguration Controller and the reconfiguration and signal integrity modules. The Avalon-MM interface uses byte addresses.

Table 17-8: Transceiver Reconfiguration Controller Address Map

Address	Link
7'h08-7'h0C	PMA Analog Control Registers
7'h10-7'h14	EyeQ Registers
7'h18-7'h1C	DFE Registers
7'h28-7'h2C	AEQ Registers
7'h30-7'h34	ATX PLL Calibration Registers
7'h38-7'h3C	Streamer Module Registers
7'h40-7'h44	PLL Reconfiguration

Transceiver Reconfiguration Controller Calibration Functions

The Transceiver Reconfiguration Controller supports various calibration functions to enhance the performance and operation of any connected transceiver PHY IP core. Refer to Resource Utilization for Stratix V Devices for the resource utilization of these calibration functions.

Offset Cancellation

The offset cancellation function adjusts the offsets within the RX PMA and the CDR parameters for process variations to achieve optimal performance.

Offset cancellation runs only once upon power-up. The RX buffers are unavailable while this function is running. This calibration feature is run automatically and enabled by default.

Duty Cycle Calibration

The TX clocks generated by the CMU and travel across the clock network may introduce duty cycle distortion (DCD). DCD calibration function reduces this distortion.

DCD runs once during device power up and you can manually trigger DCD after power up. Altera recommends that you enable DCD for Arria V and Cyclone V devices if either of the following conditions is true:

- The data rate is greater than or equal to 4.9152 Gbps
- The design dynamically reconfigures the TX PLL and the data rate is greater than or equal to 4.9152 Gbps

Related Information

AN 676: Using the Arria V and Cyclone V Reconfiguration Controller to Perform Dynamic Reconfiguration

Auxiliary Transmit (ATX) PLL Calibration

ATX calibration tunes the parameters of the ATX PLL for optimal performance. This function runs once after power up. You can rerun this function by writing into the appropriate memory-mapped registers.

The RX buffer is unavailable while this function is running. You should run the ATX calibration after reconfiguring the PLL. You may need to rerun ATX calibration if you reset an ATX PLL and it does not lock after the specified lock time.

For more information about the Auxiliary Transmit (ATX) PLL Calibration refer to ATX PLL Calibration Registers.

Refer to the *Parameterizing the Transceiver Reconfiguration Controller IP Core in the MegaWizard Plug-In Manager* section for information about how to enabled these functions.

Note: If you are using a PHY IP with DFE enabled with a reconfiguration controller and/or if you are using ATX PLLs in your design, then the reference clock to the PHY IP must be stable before the reconfiguration controller is taken out of reset state.

Transceiver Reconfiguration Controller PMA Analog Control Registers

You can use the Transceiver Reconfiguration Controller to reconfigure the following analog controls:

- Differential output voltage (V_{OD})
- Pre-emphasis taps
- Receiver equalization control
- Receiver equalization DC gain
- Reverse serial loopback

Note: All undefined register bits are reserved.

Table 17-9: PMA Analog Registers

Reconfig Addr	Bits	R/W	Register Name	Description
7'h08	[9:0]	RW	`logical channel number`	The logical channel number. Must be specified when performing dynamic updates. The Transceiver Reconfiguration Controller maps the logical address to the physical address.

Reconfig Addr	Bits	R/W	Register Name	Description
7'h0A	[9]	R	control and status	Error. When asserted, indicates an error. This bit is asserted if any of the following conditions occur: • The channel address is invalid. • The PHY address is invalid. • The PMA offset is invalid.
	[8]	R		Busy. When asserted, indicates that a reconfiguration operation is in progress.
	[1]	W		Read. Writing a 1 to this bit triggers a read operation.
	[0]	W		Write. Writing a 1 to this bit triggers a write operation.
7'h0B	[5:0]	RW	pma offset	Specifies the offset of the PMA analog setting to be reconfigured. Table 17-10 describes the valid offset values.
7'h0C	[6:0]	RW	data	Reconfiguration data for the PMA analog settings. Refer to Table 17-10 for valid data values.

Refer to the *Arria V Device Datasheet*, the *Cyclone V Device Datasheet*, or the *Stratix V Device Datasheet* for more information about the electrical characteristics of each device. The final values are currently pending full characterization of the silicon.

Note: All undefined register bits are reserved.

Table 17-10: PMA Offsets and Values

Offset	Bits	R/W	Register Name	Description
0x0	[5:0]	RW	V_{OD}	VOD. The following encodings are defined: • 6'b000000:6'b111111:0–63
0x1	[4:0]	RW	Pre-emphasis pre-tap	The following encodings are defined: • 5'b00000 and 5'b10000: 0 • 5'b00001–5'b01111: -15 to -1 • 5'b10001–5b'11111: 1 to 15
0x2	[4:0]	RW	Pre-emphasis first post-tap	The following encodings are defined: • 5'b00000–5'b11111: 0–31

Offset	Bits	R/W	Register Name	Description
0x3	[4:0]	RW	Pre-emphasis second post-tap	The following encodings are defined: • 5'b00000 and 5'b10000: 0 • 5'b00001–5'b01111: -15 to -1 • 5'b10001–5b'11111: 1 to 15
0x10	[2:0]	RW	RX equalization DC gain(18)	The following encodings are defined: • 3'b000–3'b011:0–3 • 3'b100-3'b110:4 • 3'b111:Reserved
0x11	[3:0]	RW	RX equalization control	The following encodings are defined: • 4'b0000–4'b1111: 0–15
0x20	[0]	WO	Pre-CDR Reverse Serial Loopback	Writing a 1 to this bit enables reverse serial loopback. Writing a 0 disables pre-CDR reverse serial loopback.
0x21	[0]	WO	Post-CDR Reverse Serial Loopback	Writing a 1 to this bit enables post-CDR reverse serial loopback. Writing a 0 disables post-CDR reverse serial loopback.

Refer to *Changing Transceiver Settings Using Register-Based Reconfiguration* and *Changing Transceiver Settings Using Streamer-Based Reconfiguration* for the procedures you can use to update PMA settings.

Related Information

- **Arria V Device Datasheet**
- **Cyclone V Device Datasheet**
- **Stratix V Device Datasheet**
- **Application Note 645: Dynamic Reconfiguration of PMA Controls in Stratix V Devices**

Transceiver Reconfiguration Controller EyeQ Registers

EyeQ is a debug and diagnostic tool that analyzes the incoming data, including the receiver's gain, noise level, and jitter after the receive buffer. EyeQ is only available for Stratix V and Arria V GZ devices.

(18) There are two possible methods to modify the RX linear equalization settings:

- Using the reconfiguration controller (offset 0x11)
- Using the QSF assignments

Different values are used for each method. The settings for using the reconfiguration controller range from 0 to 15 and the settings for using the QSF assignments are from 1 to 16. For example, setting 0 in the transceiver reconfiguration controller corresponds to setting 1 for the QSF assignments and so forth.

EyeQ uses a phase interpolator and sampler to estimate the vertical and horizontal eye opening using the values that you specify for the horizontal phase and vertical height as described in the **Table 17-12**table. The phase interpolator generates a sampling clock and the sampler examines the data from the sampler output. As the phase interpolator output clock phase is shifted by small increments, the data error rate goes from high to low to high if the receiver is good. The number of steps of valid data is defined as the width of the eye. If none of the steps yields valid data, the width of the eye is equal to 0, which means the eye is closed.

When the Bit Error Rate Block (BERB) is not enabled, the sampled data is deserialized and sent to the IP core; the PRBS checker determines the Bit Error Rate (BER). When the BER Block is enabled, the Bit checker determines the BER by comparing the sampled data to the CDR sampled data.

Note: If you are using the EyeQ monitor with DFE enabled, you must put the EyeQ monitor in 1D mode by writing the EyeQ 1D-eye bit. For more information, refer to the **Table 17-12** table . The EyeQ path is designed to measure the sampled eye margin. To estimate the pre-CDR eye opening using the measured eye margin data, you can add 10ps to the measured eye margin value for RX input signals with moderate amounts of jitter which is typical in most data streams.

The following table lists the memory-mapped EyeQ registers that you can access using Avalon-MM reads and writes on reconfiguration management interface.

Note: All channels connected to same Transceiver Reconfiguration Controller IP Core share one set of bit error rate block counters. You can monitor one channel at a time. If Transceiver Reconfiguration Controller is interrupted by other operations, such as channel switching or AEQ, the bit error rate data will be corrupted.

Note: All undefined register bits are reserved.

Table 17-11: Eye Monitor Registers

Note: The default value for all the register bits mentioned in this table is 0.

Reconfig Addr	Bits	R/W	Register Name	Description
7'h10	[9:0]	RW	`logical channel number`	The logical channel number. Must be specified when performing dynamic updates. The Transceiver Reconfiguration Controller maps the logical address to the physical address.
7'h12	[9]	R	`control and status`	`Error`.When asserted, indicates an invalid channel or address.
	[8]	R		`Busy`. When asserted, indicates that a reconfiguration operation is in progress.
	[1]	W		`Read`. Writing a 1 to this bit triggers a read operation.
	[0]	W		`Write`. Writing a 1 to this bit triggers a write operation.
7'h13	[5:0]	RW	`eyeq offset`	Specifies the 6-bit offset of the EyeQ register.

Reconfig Addr	Bits	R/W	Register Name	Description
7'h14	[15:0]	RW	data	Reconfiguration data for the transceiver PHY registers.

Note: All undefined register bits are reserved.

Table 17-12: EyeQ Offsets and Values

Note: The default value for all the register bits mentioned in this table is 0.

Offset	Bits	R/W	Register Name	Description
0x0	[4:3]	RW	BERB Snap Shot and Reset	Only available when you turn on the **Enable Bit Error Rate Block** in the Transceiver Reconfiguration Controller IP Core GUI. The following encodings are defined: • 2'b00: Reserved. • 2'b01: Reset everything, snapshot and counters are reset to 0. • 2'b10: Take a snapshot. Copy the counter values into local registers for read access. These values are not updated until another snapshot is taken. • 2'b11: Snapshot and reset. Take a snapshot of the counter values. Reset the counters and leave the snap shot untouched.
	[2]	RW	Counter Enable	Only available when you turn on the **Enable Bit Error Rate Block** in the Transceiver Reconfiguration Controller IP Core GUI. When set to 1, the counters accumulate bits and errors. When set to 0, pauses accumulation, preserving the current values.
	[1]	RW	BERB Enable	Only available when you turn on the **Enable Bit Error Rate Block** in the Transceiver Reconfiguration Controller IP Core GUI. When set to 1, enables the BER. When set to 0, disables the BER counters and the bit checker.
	[0]	RW	Enable Eye Monitor	Writing a 1 to this bit enables the Eye monitor.
0x1	[5:0]	RW	Horizontal phase	Taken together, the Horizontal phaseand vertical height specify the Cartesian x-y coordinates of the sample point on the eye diagram. You can increment through 64 phases over 2 UI on the horizontal axis.

Offset	Bits	R/W	Register Name	Description
0x2	[5:0]	RW	`Vertical height`	Taken together, the `horizontal phase` and `vertical height` specify the Cartesian x-y coordinates of the sample point on the eye diagram. You can specify 64 heights on the vertical axis.
0x3	[15:4]	RMW	Reserved	You should not modify these bits. To update this register, first read the value of this register then change only the value for bits that are not reserved.
	[13]	RW	`1D-Eye`	Writing a 1 to this bit selects 1D Eye mode and disables vertical height measurement. Writing a 0 to this bit selects normal 2D Eye measurement mode including both the horizontal and vertical axes. You must use 1D Eye mode if you have enabled DFE.
	[12:3]	RMW	Reserved	You should not modify these bits. To update this register, first read the value of this register then change only the value for bits that are not reserved.
	[2]	RW	`Polarity` [(19)]	Specifies the sign of the `Vertical height` . When 0, the `Vertical height` is negative. When 1, the `Vertical height` is positive.
	[1:0]	RMW	Reserved	You should not modify these bits. To update this register, first read the value of this register then change only the value for bits that are not reserved.
0x5	[31:0]	R	`Bit Counter[31:0]`	Only valid when the `BERB Enable` and `Counter Enable` bits are set. `Bit Counter[63:0]` reports the total number of bits received since you enabled or reset BER counters. Each increment represents 256 bits.
0x6	[31:0]	R	`Bit Counter[63:32]`	
0x7	[31:0]	R	`Err Counter[31:0]`	Only available when the `BERB Enable` and `Counter Enable` bits are set. `Err Counter[63:0]` reports the total number of error bits received since you enabled or reset BER counters.
0x8	[31:0]	R	`Err Conter[63:32]`	

Refer to Changing Transceiver Settings Using Register-Based Reconfiguration for the procedures you can use to control the Eye Monitor.

(19) Writing a 1 to the `Enable Eye Monitor` register will reset the polarity to be positive.

EyeQ Usage Example

This section provides an example of accessing the EyeQ registers and using the Bit Error Rate Block (BERB).

When the BERB is enabled, the serial bit checker compares the data from CDR path and EyeQ path. The BERB accumulates the total received bit numbers and the error bit numbers. You can use the BERB block as a diagnostic tool to perform in-system link analysis without interrupting the link traffic. The steps below provide BERB operation example:

- Write 3'b111 to bit[2:0] in offset 0x0 to enable BERB
- Set Horizontal Phase and/or Vertical High in offset 0x1 and/or 0x2
- Set 2'b01 to bit[4:3] in offset 0x0 to reset the counters
- Set 2'b10 to bit [4:3] in offset 0x0 to take a snapshot of the counters. Read the counter values from offsets 0x5 to 0x8.
- Repeat steps 2 to 4 to measure the bit error rate (BER) for another horizontal phase / veritical height.

Transceiver Reconfiguration Controller DFE Registers

The DFE is an infinite impulse response filter (non-linear) that compensates for inter-symbol interference (ISI). Because the values of symbols previously detected are known, the DFE engine can estimate the ISI contributed by these symbols and cancel out this ISI by subtracting the predicted value from subsequent symbols.

This mechanism allows DFE to boost the signal to noise ratio of the received data. You can use DFE in conjunction with the receiver's linear equalization and with the transmitter's pre-emphasis feature. DFE is supported by Arria V GZ and Stratix V devices.

DFE automatically runs offset calibration and phase interpolator (PI) phase calibration on all channels after power up. You can run DFE manually to determine the optimal settings by monitoring the BER of the received data at each setting and specify the DFE settings that yield the widest eye.

Note: If you are using the EyeQ monitor with DFE enabled, you must put the EyeQ monitor in 1D mode by writing the EyeQ 1D-eye bit. For more information, refer to EyeQ Offsets and Values.

Note: If you are using a PHY IP that has DFE enabled with a reconfiguration controller and/or if you are using ATX PLLs in your design, then the reference clock to the PHY IP must be stable before the reconfiguration controller is taken out of reset state.

The following table lists the direct DFE registers that you can access using Avalon-MM reads and writes on reconfiguration management interface.

Note: All undefined register bits are reserved.

Table 17-13: DFE Registers

Reconfig Addr	Bits	R/W	Register Name	Description
7'h18	[9:0]	RW	`logical channel address`	The logical channel address. Must be specified when performing dynamic updates. The Transceiver Reconfiguration Controller maps the logical address to the physical address.

Reconfig Addr	Bits	R/W	Register Name	Description
7'h1A	[9]	R	control and status	Error.When asserted, indicates an invalid channel or address.
	[8]	R		Busy. When asserted, indicates that a reconfiguration operation is in progress.
	[1]	W		Read. Writing a 1 to this bit triggers a read operation.
	[0]	W		Write. Writing a 1 to this bit triggers a write operation.
7'h1B	[5:0]	RW	dfe_offset	Specifies the 6-bit offset of the DFE register.
7'h1C	[15:0]	RW	data	Reconfiguration data for the transceiver PHY registers.

The following table describes the DFE registers that you can access to change DFE settings.

Note: All undefined register bits are reserved.

Table 17-14: DFE Offset and Values

Offset	Bits	R/W	Register Name	Description
0x0	[1]	RW	power on	Writing a 0 to this bit powers down DFE in the channel specified.
	[0]	RW	adaptation engine enable	Writing a 1 enables the adaptive equalization engine.
0x1	[3:0]	RW	tap 1	Specifies the coefficient for the first post tap. The valid range is 0–15.
0x2	[3]	RW	tap 2 polarity	Specifies the polarity of the second post tap as follows: • 0: negative polarity • 1: positive polarity
	[2:0]	RW	tap 2	Specifies the coefficient for the second post tap. The valid range is 0–7.
0x3	[3]	RW	tap 3 polarity	Specifies the polarity of the third post tap as follows: • 0: negative polarity • 1: positive polarity
	[2:0]	RW	tap 3	Specifies the coefficient for the third post tap. The valid range is 0–7.

Offset	Bits	R/W	Register Name	Description
0x4	[3]	RW	`tap 4 polarity`	Specifies the polarity of the fourth post tap as follows: • 0: negative polarity • 1: positive polarity
	[2:0]	RW	`tap 4`	Specifies the coefficient for the fourth post tap. The valid range is 0–7.
0x5	[3]	RW	`tap 5 polarity`	Specifies the polarity of the fifth post tap as follows: • 0: negative polarity • 1: positive polarity
	[2:0]	RW	`tap 5`	Specifies the coefficient for the fifth post tap. The valid range is 0–3.
0xB	[0]	RW	`DFE_adaptation`	Writing a 0 or 1 to this bit turns on the DFE power and initiates triggered DFE mode for the specified channel. Ensure busy bit is 0 to complete the reconfiguration process. Reading (0xB) register bit as 1 and busy bit as 0, indicates that the DFE is in triggered mode. To turn off the triggered DFE mode, write 0 to bit 1 of register 0x0.

Controlling DFE Using Register-Based Reconfiguration

In register-based mode, you use a sequence of Avalon-MM reads and writes to configure the DFE and to turn it on and off. There are three ways to control the DFE using a sequence of register-based reconfiguration reads and writes.

Turning on DFE Continuous Adaptive mode

Complete the following steps to turn on DFE continuous adaptive mode:

1. Read the DFE control and status register busy bit (bit 8) until it is clear.
2. Write the logical channel number of the channel to be updated to the DFE logical channel number register.
3. Write the `DFE_offset` address to 0x0.
4. Write the data value 2'b11 to the data register. This data powers on DFE and enables the DFE continuous adaptation engine.
5. Write the `control and status` register write bit to 1'b1.
6. Read the `control and status` register `busy` bit. Continue to read the `busy` bit while its value is 1b'1.
7. When busy = 1'b0, the Transceiver Reconfiguration Controller has updated the logical channel specified in Step 2 with the data specified in Steps 3 and 4.

The register-based write to turn on continuous adaptive DFE for logical channel 0 is as shown in the following example:

Example 17-1: Register-Based Write To Turn On Adaptive DFE for Logical Channel 0

```
#Setting logical channel 0
write_32 0x18 0x0

#Setting DFE offset to 0x0
write_32 0x1B 0x0

#Setting data register to 3
write_32 0x1C 0x3

#Writing the data to turn on adaptive DFE
write_32 0x1A 0x1
```

Turning on Triggered DFE Mode

Complete the following steps to turn on triggered DFE mode:

1. Read the DFE `control and status` register busy bit (bit 8) until it is clear.
2. Write the logical channel number of the channel to be updated to the DFE `logical channel number` register.
3. Write the `DFE_offset` address of 0xB.
4. Write the data value 1'b1 or 1'b0 to the `data` register.
5. Write the `control and status` register write bit to 1'b1.
   This turns on DFE power and initiates triggered DFE mode.
6. Read the DFE `control and status` register `busy` bit (bit 8) until it is clear.
7. When `busy` equals 1b'0, the Transceiver Reconfiguration Controller has updated the `logical channel` specified in Step 2 with the data specified in Steps 3 and 4.

The register-based write to turn on the triggered DFE mode for logical channel 0 is shown in the following example:

Example 17-2: Register-Based Write To Turn On Triggered DFE Mode for Logical Channel 0

```
#Setting logical channel 0
write_32 0x18 0x0

#Setting DFE offset to 0xB
write_32 0x1B 0xB

#Setting data register to 1
write_32 0x1C 0x1

#Writing the data to turn on triggered DFE
write_32 0x1A 0x1
```

Setting the First Tap Value Using DFE in Manual Mode

Complete the following steps to use DFE in Manual mode and set first DFE tap value to 5:

1. Read the DFE `control and status` register busy bit (bit 8) until it is clear.
2. Write the logical channel number of the channel to be updated to the DFE `logical channel` number register.
3. Write the `DFE_offset` address of 0x0 (DFE control register).
4. Write the data value 2'b10 to the data register to enable DFE power.
   This powers up the DFE and DFE adaptation engine is disabled.
5. Write the `control and status` register write bit to 1'b1.
6. Read the DFE `control and status` register busy bit (bit 8) until it is clear.
7. Write the `DFE_offset` address of 0x1 (DFE Tap 1 register).
8. Write the `data` value 3'b101 to the data register.
9. Write the `control and status` register write bit to 1'b1.
10. Read the `control and status` register busy bit. Continue to read the `busy` bit while its value is 1‘b1.
11. When `busy` equals 1b’0, the Transceiver Reconfiguration Controller has updated the logical channel.

The register-based write to use DFE in manual mode and set the first DFE tap value to 5 for logical channel 0 as shown in the following example:

Example 17-3: Register-Based Write To Use DFE in Manual Mode and Set the First DFE Tap Value to 5 for Logical Channel 0

```
#Setting logical channel 0
write_32 0x18 0x0

#Setting DFE offset to 0x0
write_32 0x1B 0x0

#Setting data register to 2
write_32 0x1C 0x2

#Writing the data to use DFE in Manual mode
write_32 0x1A 0x1

#Setting DFE offset to 0x1
write_32 0x1B 0x1

#Setting data register to 5
write_32 0x1C 0x5

#Writing the data to set DFE 1st tap value to 5
write_32 0x1A 0x1
```

Transceiver Reconfiguration Controller AEQ Registers

Adaptive equalization compensates for backplane losses and dispersion which degrade signal quality.

AEQ can be run once to help control the four-stage continuous time linear equalizer (CTLE), which is a manual tool that compensates for backplane losses and dispersion.

The following table lists the direct AEQ registers that you can access using Avalon-MM reads and writes on reconfiguration management interface.

Note: All undefined register bits are reserved.

Table 17-15: AEQ Registers

Reconfig Addr	Bits	R/W	Register Name	Description
7'h28	[9:0]	RW	`logical channel number`	The logical channel number of the AEQ hardware to be accessed. Must be specified when performing dynamic updates. The Transceiver Reconfiguration Controller maps the logical address to the physical address.
7'h2A	[9]	R	`control and status`	`Error`.When asserted, indicates an error. This bit is asserted when the channel address is invalid.
	[8]	R		`Busy`. When asserted, indicates that a reconfiguration operation is in progress.
	[1]	W		`Read`. Writing a 1 to this bit triggers a read operation.
	[0]	W		`Write`. Writing a 1 to this bit triggers a write operation.
7'h2B	[3:0]	RW	`aeq_offset`	Specifies the address of the AEQ register to be read or written. Refer to **Table 17-16** for details.
7'h2C	[15:0]	RW	`data`	Specifies the read or write data.

The following table describes the AEQ registers that you can access to change AEQ settings.

Note: All undefined register bits are reserved.

Table 17-16: AEQ Offsets and Values

Offset	Bits	R/W	Register Name	Description	Default Value
0x0	[8]	R	adapt_done	When asserted, indicates that adaptation has completed. In One-Time Adaptation Mode, AEQ stops searching new EQ settings even if the signal quality of incoming serial data is inadequate. For some extreme cases, when the channel loss is too much for AEQ to compensate, the adapt_done signal may never be asserted. The AEQ engine can take up to 50,000 reconfiguration clock cycles before selecting the final equalization settings.	1b'0
	[1:0]	RW	mode	Specifies the following address modes: • 2'b00: Low power manual equalization mode • 2'b01: One-time AEQ adaptation at power up • 2'b11: Reserved	2'b00
0x1	[3:0]	R	equalization results	This is the value set by the automatic AEQ adaptation performed at startup. If you choose to perform manual equalization using the linear equalizer, you can use this value as a reference. Although automatic and manual equalization do not provide identical functionality, specifying this value enables manual equalization to approximate the original setting.	4'b0000

Refer to *Changing Transceiver Settings Using Register-Based Reconfiguration* for the procedures you can use to control AEQ.

Transceiver Reconfiguration Controller ATX PLL Calibration Registers

The ATX PLL Calibration registers allow you to rerun ATX calibration after power up. The Transceiver Reconfiguration Controller automatically runs ATX calibration at power up.

Note: You may need to rerun ATX calibration if you reset an ATX PLL and it does not lock after the specified lock time.

The following table lists the direct access ATX registers that you can access using Avalon-MM reads and writes on reconfiguration management interface.

Note: All undefined register bits are reserved.

Table 17-17: ATX Tuning Registers

ATX Addr	Bits	R/W	Register Name	Description
7'h30	[9:0]	RW	logical channel number	The logical channel number. The Transceiver Reconfiguration Controller maps the logical address to the physical address.
7'h32	[9]	R	control and status	Error. When asserted, indicates an invalid channel or address. This bit is asserted after a write operation if the selected logical channel number selects a logical channel interface that is not connected to an ATX PLL. It is also be asserted if the tuning algorithm failed to converge on a working setting after a manual calibration.
	[8]	R		Busy. When asserted, indicates that a reconfiguration operation is in progress.
	[1]	W		Read. Writing a 1 to this bit triggers a read operation.
	[0]	W		Write. Writing a 1 to this bit triggers a write operation.
7'h33	[3:0]	RW	atx_offset	Specifies the 4-bit register address used for indirect accesses on the reconfiguration bus. Refer to **Table 17-18** for offsets and values.
7'h34	[15:0]	RW	data	Reconfiguration data for the transceiver PHY registers.

Table 17-18: ATX PLL Tuning Offsets and Values

Offset	Bits	R/W	Register Name	Description
0x0	[0]	RW	Control	Writing a 1 to this bit triggers ATX PLL calibration. This register self-clears. Unused bits of this register must be set to 0. The tx_cal_busy signal is asserted at initial runtime or if you reset the reconfiguration controller. It is not asserted if you manually re-trigger the ATX PLL calibration process. Writing a 1 to this bit will not trigger ATX PLL calibration if the PLL is already locked.
0x1	[1]	RW		Writing a 1 to this bit triggers the ATX PLL calibration even if the PLL is already locked.

Refer to Changing Transceiver Settings Using Register-Based Reconfiguration for the procedures you can use to control ATX tuning.

Transceiver Reconfiguration Controller PLL Reconfiguration

You can use the PLL reconfiguration registers to change the reference clock input to the TX PLL or the clock data recovery (CDR) circuitry.

The PLL registers for dynamic reconfiguration feature are available when you select one of the following transceiver PHY IP cores:

- Custom PHY IP Core
- Low Latency PHY IP Core
- Deterministic Latency PHY IP Core
- Arria V, Arria V GZ, Cyclone V, and Stratix V Native PHYs

You can establish the number of possible PLL configurations on the **Reconfiguration** tab of the appropriate transceiver PHY IP core. The **Reconfiguration** tab allows you to specify up to five input reference clocks and up to four TX PLLs. You can also change the input clock source to the CDR PLL; up to five input clock sources are possible. If you plan to dynamically reconfigure the PLLs in your design, you must also enable `Allow PLL Reconfiguration` and specify the **Main TX PLL logical index** which is the PLL that the Intel Quartus Prime software instantiates at power up. The following figures illustrates these parameters.

Figure 17-4: Reconfiguration Tab of Custom, Low Latency, and Deterministic Latency Transceiver PHYs

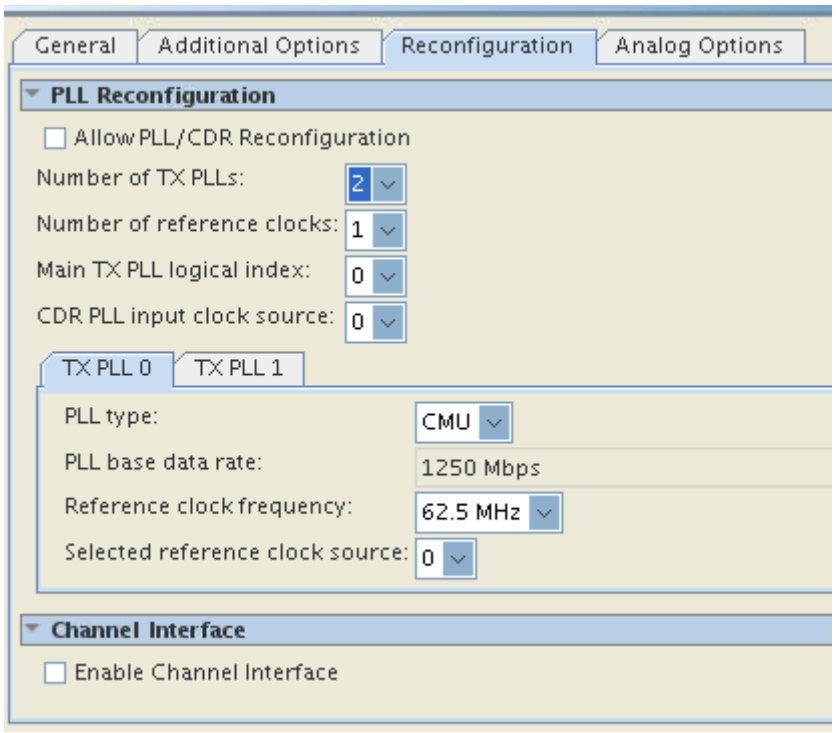
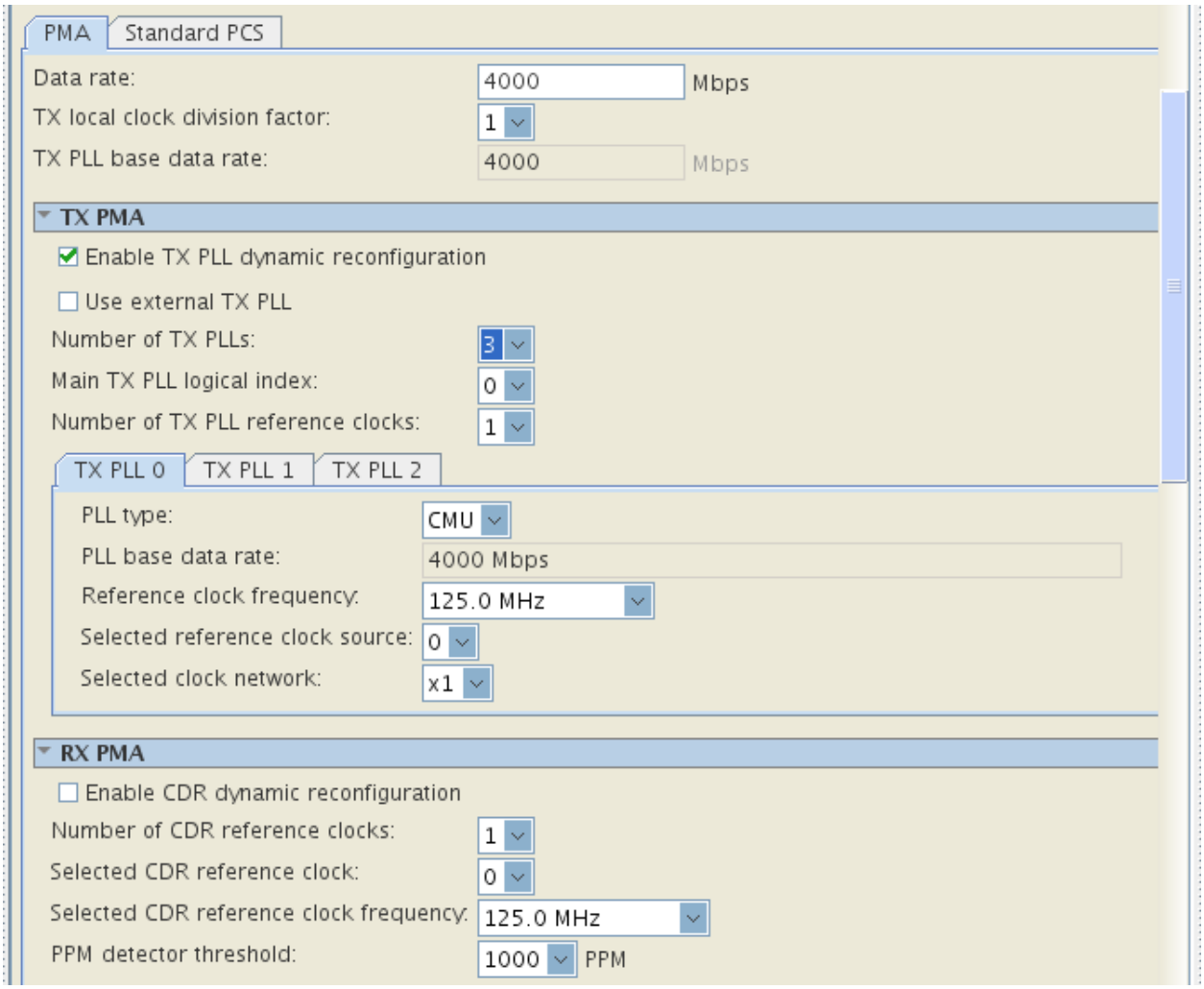


Figure 17-5: Reconfiguration Tab of Native Transceiver PHYs



Note: If you dynamically reconfigure PLLs, you must provide your own reset logic by including the Altera Reset Controller IP Core or your own custom reset logic in your design. For more information about the Altera-provided reset controller, refer to Chapter 17, Transceiver PHY Reset Controller IP Core.

For more information about the Stratix V reset sequence, refer to *Transceiver Reset Control in Stratix V Devices* in volume 2 of the *Stratix V Device Handbook*. For Arria V devices, refer to *Transceiver Reset Control and Power-Down in Arria V Devices*. For Cyclone V devices refer to *Transceiver Reset Control and Power Down in Cyclone V Devices*.

When you specify multiple PLLs, you must use the QSF assignment, `XCVR_TX_PLL_RECONFIG_GROUP`, to identify the PLLs within a reconfiguration group using the Assignment Editor. The `XCVR_TX_PLL_RECONFIG_GROUP` assignment identifies PLLs that the Intel Quartus Prime Fitter can merge. You can assign TX PLLs from different transceiver PHY IP core instances to the same group.

Note: You must create the `XCVR_TX_PLL_RECONFIG_GROUP` even if one transceiver PHY IP core instance instantiates multiple TX PLLs.

Related Information

- **Transceiver Reset Control in Stratix V Devices**
- **Transceiver Reset Control and Power-Down in Arria V Devices**
- **Transceiver Reset Control and Power Down in Cyclone V Devices**

Transceiver Reconfiguration Controller PLL Reconfiguration Registers

Lists the PLL reconfiguration registers that you can access using Avalon-MM read and write commands on reconfiguration management interface.

Note: All undefined register bits are reserved.

Table 17-19: PLL Reconfiguration Registers

Reconfig Addr	Bits	R/W	Register Name	Description
7'h40	[9:0]	RW	`logical channel number`	The logical channel number. Must be specified when performing dynamic updates. The Transceiver Reconfiguration Controller maps the logical address to the physical address. When reconfiguring the reference clock for the TX PLL you must specify the PLL's logical channel number. When reconfiguring the reference clock for the CDR you must specify the channel's logical channel number.
7'h42	[9]	R	`control and status`	When asserted, indicates an error. This bit is asserted if any of the following conditions occur: • The channel address is invalid. • The PHY address is invalid. • The address offset is invalid.
	[8]	R		`MIF Busy`. When asserted, indicates that a reconfiguration operation is in progress.
	[1]	W		`Read`. Writing a 1 to this bit triggers a read operation.
	[0]	W		`Write`. Writing a 1 to this bit triggers a write operation.
7'h43	[3:0]	RW	`pll_offset`	Specifies the 4-bit register address used for indirect to the PLL registers on the reconfiguration bus. Refer to Table 16–21 for offsets and values.

Reconfig Addr	Bits	R/W	Register Name	Description
7'h44	[15:0]	RW	data	Specifies the read or write data.

Note: All undefined register bits are reserved.

Table 17-20: PLL Reconfiguration Offsets and Values

Offset	Bits	R/W	Name	Description
0x0	[2:0]	RW	logical refclk selection	When written initiates reference clock change to the logical reference clock indexed by bits [2:0]. This index refers to the **Number of input clocks** on the **Reconfiguration** tab. You can specify up to 5 input clocks. When performing a reference clock switch for an ATX PLL you must stream in an ATX MIF. This offset is used to switch the reference clock for CMU PLLs. To perform a reference clock switch for ATX PLLs use MIF mode 0 and stream the ATX PLL MIF.
0x1	[2:0]	RW	logical PLL selection	When written initiates a clock generation block (CGB) switch to logical PLL indexed by bits [2:0]. This index refers to the **Number of TX PLLs** selected on the **Reconfiguration** tab. You can specify up to 4 input clocks. If you set the **Main TX PLL logical index** to 0, the Intel Quartus Prime software initializes your design using the first PLL defined.
0x2	[24:0]	RO	refclk physical mapping	Specifies the logical to physical refclk for current logical channel.
0x3	[14:0]	RO	PLL physical mapping	Specifies the logical to physical clock generation block word for current logical channel.

Transceiver Reconfiguration Controller DCD Calibration Registers

DCD runs automatically at power up. After power up, you can rerun DCD by writing to the DCD control register. Altera recommends that you run DCD calibration for Arria V and Cyclone V devices if the data rate is greater than 4.9152 Gbps.

Note: All undefined register bits are reserved.

Table 17-21: DCD Registers

Reconfig Addr	Bits	R/W	Register Name	Description
7'h48	[9:0]	RW	`logical channel number`	The logical channel number. Must be specified when performing dynamic updates. The Transceiver Reconfiguration Controller maps the logical address to the physical address.
7'h4B	[6:0]	RW	`dcd offset`	Specifies the offset of the DCD setting to be reconfigured. **#nik1398984276755/table_A420703A5C8042E3872CA7C66487930E** describes the valid offset values.
7'h4C	[6:0]	RW	`dcd_data`	Reconfiguration data for the PMA analog settings.

Note: All undefined register bits are reserved.

Table 17-22: DCD Offsets and Values

Offset	Bits	R/W	Register Name	Description
0x0	[5:0]	RW	dcd_control	Writing 1'b1 to this bit to manually triggers DCD calibration.

Transceiver Reconfiguration Controller Channel and PLL Reconfiguration

You can use channel and PLL reconfiguration to dynamically reconfigure the channel and PLL settings in a transceiver PHY IP core.

Among the settings that you can change dynamically are the data rate and interface width. Refer to *Device Support for Dynamic Reconfiguration* for specific information about reconfiguration in Arria V, Cyclone V, and Stratix V devices.

The Transceiver Reconfiguration Controller's Streamer Module implements channel and PLL reconfiguration. Refer to the *Streamer Module Registers* for more information about this module.

Note: Channel and PLL reconfiguration are available for the Custom, Low Latency, Deterministic Latency PHY IP Cores, the Arria V Native PHY, the Arria V GZ Native PHY, the Cyclone V Native PHY, and the Stratix V Native PHY.

Channel Reconfiguration

If you turn on **Enable channel/PLL reconfiguration** in the Transceiver Reconfiguration Controller GUI, you can change the following channel settings:

- TX PMA settings
- RX PMA settings
- RX CDR input clock
- Reference clock inputs
- FPGA fabric transceiver width

When you select **Enable Channel Interface**, in the Custom, Low Latency, Deterministic Latency Transceiver PHY GUIs, the default width of the FPGA fabric to transceiver interface increases for both the **Standard** and **10G** datapaths as follows:

- Standard datapath—The TX interface is 44 bits. The RX interface is 64 bits.
- 10G datapath— TX only, RX only, and duplex channels are all 64 bits.

However, depending upon the FPGA fabric transceiver width specified, only a subset of the 64 bits may carry valid data. Specifically, in the wider bus, only the lower <n> bits are used, where <n> is equal to the width of the FPGA fabric width specified in the transceiver PHY IP core. The following table illustrates this point for the 10G datapath, showing three examples where the FPGA fabric interface width is less than 64 bits.

Table 17-23: Channel Reconfiguration Bit Ordering

Number of Lanes	Specified FPGA Fabric Width (Total Bits)	Default Channel Width (Total Bits)	Used Bits
1	32 bits (32 bits)	64 bits/lane (64 bits)	Lane 0: [31:0]
2	40 bits (80 bits)	64 bits/lane (128 bits)	Lane 0: [39:0] Lane 1: [103:64]
3	40 bits (120 bits)	64 bits/lane (192 bits)	Lane 0: [39:0] Lane 1: [103:64] Lane 2: [167:128]

PLL Reconfiguration

If you turn on **Enable PLL reconfiguration support block** in the Transceiver Reconfiguration Controller GUI, you can change the following channel settings:

- TX PLL settings
- TX PLL selection

Note: When you specify multiple PLLs, you must use the QSF assignment, `XCVR_TX_PLL_RECONFIG_GROUP`, to identify the PLLs within a reconfiguration group. The `XCVR_TX_PLL_RECONFIG_GROUP` assignment identifies PLLs that the Intel Quartus Prime Fitter can merge.

Transceiver Reconfiguration Controller Streamer Module Registers

The Streamer module defines the following two modes for channel and PLL reconfiguration:

- Mode 0—MIF. Uses a memory initialization file (.mif) to reconfigure settings.
- Mode 1—Direct Write. Uses a series of Avalon-MM writes on the reconfiguration management interface to change settings.

Note: All undefined register bits are reserved.

Table 17-24: Streamer Module Registers

PHY Addr	Bits	R/W	Register Name	Description
7'h38	[9:0]	RW	`logical channel number`	The logical channel number. Must be specified when performing dynamic updates. The Transceiver Reconfiguration Controller maps the logical address to the physical address.
7'h3A	[9]	R	`control and status`	`Error`. When asserted, indicates an error. This bit is asserted if any of the following conditions occur: • The channel address is invalid. • The PHY address is invalid. • The offset register address is invalid.
	[8]	R		`Busy`. When asserted, indicates that a reconfiguration operation is in progress.

PHY Addr	Bits	R/W	Register Name	Description

	[3:2]	RW		`Mode`. The following encodings are defined:
				• 2'b00: MIF. This mode continuously reads and transfers a .mif file, which contains the reconfiguration data.
				• 2'b01: Direct Write. In this mode, you specify a logical channel, a register offset, and data. Depending on the logical channel specified, the Transceiver Reconfiguration Controller may mask some of the data specified to prevent read-only values that were optimized during startup, from being over-written. In particular, this mode protects the following settings:
				  • Decision feedback equalization controls
				  • RX buffer offset calibration adjustments
				  • Duty cycle distortion adjustments
				  • PMA clock settings
				• 2'b10: Reserved
				• 2'b11: Reserved

	[1]	W		`Read`. Writing a 1 to this bit triggers a read operation. This bit is self clearing.
	[0]	W		`Write`. Writing a 1 to this bit triggers a write operation. This bit is self clearing.
7'h3B	[15:0]	RW	`streamer offset`	When the `MIF Mode` = 2'b00, the offset register specifies a an internal MIF Streamer register. This register cannot be set to a value greater than 0x2 when `control and status` register is set to `MIF mode`. You must ensure that appropriate values are set for this register, when you switch between `MIF mode` and `Direct Write mode`. Refer to **Table 17-25** for definitions of these registers. When`MIF Mode` = 2'b01, offset register specifies register in the transceiver.
7'h3C	[31:0]	RW	`data`	When the `MIF Mode` = 2'b00, the data register stores read or write data for indirect access to the location specified in the offset register. When `MIF Mode` = 2'b01, data holds an update for transceiver to be dynamically reconfigured.

Note: All undefined register bits are reserved.

Table 17-25: Streamer Module Internal MIF Register Offsets

Offset	Bits	R/W	Register Name	Description
0x0	[31:0]	RW	`MIF base address`	Specifies the MIF base address.
0x1	[2]	RW	`Clear error status`	Writing a 1 to this bit clears any error currently recorded in an indirect register. This register self clears. Any error detected in the error registers prevents MIF streaming. If an error occurs, you must clear the error register before restarting the Streamer.
	[1]	RW	`MIF address mode`	When set to 0, the streamer uses byte addresses. When set to 1, the streamer uses word addresses (16 bits).
	[0]	RW	`Start MIF stream`	Writing a 1 to this register, triggers a MIF streaming operation. This register self clears.
0x2	[4]	RO	`MIF or Channel mismatch`	When asserted, indicates the MIF type specified is incorrect. For example, the logical channel is duplex, but the MIF type specifies an RX only channel. The following 5 MIF types are defined: • Duplex • TX PLL (CMU) • RX only channel • TX only channel • TX PLL (ATX)
	[2]	RO	`PLL reconfiguration IP error`	When asserted, indicates that an error occurred changing a refclk or clock generation block setting.
	[1]	RO	`MIF opcode error`	When asserted, indicates that an undefined opcode ID was specified in the **.mif** file, or the first entry in the **.mif** file was not a start of MIF opcode.
	[0]	RO	`Invalid register access`	When asserted, indicates that the offset register address specified is out of range.

Mode 0 Streaming a MIF for Reconfiguration

In mode 0, you can stream the contents of a MIF containing the reconfiguration data to the transceiver PHY IP core instance.

You specify this mode by writing a value of 2'b00 into bits 2 and 3 of the control and status register, as indicated in *Streamer Module Registers*. Mode 0 simplifies the reconfiguration process because all reconfiguration data is stored in the MIF, which is streamed to the transceiver PHY IP in a single step.

The MIF can change PLL settings, reference clock inputs, or the TX PLL selection. After the MIF streaming update is complete, all transceiver PHY IP core settings reflect the value specified by the MIF. Refer to *Streamer-Based Reconfiguration* for an example of a MIF update.

Mode 1 Avalon-MM Direct Writes for Reconfiguration

This section describes mode 1 Avalon-MM direct writes for reconfiguration.

You specify this mode by writing a value of 2'b01 into bits 2 and 3 of the control and status register, as indicated in Streamer Module Registers. In this mode, you can write directly to transceiver PHY IP core registers to perform reconfiguration. Refer to "Direct Write Reconfiguration" for an example of an update using mode 1. In mode 1, you can selectively reconfigure portions of the transceiver PHY IP core. Unlike mode 0, mode 1 allows you to write only the data required for a reconfiguration.

MIF Generation

The MIF stores the configuration data for the transceiver PHY IP cores. The Intel Quartus Prime software automatically generates MIFs after each successful compilation.

MIFs are stored in the **reconfig_mif** folder of the project's working directory. This folder stores all MIFs associated with the compiled project for each transceiver PHY IP core instance in the design. The parameter settings of PHY IP core instance reflect the currently specified MIF. You can store the MIF in an on-chip ROM or any other type of memory. This memory must connect to the MIF reconfiguration management interface.

The following example shows file names for the **.mif** files for a design with two channels. This design example includes two transceiver PHY IP core instances running at different data rates. Both transceiver PHY IP core instances have two TX PLLs specified to support both 1 Gbps and 2.5 Gbps data rates. The Intel Quartus Prime software generates two TX PLL **.mif** files for each PLL. The difference between the **.mif** files is the PLL reference clock specified. To dynamically reconfigure the channel from the initially specified data rate to a new data rate, you can use the MIF streaming function to load the other **.mif**.

Note: When reconfiguration is limited to a few settings, you can create a partial **.mif** that only includes the settings that must be updated. Refer to Reduced MIF Creation for more information about creating a partial **.mif** file.

Example 17-4: Intel Quartus Prime Generated MIF Files

<*project_dir*>/**reconfig_mif/inst0_1g_channel.mif**

<*project_dir*>/**reconfig_mif/inst0_1g_txpll0.mif**

<*project_dir*>/**reconfig_mif/inst0_1g_txpll1.mif**

<*project_dir*>/**reconfig_mif/inst0_2p5g_channel.mif**

<*project_dir*>/**reconfig_mif/inst0_2p5g_txpll0.mif**

<*project_dir*>/**reconfig_mif/inst0_2p5g_txpll1.mif**

Creating MIFs for Designs that Include Bonded or GT Channels

You can generate MIF files for projects that include bonded or GT channels using the following procedure:

1. Create separate 1-channel designs for each frequency TX PLL frequency that your actual design requires.
2. Compile each design with the Intel Quartus Prime software.
3. Save the MIF files that Intel Quartus Prime software generates.
4. Use the MIF files that you have created from your 1-channel designs for reconfiguration in your design that includes bonded clocks.

The Intel Quartus Prime software automatically generates MIF for all designs that support POF generation with the following exceptions:

- Designs that use bonded channels so that the same TX PLL output drives several channels
- GT channels
- Non-bonded channels in a design that also includes bonded channels

MIF Format

The MIF file is organized into records where each record contains the information necessary to carry out the reconfiguration process.

There are two types of records: non-data records and data records. A MIF can contain a variable number of records, depending on the target transceiver channel. Both data records and non-data records are 16-bits long.

For both record types the high-order 5 bits represent the `length` field. A `length` field of 5'b0, indicates a non-data record which contains an opcode. A length field that is not zero indicates a data record.

For a non-data record, the opcode is represented by the lower 5-bits in the record.

Table 17-26: Opcodes for MIF Files

Opcode	Opcode Description
5'b00000	Reserved
5'b00001	Start of MIF
5'b00010	Channel format indicator specifying the MIF channel type. The following encodings are defined: • 3'b000: Duplex channel • 3'b001: TX PLL (CMU) • 3'b010: RX only channel • 3'b011: TX only channel • 3'b100: TX PLL (ATX)
5'b00011	Reference Clock switch
5'b00100	CGB switch

Opcode	Opcode Description
5'b00101-5'b11110	Reserved
5'b11111	End of MIF (EOM)

For data records, the low-order 11 bits provide a logical offset address. In this case, the length field indicates the number of data records that are written into the specified address. For example, if the length field is set to two, the next two records belong the data record and are written into the offset address.

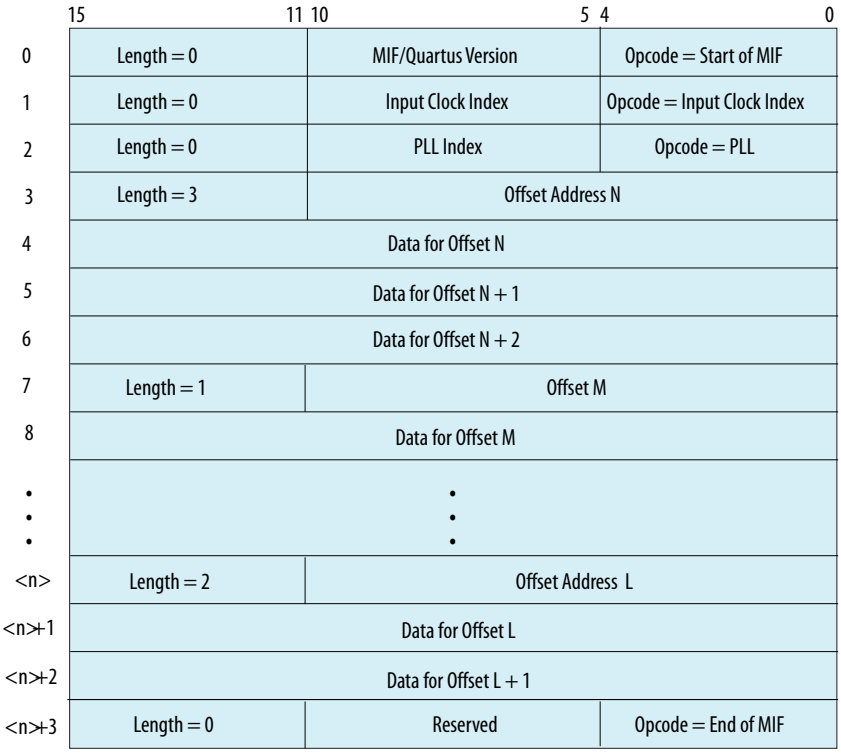
All MIF files must contain the lines in the following table.

Table 17-27: Required Lines for All MIFs

Line Number	Description	Content Includes
0	Specifies start of the reconfiguration MIF	`Start of MIF opcode`
1	Specifies the type of MIF	`Type of MIF opcode`
2	Specifies the reference clock	`RefClk switch opcode`
3	Specifies the PLL switch	`CGB PLL switch opcode`
Last	Specifies end of reconfiguration MIF	`End of MIF Opcode`

The following figure provides an example of a typical MIF format; entries 3, 7, and <n> are data records.

Figure 17-6: MIF File Format



xcvr_diffmifgen Utility

This section describes the xcvr_diffmifgen utility.

The `xcvr_diffmifgen` utility allows you to create a **.mif** file that includes the differences in settings between two configurations. For example, if you have two configurations, **inst0_1g_txpll0.mif** that sets a TX PLL0 bandwidth to 1 Gbps and **inst0_5g_txpll0.mif** that sets the TX PLL0 bandwidth to 5 Gbps, the `xcvr_diffmifgen` utility creates **to_inst0_1g_txpll0.mif** and **to_inst0_5g_txpll0.mif** that include the information necessary to change from 1 Gbps to 5 Gbps and from 5 Gbps to 1 Gbps. You can use these files to reduce reconfiguration and simulation times.

The `xcvr_diffmifgen` utility can operate on up to five MIF files. This utility only works on MIF files at the same revision level. If you try to compare MIF files that are not at the same revision level, `xcvr_diffmifgen` issues a warning.

Note: You can also use the *Reduced MIF Creation* to create reduced MIF files.

Example 17-5: xcvr_diffmifgen

```
xcvr_diffmifgen <options> <MIF file 1> <MIF file 2> <Mif file n>

Arguments:
-h: Displays help
-noopt: The output file is not optimized
```

The format of the reduced MIF file is the same as for the original MIF files as described in MIF Format. The reduced MIF file, preserves the lines shown in the following table:

Table 17-28: Required Lines for All MIFs

Line Number	Description	Content Includes
0	Specifies start of the reconfiguration MIF	`Start of MIF opcode`
1	Specifies the type of MIF	`Type of MIF opcode`
2	Specifies the reference clock	`RefClk switch opcode`
3	Specifies the PLL switch	`CGB PLL switch opcode`
Last	Specifies end of reconfiguration MIF	`End of MIF Opcode`

For each difference between the files compared, the reduced MIF file includes the following two records:

1. A record indicating the length and address of the change.
2. The changed data.

The following example shows part of two MIF files, MIF_A and MIF_B. Line 6, 16, and 20 are different.

Example 17-6: Two Partial MIF files

```
MIF_A
WIDTH=16;
DEPTH=81;
ADDRESS_RADIX=UNS
DATA_RADIX=BIN;

0: 0000000000100001; -- Start of MIF opcode - FAMILY - Stratix V
1: 0000000000000010; -- Type of MIF opcode
2: 0000000000000011; -- RefClk switch opcode
3: 0000000000000100; -- CGB PLL switch opcode
4: 0011100000000000; -- PMA - TX
5: 0000010100100000;
6: 0000100011000000;
7: 0000000100101100;
8: 0011100000000001;
9: 0000001010001100;
10: 0000000000000000;
11: 0000110000000000;
12: 0010100000001110; -- PMA - RX (PLL section)
13: 0001011010000000;
14: 0000000000000000;
15: 1000001101100000;
16: 0000000000000001;
17: 0001000100000000;
18: 0010100000010011; -- PMA - RX
19: 1110010000000011;
20: 0000111100010010;
21: 0000000001000001;
22: 0000000001010011;
23: 0011000000000000;
24: 0000100011111010; -- PCS PLD IF - TX
25: 0000000000000000;
26: 0000100011111110; -- PCS PLD IF - RX
27: 0000000000000000;
```

```
MIF_B;
WIDTH=16;
DEPTH=81;
ADDRESS_RADIX=UNS
DATA_RADIX=BIN;

0: 0000000000100001; --Start of MIF opcode- FAMILY - Stratix V
1: 0000000000000010; -- Type of MIF opcode
2: 0000000000000011; -- RefClk switch opcode
3: 0000000000000100; -- CGB PLL switch opcode
4: 0011100000000000; -- PMA - TX
5: 0000010100100000;
6: 1111111111111111;
7: 0000000100101100;
8: 0011100000000001;
9: 0000001010001100;
10: 0000000000000000;
11: 0000110000000000;
12: 0010100000001110; -- PMA - RX (PLL section)
13: 0001011010000000;
14: 0000000000000000;
15: 1000001101100000;
16: 1111111111111111;
17: 0001000100000000;
18: 0010100000010011; -- PMA - RX
19: 1110010000000011;
20: 1111111111111111;
21: 0000000001000001;
22: 0000000001010011;
23: 0011000000000000;
24: 0000100011111010; -- PCS PLD IF - TX
25: 0000000000000000;
26: 0000100011111110; -- PCS PLD IF - RX
27: 0000000000000000;
```

The following example shows and the reduced MIF file, to_MIF_A created by the xcvr_diffmifgen utility:

Example 17-7: Reduced MIF File to_MIF_A

```
WIDTH=16;
DEPTH=81;
ADDRESS_RADIX=UNS
DATA_RADIX=BIN;

0: 0000000000100001; -- Start of MIF opcode - FAMILY - Stratix V
1: 0000000000000010; -- Type of MIF opcode
2: 0000000000000011; -- RefClk switch opcode
3: 0000000000000100; -- CGB PLL switch opcode
4: 0000100000000001; -- length and address corresponds to address 1 of the original file
5: 0000100011000000; -- data
6: 0000100000010001; -- length and address corresponds to address 17 of the original file
7: 0000000000000001; -- data
8: 0000100000010100; -- length and address corresponds to address 20 of the original file
9: 0000111100010010; -- data
10:0000000000011111; -- End of MIF
```

Note: The `xcvr_diffmif` utility only works for Intel Quartus Prime post-fit simulation and hardware.

Reduced MIF Creation

The procedure described here is an alternative way to generate a reduced MIF file. You can also use the `xcvr_diffmifgen` Utility. Follow these steps to generate a reduced MIF:

1. Determine the content differences between the original MIF and the reconfigured MIF. For this example, assume there are bit differences at offset 5 and offset 20. These offsets reside in the `PMA-TX` and `PMA-RX` sections of the MIF.
2. Use a text editor to create a new reduced MIF file. In this example, we will call the reduced MIF **reduced_mif.mif**. Copy the `WIDTH`, `DEPTH`, `ADDRESS_RADIX`, `DATA_RADIX` and `CONTENT BEGIN` lines from the original MIF to **reduced_mif.mif**.
3. Copy offsets 0-3 as described **Required Lines for All MIFs** from the original MIF to **reduced_mif.mif**. The reconfiguration MIF must always include these lines.
4. Copy all offsets of the `PMA-TX` and `PMA-RX` sections from the reconfigured MIF to **reduced_mif.mif**.
5. Copy the `End of MIF` opcode offset and `END`; from the original MIF to **reduced_mif.mif**.
6. Renumber **reduced_mif.mif** sequentially and update the `DEPTH` variable with the new value. The new value equals the number offsets in **reduced_mif.mif**.

You can now use **reduced_mif.mif** to reconfigure the transceiver.

You can create a reduced MIF from the following two MIFs:

- Original MIF—contains the transceiver settings that were specified during the initial compilation
- Reconfigured MIF—contains the new transceiver settings. You generate the reconfigured MIF by modifying the original transceiver settings. For example, if the original compilation specifies a clock divider value of 1 and the reconfigured compilation specifies a clock divider value of 2, the MIF files reflect that change. The reduced MIF contains only the changed content. In this example, the difference between the two MIFs would be the clock divider value.

Changing Transceiver Settings Using Register-Based Reconfiguration

This section describes changing the transceiver settings.

In register-based mode, you use a sequence of Avalon-MM writes and reads to update individual transceiver settings. The following section describes how to perform a register-based reconfiguration read and write.

Register-Based Write

Complete the following steps, using a state machine as an example, to perform a register-based write:

1. Read the `control and status` register `busy` bit (bit 8) until it is clear.
2. Write the logical channel number of the channel to be updated to the `logical channel number` register.
3. Write the <*feature*> offset address.
4. Write the appropriate data value to the data register.
5. Write the `control and status` register `write` bit to 1'b1.
6. Read the `control and status` register busy bit. Continue to read the `busy` bit while its value is one.
7. When `busy` = 0, the Transceiver Reconfiguration Controller has updated the logical channel specified in Step 2 with the data specified in Step 3.

Example 17-8: Register-Based Write of Logical Channel 0 V_{OD} Setting

System Console is used for the following settings:

```
#Setting logical channel 0
write_32 0x8 0x0

#Setting offset to VOD
write_32 0xB 0x0

#Setting data register to 40
write_32 0xC 0x28

#Writing all data
write_32 0xA 0x1
Ā
```

Register-Based Read

Complete the following steps, using a state machine as an example, for a read:

1. Read the `control and status` register `busy` bit (bit 8) until it is clear.
2. Write the logical channel number of the channel to be read to the `logical channel number` register.
3. Write the <*feature*> offset address.
4. Write the `control and status` register `read` bit to 1'b1.
5. Read the `control and status` register `busy` bit. Continue to read the busy until the value is zero.
6. Read the `data` register to get the data.

Example 17-9: Register-Based Read of Logical Channel 2 Pre-Emphasis Pretap Setting

System Console is used for the following settings:

```
#Setting logical channel 2
write_32 0x8 0x2

#Setting offset to pre-emphasis pretap
write_32 0xB 0x1

#Writing the logical channel and offset for pre-emphasis pretap
write_32 0xA 0x2

#Reading data register for the pre-emphasis pretap value
read_32 0xC
```

Changing Transceiver Settings Using Streamer-Based Reconfiguration

The Streamer's registers allow you to change to the PCS datapath settings, clock settings, PRBS settings, and PLL parameters by reading the new settings from an on- or off-chip ROM.

Streamer Module Registers lists the Streamer's memory-mapped registers that you can access using Avalon-MM read and write commands on reconfiguration management interface.

The following sections show how to change transceiver settings using Streamer modes 0 and 1.

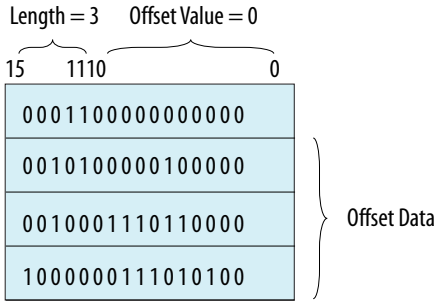
Direct Write Reconfiguration

Follow these steps to reconfigure a transceiver setting using a series of Avalon-MM direct writes.

1. Write the logical channel number to the Streamer `logical` channel register.
2. Write Direct Mode, 2'b01, to the Streamer `control and status` register `mode` bits.
3. Write the offset address to the Streamer `offset` register.
4. Write the offset data to the Streamer `data` register.
5. Write the Streamer `control and status` register `write` bit to 1'b1 to initiate a write of all the data set in the previous steps.
6. Repeat steps 3 through 5 if the offset data `length` is greater than 1. Increment the offset value by 1 for each additional data record.
7. Read the `control and status` register `busy` bit. When the `busy` bit is deasserted, the operation has completed.

In Steps 3 and 4, you must specify an offset value and offset data. You can determine the values of the offset address and offset data by examining the data records specified in either the channel or PLL MIFs.

Figure 17-7: Sample MIF



For the sample data record, the length field specifies three data records. The offset value is 0, as indicated by bits 10–0. The offset data are the three subsequent entries. The following example performs a direct write in Streamer mode 1. This example writes the sample MIF into the Streamer module which writes this data to logical channel 0.

Example 17-10: Streamer Mode 1 Reconfiguration

```
#Setting logical channel 0
write_32 0x38 0x0
#Read the busy bit to determine when the operation completes
read_32 0x3a
#Setting Streamer to mode to 1
write_32 0x3A 4'b0100
#Read the busy bit to determine when the operation completes
read_32 0x3a
#Setting Streamer offset register to the offset address
#In the example record, the first offset address is 0x0
write_32 0x3B 0x0
#Read the busy bit to determine when the operation completes
read_32 0x3a
#Setting data register with the first data record
write_32 0x3C 16'b0010100000100000
#Read the busy bit to determine when the operation completes
read_32 0x3a
#Writing first data to the Streamer
write_32 0x3A 0x5
#Read the busy bit to determine when the operation completes
read_32 0x3a
#Incrementing Streamer offset register offset address
write_32 0x3B 0x1
#Read the busy bit to determine when the operation completes
read_32 0x3a
#Setting data register with the second data record
write_32 0x3C 16'b0010001110110000
#Read the busy bit to determine when the operation completes
read_32 0x3a
#Writing second data to the Streamer
write_32 0x3A 0x5
#Read the busy bit to determine when the operation completes
read_32 0x3a
#Incrementing Streamer offset register offset address
write_32 0x3B 0x2
#Read the busy bit to determine when the operation completes
read_32 0x3a
#Setting data register with the third data record
write_32 0x3C 16'b1000000111010100
#Read the busy bit to determine when the operation completes
```

```
read_32 0x3a
#Writing third data record to the Streamer
write_32 0x3A 0x5
#Read the busy bit to determine when the operation completes
read_32 0x3a
#Read the busy bit to determine when the operation completes
read_32 0x3a
```

Streamer-Based Reconfiguration

Follow these steps to reconfigure a transceiver setting by streaming the contents of a MIF file through the Streamer Module.

1. Write the logical channel number to the Streamer `logical channel` register.
2. Write MIF mode, 2'b00, to the Streamer `control and status` register mode bits.
3. Write the MIF base address, 0x0, to the Streamer `offset` register.
4. Write the base address of the MIF file to the Streamer `data` register.
5. Write the Streamer `control and status` register `write` bit to 1'b1 to initiate a write of all the data set in the previous steps.
6. Write to the Streamer `offset` register with the value to start a MIF stream, 0x1.
7. Write the Streamer internal `data` register with the value 0x3 to setup the streaming of the MIF.
8. Write to the Streamer `control and status` register to 1'b1, to initiate the streaming operation.
9. Read the `control and status` register `busy` bit. When the `busy` bit is deasserted, the MIF streaming operation has completed.

The following example illustrates the reconfiguration of logical channel 0 using a MIF with a base address of 0x100.

Example 17-11: Reconfiguration of Logical Channel 0 Using a MIF

```
#Setting logical channel 0
write_32 0x38 0x0
#Setting Streamer mode to 0
write_32 0x3A 0x0
#Setting Streamer offset register to the MIF base address (0x0)
write_32 0x3B 0x0
#Setting data register with the MIF base address
write_32 0x3C 0x100
#Writing all data to the Streamer
write_32 0x3A 0x1
#Setting Streamer Module offset for Start MIF stream
write_32 0x3B 0x1
#Setting data register with 0x3 to setup for streaming
write_32 0x3C 0x3
#Writing all data to the Streamer to start streaming the MIF
write_32 0x3A 0x1
#Read the busy bit to determine when the write has completed
read_32 0x3A
```

Pattern Generators for the Stratix V and Arria V GZ Native PHYs

Both the Standard and 10G PCS contain dedicated pattern generators that you can use for verification or diagnostics. The pattern generator blocks support the following patterns:

- Pseudo-random binary sequence (PRBS)
- Pseudo-random pattern
- Square wave

You enable and disable the pattern generator using the Streamer module.

Enabling the Standard PCS PRBS Verifier Using Streamer-Based Reconfiguration

Complete the following reads and writes to the Streamer module to enable the PRBS verifier in the Standard PCS:

1. Read the Streamer Module `control and status` register `busy` bit (7'h3A, bit[8]) until it is clear.
2. Write the Streamer Module logical channel number to the Streamer `logical channel number` register at address 0x38.
3. Set the Streamer Module `control and status` register `Mode` bits (7'h3A, bits[3:2]) to 1.
4. Determine the PRBS pattern from the table above and note the corresponding word aligner size and word aligner pattern. The word aligner size and word aligner pattern are used in the next two steps. For example, using a 16-bit PCS/PMA width and a PRBS-23 pattern the corresponding word aligner size is 3'b101 and word aligner pattern is 0x00007FFFFF.
5. Perform a read-modify-write to the `Word Aligner Size` field (offset 0xA1, bits[10:8]) to change the word aligner size.
6. Because the word aligner pattern is specified in three separate register fields, to change the word aligner pattern, you must perform read-modify-writes to the following three register fields:
   a. `Word Aligner Pattern`, bits [39:32] (offset 0xA1, bits [7:0])
   b. `Word Aligner Pattern`, bits [31:16] (offset 0xA2, bits [15:0])
   c. `Word Aligner Pattern`, bits [15:0] (offset 0xA2, bits [15:0])
7. To enable the PRBS verifier, perform the following three read-modify-write operations to set the values of these bits to 0:
   a. `Sync badcg`, (offset 0xA1, bits[15:14])
   b. `Enable Comma Detect`, (offset 0xA1, bit[13])
   c. `Enable Polarity`, (offset, 0xA1, bit[11])
8. Now, you must set the proper value for the `Sync State Machine Disable` bit.
   - If your PCS/PMA interface width is 8 or 10 bits, perform a read-modify-write with a value of 1'b1 to `Sync State Machine Disable` (offset 0xA4, bit[15]).
   - If your PCS/PMA interface width is 16 or 20 bits, perform a read-modify-write with a value of 1'b0 to `Sync State Machine Disable` (offset 0xA4, bit[15]).
9. To complete the necessary programming,
   a. Perform read-modify-writes to set the following bits to 0:

- `Auto Byte Align Disable` (offset 0xA6, bit[5])
- `DW Sync State Machine Enable` (offset 0xB8, bit[13])
- `Deterministic Latency State Machine Enable` (offset 0xB9, bit[11])
- `Clock Power Down RX` (offset 0xBA, bit[11])

b. Perform read-modify-writes to set the following bits to 1:

- `PRBS RX Enable` (offset 0xA0, bit[5])

10. Assert the channel reset.

Enabling the Standard PCS PRBS Generator Using Streamer-Based Reconfiguration

Complete the following reads and writes to the Streamer module to enable the PRBS generator in the Standard PCS:

1. Read the Streamer Module `control and status` register `busy` bit (7'h3A, bit[8]) until it is clear.
2. Write the Streamer Module logical channel number to the Streamer `logical channel number` register at address 0x38.
3. Set the Streamer Module `control and status` register `Mode` bits (7'h3A, bits[3:2]) to 1.
4. Determine the PRBS pattern from *Word Aligner Size and Word Aligner Pattern* table in **Standard PCS Pattern Generators** on page 13-25 and note the corresponding `PRBS Pattern Select` encoding. For example, using a 16-bit PCS-PMA width and a PRBS-23 pattern, the corresponding PRBS select value is 3'b001.
5. Determine the PRBS pattern from the and note the cooresponding PRBS Pattern Select encoding. For example, using a 16-bit PCS-PMA width and a PRBS-23 pattern, the corresponding PRBS select value is 3'b001.
6. To complete the necessary programming, perform read-modify-writes to specify the `PRBS TX Enable` and at the following addresses:
   a. `PRBS TX Enable`, (0x97, bit[9])
   b. `PRBS Pattern Select`, (0x97, bit[8:6])
7. Assert the channel reset to begin testing on the new PRBS pattern.

Enabling the 10G PCS PRBS Generator or Verifier Using Streamer-Based Reconfiguration

Complete the following reads and writes to the Streamer module of the Transceiver Reconfiguration Controller to enable one of the three pattern generators in the 10G PCS.

1. Read the Streamer Module `control and status` register `busy` bit (7'h3A, bit[8]) until it is clear.
2. Write the Streamer Module logical channel number to the Streamer `logical channel number` register at address 0x38.
3. Set the Streamer Module `control and status` register `Mode` bits (7'h3A, bits[3:2]) to Mode 1.
4. Write the pattern type (0x135 for the PRBS pattern generator or 0x15E for the PRBS pattern verifier) to the Streamer Module `streamer offset` register at address 0x3B.
5. Write the Streamer Module `control and status` register (0x3A) with a value of 0x6 to initiate a read.
6. Read the Streamer Module `data` register at address 0x3C.
7. Perform a read-modify-write to the generator or verifier bits using the value read in step 6 to retain values for the bits that should not change.

   For example, to select the PRBS31 generator and to enable the PRBS generator, perform a read-modify-write using the value 16'b0000-0-0-0-11-00 read from address 0x3C. In this 16-bit value, the hyphens represent bits that should not be modified.

8. Write the new value from step 7 to the Streamer `data` register at address 0x3C
9. Write the Streamer `control and status` register (7'h3A) with a value of 0x5.
10. Repeat steps 3-9 to the `TX PRBS Clock Enable` (0x137) for `RX PRBS Clock Enable` (0x164).
11. Assert the channel resets.

Note: You can only enable one of the three pattern generators at a time.

Example 17-12: Enable the PRBS 31 Generator

```
//PRBS31 Pattern Generator selection and setup
read_32   0x3A  //Read the control and status register busy
    //bit[8] until it is clear
write_32 0x38 0x0   //write the logical channel to 0x38
write_32 0x3A 0x4   //set the MIF mode 1 to address 0x3A
write_32 0x3B 0x135 //write the pattern type offset
write_32 0x3A 0x6   //write the control and status register
    //with a value of 0x6 to address 0x3A to initiate a read
    //read_32 0x3C Read the value at address 0x3C
RMW {16'b0000-0-0-0-11-00,  {Read_32 0x3C}} //Perform a
    // read-modify-write with the generator or bits and the
    // value read from above
write_32 0x3C 0x<result from above > //Write the new value from
    //above to the dataregister at address 0x3C
write_32 0x3A0x5 //Write the control and status register
    //with a value of 0x5 to address 0x3A

//Generator clock setup
read_32   0x3A  //Read the control and status register busy
    //bit[8] until it is clear
write_32 0x38 0x0    //write logical channel to 0x38
write_32 0x3A 0x4    //set the MIF mode 1 to address 0x3A
write_32 0x3B 0x137  //write the clock
write_32 0x3A 0x6    //write the control and status register
    //with a value of 0x6 to address 0x3A to initiate a read
read_32 0x3C         //Read the value at address 0x3C
RMW {3'b10-,{read_32 0x3C}} //Perform a read-modify-write
    //with the generator or bits and the value read from above
write_32 0x3C 0x<result from above > //Write the new value
    //from above to the data register at address 0x3C
write_32 0x3A 0x5 //Write the control and status register
    //with a value of 0x5 to address 0x3A

//Assert the channel resets
```

Enable the Square Wave Generator

```
//Enable the square wave generator with 5 consecutive 1s and 0s
//Generator selection and setup
read_32  0x3A      //Read the control and status register
    //busy bit[8] until it is clear
write_32 0x38    0x0     //write logical channel to 0x38
write_32 0x3A    0x4     //set the MIF mode 1 to address 0x3A
write_32 0x3B    0x135     //write the pattern type offset
write_32 0x3A    0x6    //write the control and status register
   //with a value of 0x6 to address 0x3A to initiate a read
read_32 0x3C         //Read the value at address 0x3C
RMW {16'b0101-0-0-0-00-10,  {read_32 0x3C}}  //Perform a
   //read-modify-write with the generator or bits
   //and the value read from above
write_32 0x3C    0x<result from above > //Write the new value from
   //above to the data register at address 0x3C
write_32 0x3A    0x5     //Write the control and status register
```

```
    // with a value of 0x5 to address 0x3A

//Generator clock setup
read_32   0x3A      //Read the control and status register
    //busy bit[8] until it is clear
write_32 0x38    0x0     //write logical channel to 0x38
write_32 0x3A    0x4     //set the MIF mode 1  to address 0x3A
write_32 0x3B    0x137     //write the pattern type offset
write_32 0x3A    0x6    //write the control and status register with
   // a value of 0x6 to address 0x3A to initiate a read
read_32 0x3C         //Read the value at address 0x3C
RMW {3'b01-,  {read_32 0x3C}} //Perform a read-modified-write
   //with the generator or bits and the value read from above
write_32 0x3C    0x<result from above > //Write the new value
   //from above to the data register at address 0x3C
write_32 0x3A    0x5     //Write the control and status register
   //with a value of 0x5 to address 0x3A

//Assert the channel resets
```

Enable the Pseudo-Random Generator

```
//Enable the pseudo-random pattern generator
//Seed A is set to 0x5. Use 2 local faults
read_32 0x3A //Read the control and status register busy
    // bit[8] until it is clear
write_32 0x38 0x0   //write logical channel to 0x38
write_32 0x3A 0x4   //set the MIF mode 1 to address 0x3A
write_32 0x3B 0x12D //write the pattern type offset
write_32 0x3C 0x5   //Write the new value from above
    //to the data register at address 0x3C
write_32 0x3A 0x5   //Write the control and status register
    //with a value of 0x5 to address 0x3A

//Generator selection and setup
read_32 0x3A        //Read the control and status register
    //busy bit[8] until it is clear
write_32 0x38 0x0   //write logical channel to 0x38
write_32 0x3A 0x4   //set the MIF mode 1 to address 0x3A
write_32 0x3B 0x135 //write the pattern type offset
write_32 0x3A 0x6//write the control and status register
   //with a value of 0x6 to address 0x3A to initiate a read
read_32 0x3C //Read the value at address 0x3C
RMW {3'b1-01, {read_32 0x3C}} //Perform a read-modify-write
    //with the generator or bits and the value read from above
write_32 0x3C 0x<result from above > //Write the new value
    //from above to the data register at address 0x3C
write_32 0x3A 0x5 //Write the control and status register with
    //a value of 0x5 to address 0x3A

//Assert the channel resets
```

Disabling the Standard PCS PRBS Generator and Verifier Using Streamer-Based Reconfiguration

To disable the PRBS generator or verifier, restore the original values of all registers written to enable the PRBS generator or verifier. Restoring the original values requires you to save them while performing the read-modify-write operations.

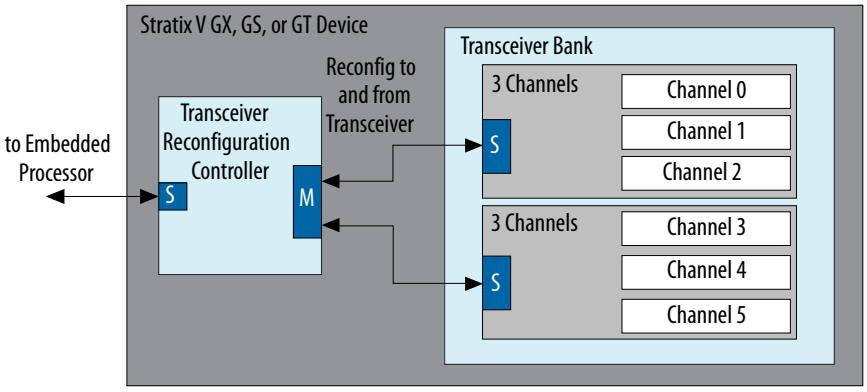
Understanding Logical Channel Numbering

This discussion of channel numbering, uses the following definitions:

- Reconfiguration interface—A bundle of signals that connect the Transceiver Reconfiguration Controller to a transceiver PHY data channel or TX PLL.
- Logical channels—An abstract representation of a channel or TX PLL that does not include physical location information.
- Bonded channel—A channel that shares a clock source with at least one other channel.
- Physical channel—The physical channel associated with a logical channel.

The following figure illustrates the connections between the Transceiver Reconfiguration Controller and a transceiver bank after running the Intel Quartus Prime Fitter.

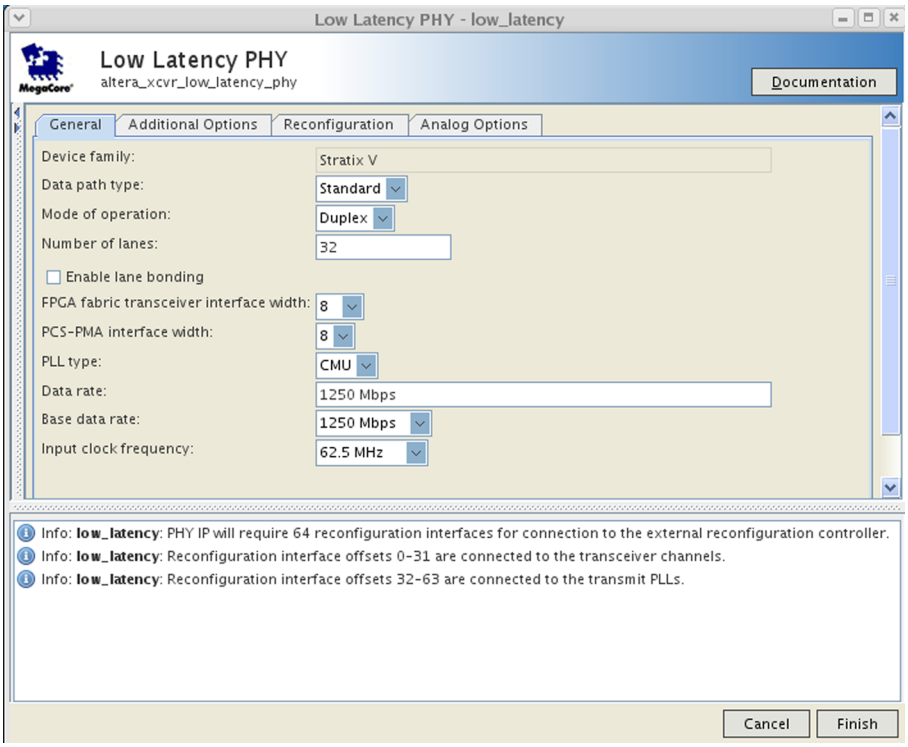
Figure 17-8: Post-Fit Connectivity



The transceiver PHY IP cores create a separate reconfiguration interface for each channel and each TX PLL. Each transceiver PHY IP core reports the number of reconfiguration interfaces it requires in the message pane of its GUI. You must take note of this number so that you can enter it as a parameter in the Transceiver Reconfiguration Controller.

The following figure shows the Low Latency PHY IP ore GUI specifying 32 channels. The message pane indicates that reconfiguration interfaces 0–31 are for the transceiver channels and reconfiguration interfaces 32–63 are for the TX PLLs.

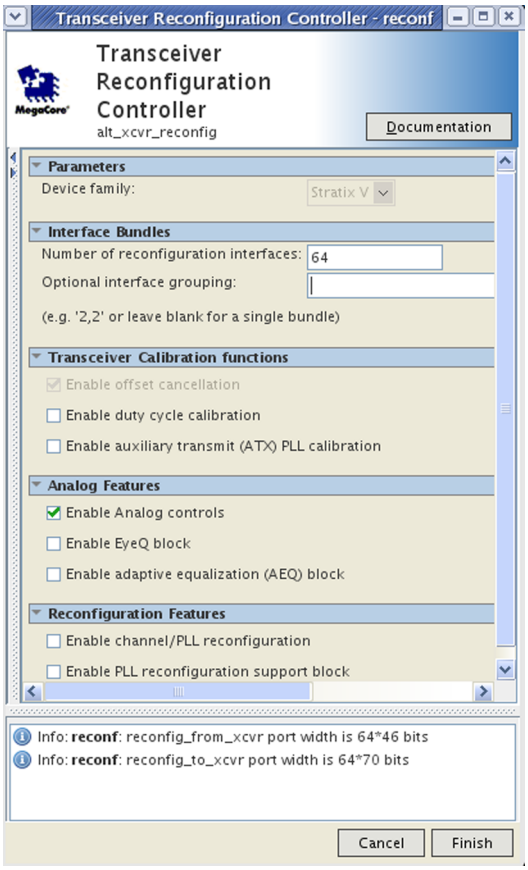
Figure 17-9: Low Latency Transceiver PHY Example



Note: After Intel Quartus Prime compilation, many of the interfaces are merged.

The following figure illustrates the GUI for the Transceiver Reconfiguration Controller. To connect the Low Latency PHY IP Core instance to the Transceiver Reconfiguration Controller, you would enter 64 for **Number of reconfiguration interfaces**. You would not need to enter any values for the **Optional interface grouping** parameter because all of the interfaces belong to the same transceiver PHY IP core instance.

Figure 17-10: Transceiver Reconfiguration Controller Interface Bundles



The following figure shows a design with two transceiver PHY IP core instances, each with four channels. For this design you would enter 16 for the **Number of reconfiguration interfaces** and 8, 8 for the **Optional interface grouping** parameter.

Depending upon the transceiver PHY IP core and the parameters specified, the number of reconfiguration interfaces varies. For a single-channel, RX-only transceiver instance, there is a single reconfiguration interface. One reconfiguration interface is created for a single-channel Low Latency PHY setup as a RX only channel. Two reconfiguration interfaces are created for a single-channel Custom PHY setup as a duplex channel. The reconfiguration interfaces do not appear as separate buses, but as a single bus of concatenated reconfiguration interfaces, that grows linearly with the number of reconfiguration interfaces.

Although you must create a separate logical reconfiguration interface for each PHY IP core instance, when the Intel Quartus Prime software compiles your design, it reduces original number of logical interfaces by merging them. Allowing the Intel Quartus Prime software to merge reconfiguration interfaces gives the Fitter more flexibility in placing transceiver channels. However, the logical channel number remains the same.

Note: You cannot use SignalTap™ to observe the reconfiguration interfaces.

You do not have to assign numbers to the reconfiguration interfaces. The logical interface numbering is determined by the order of the interfaces in the connection between the transceiver PHY IP and the Transceiver Reconfiguration Controller.

Two PHY IP Core Instances Each with Four Bonded Channels

This section describes logical channel numbering for two transceiver PHY instances, each with four bonded channels, connected to a Transceiver Reconfiguration Controller.

When two transceiver PHY instances, each with four bonded channels, are connected to a Transceiver Reconfiguration Controller, the reconfiguration buses of the two instances are concatenated. The following figure and table show the order and numbering of reconfiguration interfaces. The Intel Quartus Prime software assigns the data channels logical channel numbers 0 to 3 for each transceiver PHY instance. The Intel Quartus Prime software assigns the TX PLLs logical channel numbers 4 to 7 for each transceiver PHY instance. During Intel Quartus Prime place and route, the Fitter maps the four logical TX PLLs in each transceiver PHY instance to a single physical TX PLL.

Figure 17-11: Interface Ordering with Multiple Transceiver PHY Instances

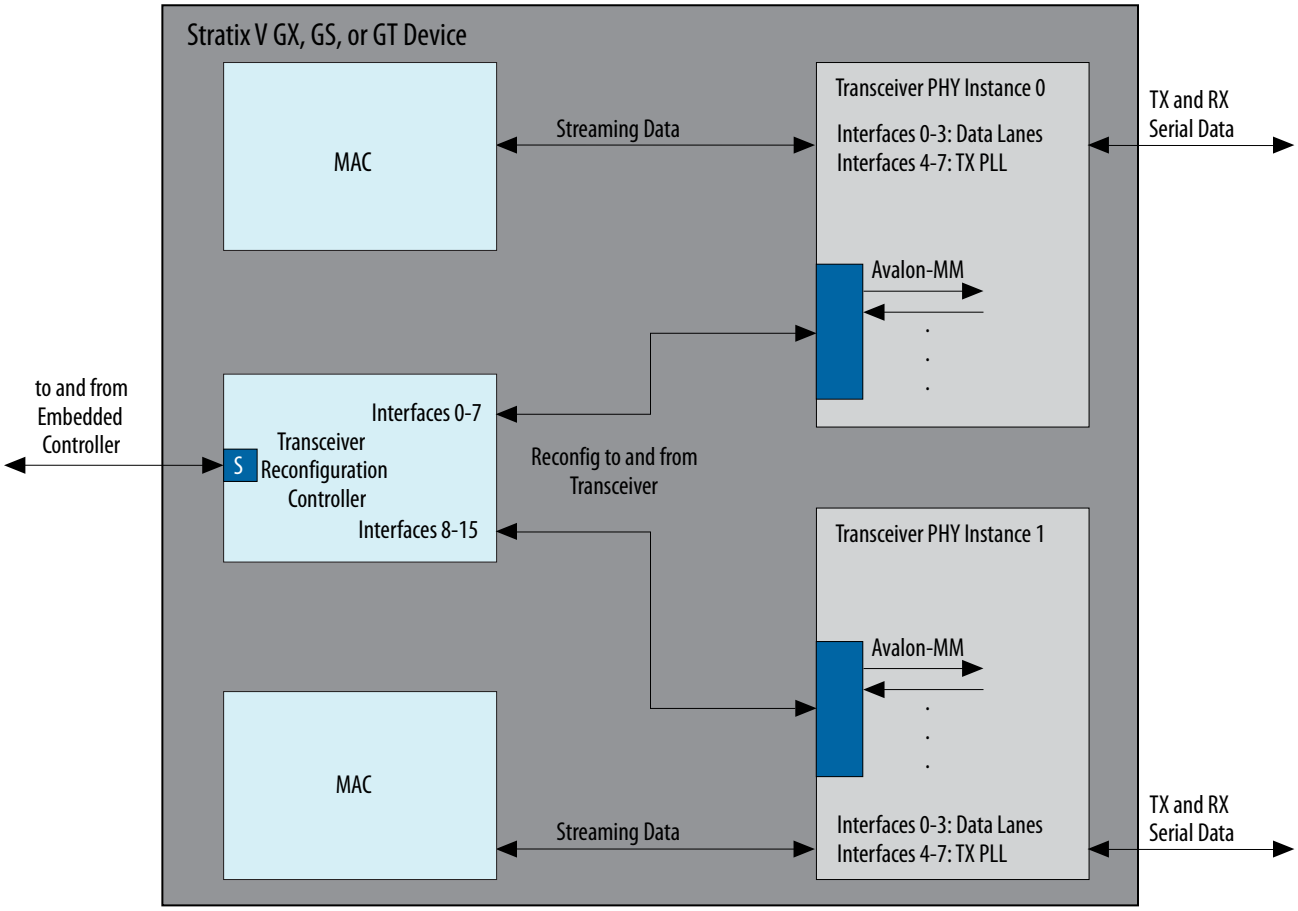


Table 17-29: Channel Ordering for Concatenated Transceiver Instances

Logical Interface Number	PHY Instance, Interface, or PLL
0-3	Instance 0, interfaces 0-3.
4-7	Instance 0, TX PLL. The Fitter assigns all 4 logical TX PLLs to a single physical PLL.
8-11	Instance 1, interfaces 0-3.

Logical Interface Number	PHY Instance, Interface, or PLL
12-15	Instance 1, TX PLL. The Fitter assigns all 4 logical TX PLLs to a single physical PLL.

One PHY IP Core Instance with Eight Bonded Channels

This section describes logical channel numbering for one transceiver instance with eight bonded channels.

This example requires the Intel Quartus Prime Fitter to place channels in two, contiguous transceiver banks. To preserve flexibility for the Fitter, each channel and TX PLL is numbered separately. During place and route, the Fitter maps the eight logical TX PLLs to a single physical TX PLL.

The following table illustrates the logical channel numbering. In this table, logical address 0 accesses data channel 0 and logical address 8 accesses the TX PLL for data channel 0; logical address 1 accesses data channel 1 and logical address 9 accesses the TX PLL for data channel 1, and so on. In simulation, to reconfigure the TX PLL for channel 0, specify logical address 8 in the Streamer module's `logical channel number`. The Streamer module maps the logical channel to the physical channel which would be the same value for all eight channels.

Table 17-30: Initial Number of Eight Bonded Channels

Channel	Logical Channel Number
Channel 0	0
Channel 1	1
Channel 2	2
Channel 3	3
Channel 4	4
Channel 5	5
Channel 6	6
Channel 7	7
CMU 0	8
CMU 1	9
CMU 2	10
CMU 3	11
CMU 4	12
CMU 5	13
CMU 6	14
CMU 7	15

Note: Because all of the channels in a transceiver bank share a PLL, this original numbering allows the Fitter to select the optimal CMU PLL from a placement perspective by considering all of the TX PLLs in the bank.

The following table shows the channel numbers for post-Fitter and hardware simulations. At this point, you should have assigned channels to pins of the device.

Table 17-31: Post-Fit Logical Channel Numbers for Eight Bonded Channels

Channel	Logical Channel Number
Channel 0	0
Channel 1	1
Channel 2	2
Channel 3	3
CMU (0-4)	8-12
Channel 4	4
Channel 5	5
CMU (5-7)	13-15
Channel 6	6
Channel 7	7

Two PHY IP Core Instances Each with Non-Bonded Channels

This section describes two instances with non-bonded channels.

For each transceiver PHY IP core instance, the Intel Quartus Prime software assigns the data channels sequentially beginning at logical address 0 and assigns the TX PLLs the subsequent logical addresses.

The following table illustrates the logical channel numbering for two transceiver PHY IP cores, one with 4 channels and one with 2 channels.

Table 17-32: Initial Number of Eight Bonded Channels

Instance	Channel	Logical Channel Number
Instance 0	Channel 0	0
	Channel 1	1
	Channel 2	2
	Channel 3	3
	CMU 0	4
	CMU 1	5
	CMU 2	6
	CMU 3	7
Instance 1	Channel 0	8
	Channel 1	9
	CMU 0	10
	CMU 1	11

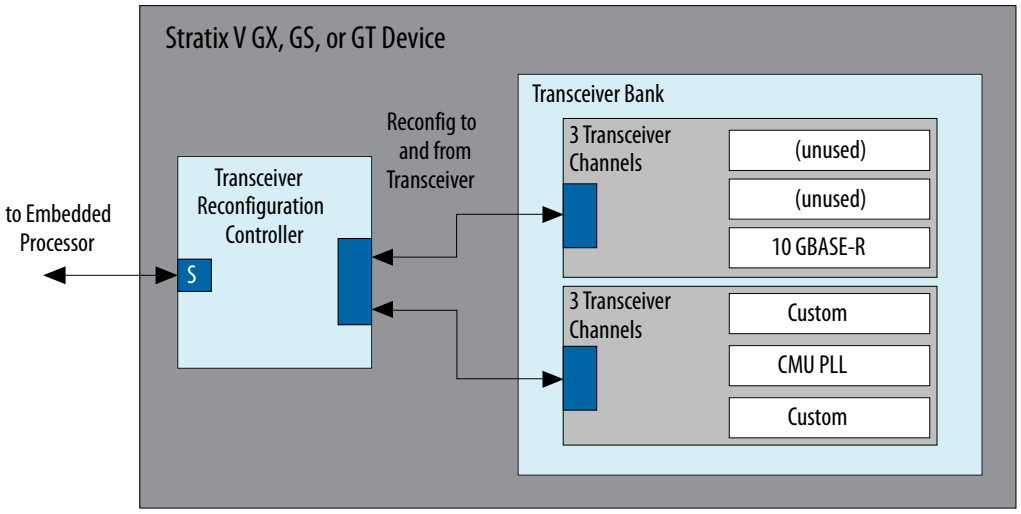
Transceiver Reconfiguration Controller to PHY IP Connectivity

This section describes connecting a Transceiver Reconfiguration Controller to the transceiver channels and PLLs in your design.

You can connect a single Transceiver Reconfiguration Controller to all of the transceiver channels and PLLs in your design. You can also use multiple Transceiver Reconfiguration Controllers to facilitate placement and routing of the FPGA. However, the three, upper or lower contiguous channels in a transceiver bank must be connected to the same reconfiguration controller.

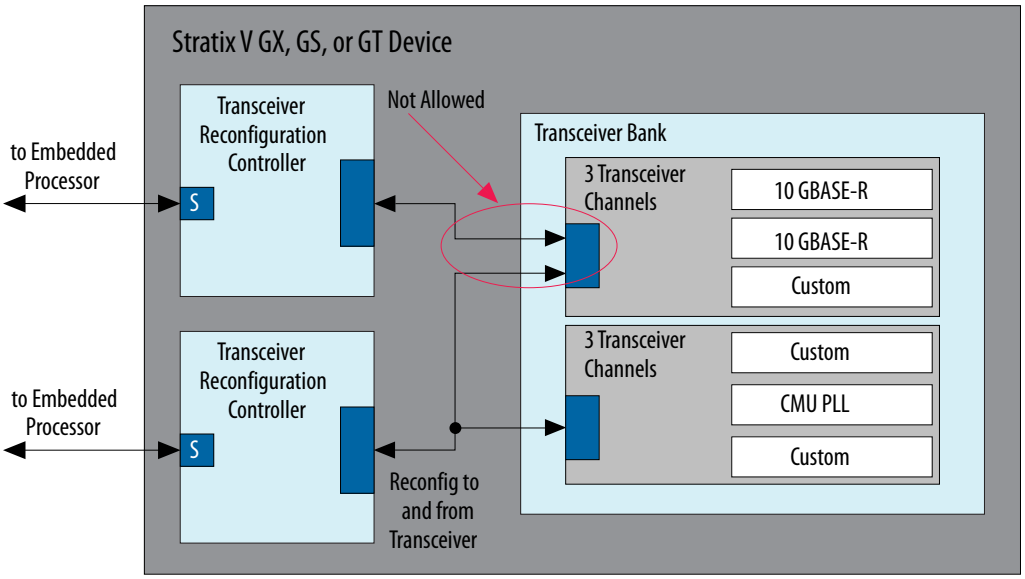
The following figure illustrates connections between the Transceiver Reconfiguration Controller and transceiver channels after Intel Quartus Prime compilation.

Figure 17-12: Correct Connections



The following figure illustrates incorrect connections between two Transceiver Reconfiguration Controllers and six transceiver channels. Two Transceiver Reconfiguration Controllers cannot access a single reconfiguration interface because there is no arbitration logic to prevent concurrent access. The configuration shown results in a Intel Quartus Prime compilation error.

Figure 17-13: Incorrect Connections



Merging TX PLLs In Multiple Transceiver PHY Instances

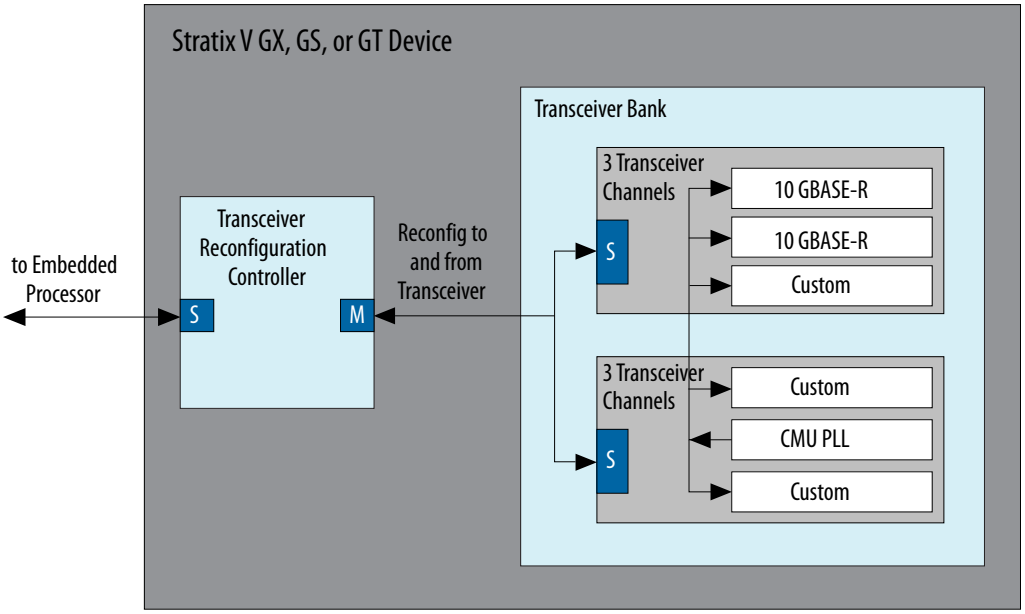
This section describes merging instances of the transceiver PHY.

The Intel Quartus Prime Fitter can merge the TX PLLs for multiple transceiver PHY IP cores under the following conditions:

- The PLLs connect to the same reset pin.
- The PLLs connect to the same reference clock.
- The PLLs connect to the same Transceiver Reconfiguration Controller.

The following figure illustrates a design where the CMU PLL in channel 1 provides the clock to three Custom PHY channels and two 10GBASE-R PHY channels.

Figure 17-14: PLL Shared by Multiple Transceiver PHY IP Cores in a Single Transceiver Bank



Sharing Reconfiguration Interface for Multi-Channel Transceiver Designs

Although you must initially create a separate reconfiguration interface for each channel and TX PLL in your design, when the Intel Quartus Prime software compiles your design, it reduces the number of reconfiguration interfaces by merging reconfiguration interfaces. The synthesized design typically includes a reconfiguration interface for at least three channels because three channels share an Avalon-MM slave interface which connects to the Transceiver Reconfiguration Controller IP Core. Conversely, you cannot connect the three channels that share an Avalon-MM interface to different Transceiver Reconfiguration Controller IP Cores. Doing so causes a Fitter error.

Loopback Modes

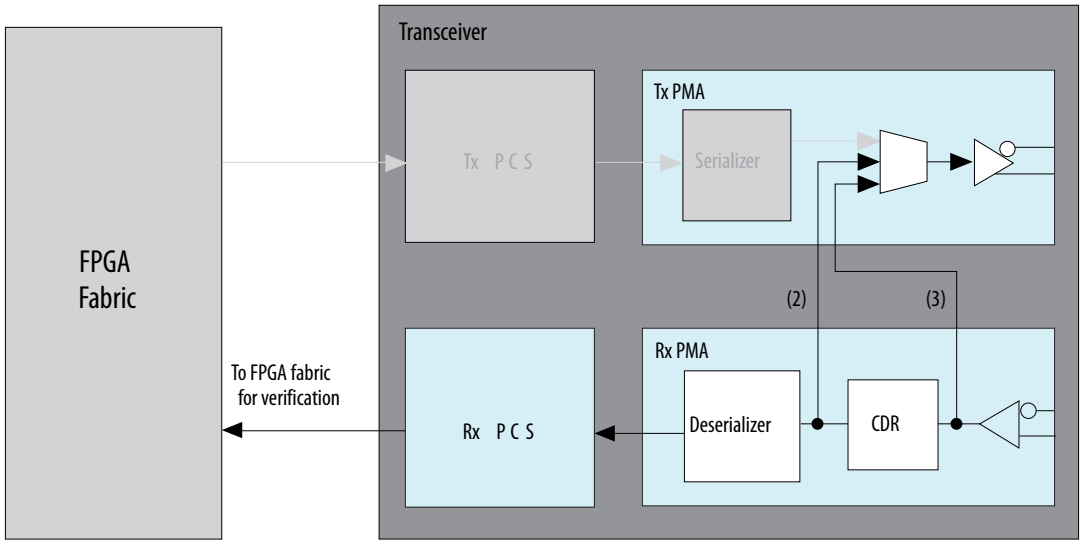
The PMA analog registers allow you to enable pre- and post-CDR serial loopback modes.

You can enable the pre- and post-CDR reverse serial loopback modes by writing the appropriate bits of the Transceiver Reconfiguration Controller `pma_offset` register described in *PMA Analog Registers*. In pre-CDR mode, data received through the RX input buffer is looped back to the TX output buffer. In post-

CDR mode, received data passes through the RX CDR and then loops back to the TX output buffer. The RX data is also available to the FPGA fabric. In the TX channel, only the TX buffer is active.

Figure 17-15: Pre- and Post-CDR Reverse Serial Loopback Paths

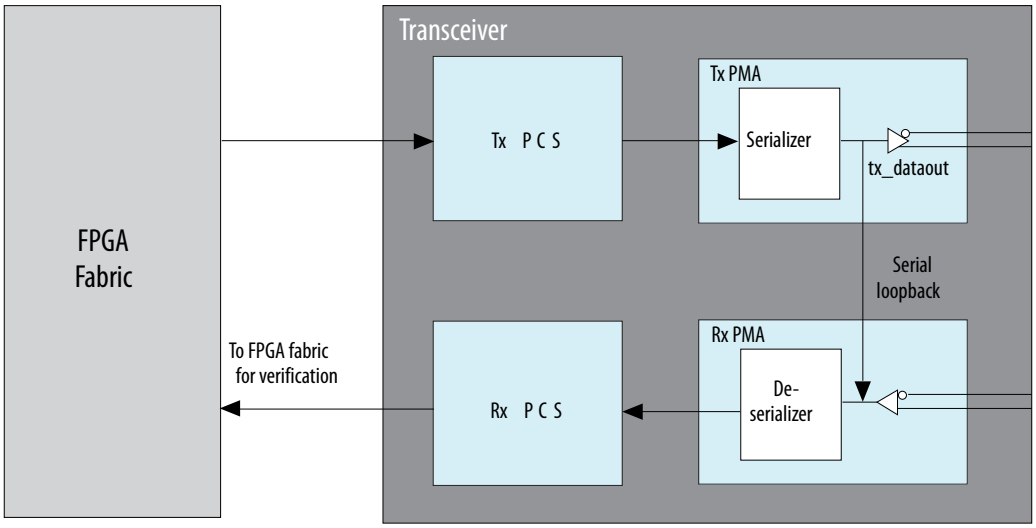
In this figure, grayed-out blocks are not active in these modes. The number (2) shows the post-CDR loopback path and the number (3) shows pre-CDR reverse serial loopback path.



In addition to the pre-CDR and post-CDR loopback modes available in the Transceiver Reconfiguration Controller register map, all the of PHYs, with the exception of PCI Express, support serial loopback mode. You enable this mode by writing the `phy_serial_loopback` register (0x061) using the Avalon-MM PHY management interface except for the Native PHY IP. In Native PHY IP, you can enable the serial loopback mode by driving `rx_seriallpbken` input port to 1'b1. Also, PCI Express supports reverse parallel loopback mode as required by the *PCI Express Base Specification*.

The following figure shows the datapath for serial loopback. The data from the FPGA fabric passes through the TX channel and is looped back to the RX channel, bypassing the RX buffer. The received data is available to the FPGA fabric for verification. Using the serial loopback option, you can check the operation of all enabled PCS and PMA functional blocks in the TX and RX channels. When serial loopback is enabled, the TX channel sends the data to both the tx_serial_data output port and the RX channel.

Figure 17-16: Serial Loopback



Related Information

PCI Express Base Specification

Transceiver PHY Reset Controller IP Core 18

2020.06.02

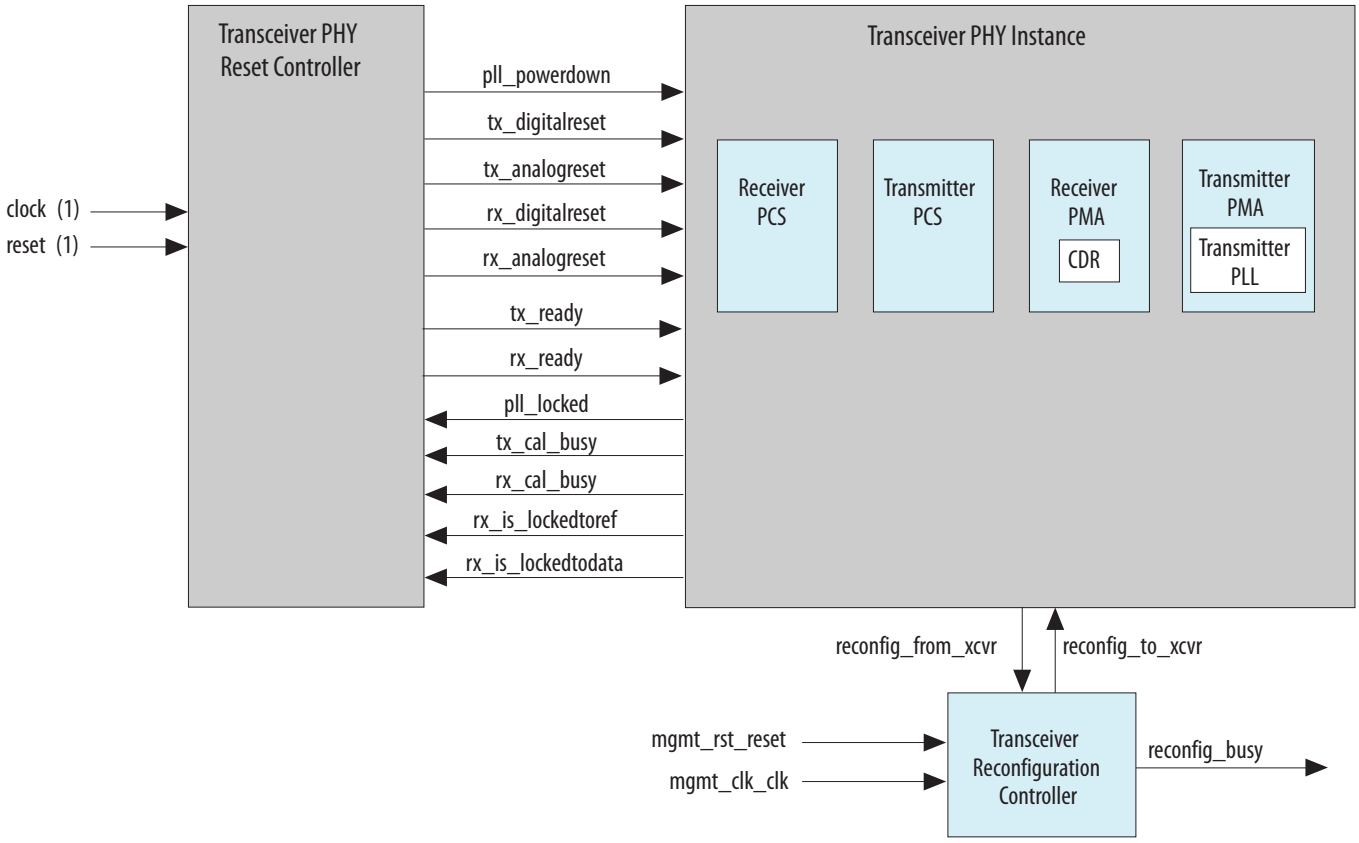
UG-01080 Subscribe Send Feedback

The Transceiver PHY Reset Controller IP Core is a highly configurable core that you can use to reset transceivers in Arria V, Arria V GZ, Cyclone V, or Stratix V devices. This reset controller is an alternate controller that you can use instead of the embedded reset controller for the Custom, Low Latency, and Deterministic Latency PHY IP cores. And, you can use it to reset the Stratix V, Arria V, Arria V GZ, and Cyclone V Native Transceiver PHYs which do not include imbedded reset controllers. You can use it to specify a custom reset sequence. You can also modify the clear text Verilog HDL file provided to implement custom reset logic. The Reset Controller handles all reset sequencing of the transceiver to enable successful operation. It provides the functionality of the embedded reset controller and the following additional options:

- Separate or shared reset controls per channel
- Separate controls for the TX and RX channels and PLLs
- Synchronization of the reset inputs
- Hysteresis for PLL locked status inputs
- Configurable reset timings
- Automatic or manual reset recovery mode

Figure 18-1: Typical System Diagram for the Transceiver PHY Reset Controller IP Core

This figure illustrates the typical use of Transceiver PHY Reset Controller in a design that includes a transceiver PHY instance and the Transceiver Reconfiguration Controller IP Core. You can use the phy_mgmt_clk and phy_mgmt_clk_reset as the clock and reset to the user-controller reset logic.



(20)

As figure illustrates, the Transceiver PHY Reset Controller connects to a Transceiver PHY. The Transceiver PHY Reset Controller IP Core drives TX and RX resets to the Transceiver PHY and receives status from the Transceiver PHY. Depending on the components in the design, the calibration busy signal may be an output of the Transceiver PHY or the Transceiver Reconfiguration Controller. The following transceiver PHY IP support the removal of the embedded reset controller:

- Custom Transceiver PHY IP Core
- Low Latency PHY IP Core
- Deterministic Latency PHY IP Core
- Arria V and Stratix V Native PHY IP Cores

These transceiver PHYs drive the TX and RX calibration busy signals to the Transceiver PHY Reset Controller IP Core.

(20) You can use the phy_mgmt_clk and phy_mgmt_clk_reset as the clock and reset to the user-controller reset logic.

Related Information

- **Transceiver Reset Control in Arria V Devices**
- **Transceiver Reset Control in Cyclone V Devices**
- **Transceiver Reset Control in Stratix V Devices**

Device Family Support for Transceiver PHY Reset Controller

This section describes the transceiver PHY reset controller IP core device family support.

IP cores provide either final or preliminary support for target Altera device families. These terms have the following definitions:

- Final support—Verified with final timing models for this device.
- Preliminary support—Verified with preliminary timing models for this device.

Table 18-1: Device Family Support

This table lists the level of support offered by the Transceiver PHY Reset Controller IP core for Altera device families.

Device Family	Support
Cyclone V devices	Final
Arria V devices	Final
Arria V GZ	Final
Stratix V devices	Final
Other device families	No support

Performance and Resource Utilization for Transceiver PHY Reset Controller

This section describes the performance and resource utilization for the transceiver PHY reset controller.

Table 18-2: Reset Controller Resource Utilization—Stratix V Devices

This table lists the typical expected device resource utilization, rounded to the nearest 50, for two configurations using the current version of the Intel Quartus Prime software targeting a Stratix V GX device. The numbers are rounded to the nearest 50.

Configuration	Combinational ALUTs	Logic Registers
Single channel	50	50
4 channels, shared TX reset, separate RX resets	100	150

Parameterizing the Transceiver PHY Reset Controller IP

This section lists steps to configure the Transceiver PHY Reset Controller IP Core in the IP Catalog. You can customize the following Transceiver PHY Reset Controller parameters for different modes of operation by clicking **Tools** > **IP Catalog**.

To parameterize and instantiate the Transceiver PHY Reset Controller IP core:

1. For **Device Family**, select your target device from the list.
2. Click **Installed IP** > **Library** > **Interface Protocols** > **Transceiver PHY** > **Transceiver PHY Reset Controller**.
3. Select the options required for your design. For a description of these options, refer to the **Transceiver PHY Reset Controller Parameters**.
4. Click **Finish**. The wizard generates files representing your parameterized IP variation for synthesis and simulation.

Transceiver PHY Reset Controller Parameters

The Intel Quartus Prime software provides a GUI to define and instantiate a Transceiver PHY Reset Controller to reset transceiver PHY and external PLL.

Table 18-3: General Options

Name	Range	Description
Number of transceiver channels	`1-1000`	Specifies the number of channels that connect to the Transceiver PHY Reset Controller IP core. The upper limit of the range is determined by your FPGA architecture.
Number of TX PLLs	`1-1000`	Specifies the number of TX PLLs that connect to the Transceiver PHY Reset Controller IP core.
Input clock frequency	`1-500 MHz`	Input clock to the Transceiver PHY Reset Controller IP core. The frequency of the input clock in MHz. The upper limit on the input clock frequency is the frequency achieved in timing closure.
Synchronize reset input	**On /Off**	When **On**, the Transceiver PHY Reset Controller synchronizes the reset to the Transceiver PHY Reset Controller input clock before driving it to the internal reset logic. When **Off**, the reset input is not synchronized.
Use fast reset for simulation	**On /Off**	When **On**, the Transceiver PHY Reset Controller uses reduced reset counters for simulation.
Separate interface per channel/ PLL	**On /Off**	When **On**, the Transceiver PHY Reset Controller provides a separate reset interface for each channel and PLL.
TX PLL		

Name	Range	Description
Enable TX PLL reset control	**On /Off**	When **On**, the Transceiver PHY Reset Controller IP core enables the reset control of the TX PLL. When **Off**, the TX PLL reset control is disabled.
pll_powerdown duration	`1-999999999`	Specifies the duration of the PLL powerdown period in ns. The value is rounded up to the nearest clock cycle. The default value is 1000 ns.
Synchronize reset input for PLL powerdown	**On /Off**	When **On**, the Transceiver PHY Reset Controller synchronizes the PLL powerdown reset with the Transceiver PHY Reset Controller input clock. When **Off**, the PLL powerdown reset is not synchronized.
TX Channel		
Enable TX channel reset control	**On /Off**	When **On**, the Transceiver PHY Reset Controller enables the control logic and associated status signals for TX reset. When **Off**, disables TX reset control and status signals.
Use separate TX reset per channel	**On /Off**	When **On**, each TX channel has a separate reset. When **Off**, the Transceiver PHY Reset Controller uses a shared TX reset controller for all channels.
TX digital reset mode	**Auto**, **Manual**, **Expose Port**	Specifies the Transceiver PHY Reset Controller behavior when the `pll_locked` signal is deasserted. The following modes are available: • **Auto**—The associated `tx_digitalreset` controller automatically resets whenever the `pll_locked` signal is deasserted. Intel recommends this mode. • **Manual**—The associated `tx_digitalreset` controller is not reset when the `pll_locked` signal is deasserted, allowing you to choose corrective action. • **Expose Port**—The `tx_manual` signal is a top-level signal of the IP core. You can dynamically change this port to Auto or Manual. (1= Manual , 0 = Auto)
tx_analogreset duration	`1-999999999`	Specifies the time in ns to continue to assert `tx_analoglreset` after the reset input and all other gating conditions are removed. The value is rounded up to the nearest clock cycle. **Note:** Model 1 requires this to be set to 70 µs. Select the **Arria 10 Default Settings** preset.

Name	Range	Description
tx_digitalreset duration	`1-999999999`	Specifies the time in ns to continue to assert the `tx_digitalreset` after the reset input and all other gating conditions are removed. The value is rounded up to the nearest clock cycle. **Note:** Model 1 requires this to be set to 70 µs. Select the **<Device> Default Settings** preset. The default value for Model 2 is 20 ns.
pll_locked input hysteresis	`0-999999999`	Specifies the amount of hysteresis in ns to add to the `pll_locked` status input to filter spurious unreliable assertions of the `pll_locked` signal. A value of 0 adds no hysteresis. A higher value filters glitches on the `pll_locked` signal. Intel recommends that the amount of hysteresis be longer than t_{req} = 70 µs.
RX Channel		
Enable RX channel reset control	**On /Off**	When enabled, the IP enables control logic and status signals for the RX reset signals.
Use separate RX reset per channel	**On /Off**	When **On**, each RX channel has a separate reset input. When **Off**, uses a shared RX reset controller for all channels.
RX digital reset mode	**Auto**, **Manual**, **Expose Port**	Specifies the Transceiver PHY Reset Controller behavior when the PLL lock signal is deasserted. The following modes are available: • **Auto**—The associated `rx_digitalreset` controller automatically resets whenever the `rx_is_lockedtodata` signal is deasserted. • **Manual**—The associated `rx_digitalreset` controller is not reset when the `rx_is_lockedtodata` signal is deasserted, allowing you to choose corrective action. • **Expose Port**—The `rx_manual` signal is a top-level signal of the IP core. If the core includes separate reset control for each RX channel, each RX channel uses its respective `rx_is_lockedtodata` signal for automatic reset control; otherwise, the inputs are ANDed to provide internal status for the shared reset controller.

Name	Range	Description
rx_analogreset duration	1-999999999	Specifies the time in ns to continue to assert the rx_analogreset after the reset input and all other gating conditions are removed. The value is rounded up to the nearest clock cycle. The default value is 40 ns. **Note:** Model 1 requires this to be set to 70 µs. Select the **<Device> Default Settings** preset.
rx_digitalreset duration	1-999999999	Specifies the time in ns to continue to assert the rx_digitalreset after the reset input and all other gating conditions are removed. The value is rounded up to the nearest clock cycle. The default value is 4000 ns.

Transceiver PHY Reset Controller Interfaces

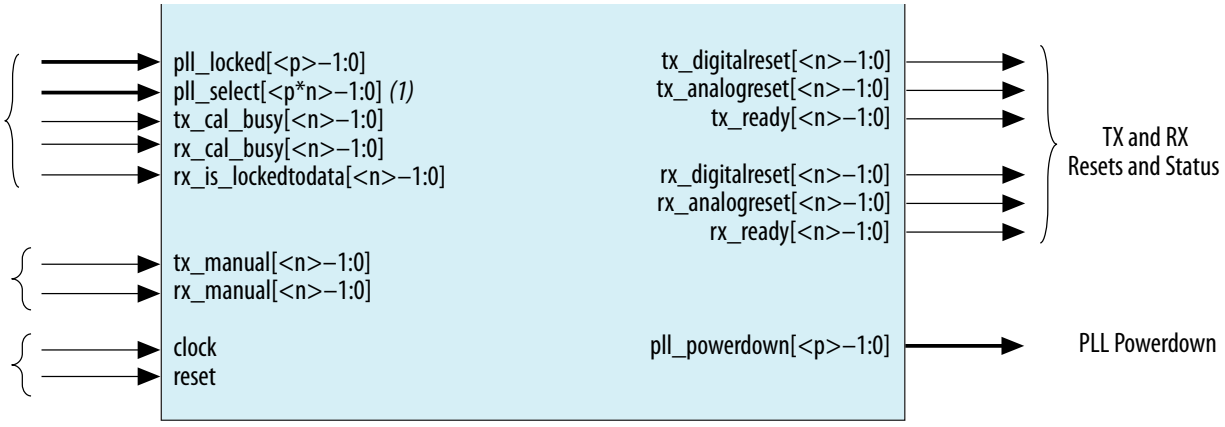
This section describes the top-level signals for the Transceiver PHY Reset Controller IP core.

The following figure illustrates the top-level signals of the Transceiver PHY Reset Controller IP core. Many of the signals in the figure become buses if you choose separate reset controls. The variables in the figure represent the following parameters:

- <*n*>—The number of lanes
- <*p*>—The number of PLLs

Figure 18-2: Transceiver PHY Reset Controller IP Core Top-Level Signals

Generating the IP core creates signals and ports based on your parameter settings.



pll_select signal width when a single TX reset sequence is used for all channels.

Note: PLL control is available when you enable the **Expose Port** parameter.

Table 18-4: Top-Level Signals

This table describes the signals in the above figure in the order that they are shown in the figure.

Signal Name	Direction	Clock Domain	Description
`pll_locked[<p>-1:0]`	Input	Asynchronous	Provides the PLL locked status input from each PLL. When asserted, indicates that the TX PLL is locked. When deasserted, the PLL is not locked. There is one signal per PLL.
`pll_select[<p*n>-1:0]`	Input	Synchronous to the Transceiver PHY Reset Controller input clock. Set to zero when not using multiple PLLs.	When you select **Use separate TX reset per channel**, this bus provides enough inputs to specify an index for each `pll_locked` signal to listen to for each channel. When **Use separate TX reset per channel** is disabled, the `pll_select` signal is used for all channels. n=1 when a single TX reset sequence is used for all channels.
`tx_cal_busy[<n> -1:0]`	Input	Asynchronous	This is the calibration status signal that results from the logical OR of `pll_cal_busy` and `tx_cal_busy` signals. The signal goes high when either the TX PLL or Transceiver PHY initial calibration is active. It is not asserted if you manually re-trigger the calibration IP. The signal goes low when calibration is completed. This signal gates the TX reset sequence. The width of this signals depends on the number of TX channels.
`rx_cal_busy[<n> -1:0]`	Input	Asynchronous	This is calibration status signal from the Transceiver PHY IP core. When asserted, the initial calibration is active. When deasserted, calibration has completed. This signal gates the RX reset sequence. The width of this signals depends on the number of RX channels.
`rx_is_lockedto-data[<n>-1:0]`	Input	Synchronous to CDR	Provides the `rx_is_lockedtodata` status from each RX CDR. When asserted, indicates that a particular RX CDR is ready to receive input data. If you do not choose separate controls for the RX channels, these inputs are ANDed together internally to provide a single status signal.

Signal Name	Direction	Clock Domain	Description
tx_manual[<n>-1:0]	Input	Asynchronous	This optional signal places tx_digitalreset controller under automatic or manual control. When asserted, the associated tx_digital-reset controller logic does not automatically respond to deassertion of the pll_locked signal. However, the initial tx_digitalreset sequence still requires a one-time rising edge on pll_locked before proceeding. When deasserted, the associated tx_digitalreset controller automatically begins its reset sequence whenever the selected pll_locked signal is deasserted.
rx_manual[<n> - 1:0]	Input	Asynchronous	This optional signal places rx_digitalreset logic controller under automatic or manual control. In manual mode, the rx_digital-reset controller does not respond to the assertion or deassertion of the rx_is_lockedtodata signal. The rx_digitalreset controller asserts rx_ready when the rx_is_lockedtodata signal is asserted.
clock	Input	N/A	A free running system clock input to the Transceiver PHY Reset Controller from which all internal logic is driven. If a free running clock is not available, hold reset until the system clock is stable.
reset	Input	Asynchronous	Asynchronous reset input to the Transceiver PHY Reset Controller. When asserted, all configured reset outputs are asserted. Holding the reset input signal asserted holds all other reset outputs asserted. An option is available to synchronize with the system clock. In synchronous mode, the reset signal needs to stay asserted for at least (2) clock cycles by default.
tx_digital-reset[<n>-1:0]	Output	Synchronous to the Transceiver PHY Reset Controller input clock.	Digital reset for TX channels. The width of this signal depends on the number of TX channels. This signal is asserted when any of the following conditions is true: • reset is asserted • pll_powerdown is asserted • pll_cal_busy is asserted • tx_cal_busy is asserted • PLL has not reached the initial lock (pll_locked deasserted) • pll_locked is deasserted and tx_manual is deasserted When all of these conditions are false, the reset counter begins its countdown for deassertion of tx_digitalreset.

Signal Name	Direction	Clock Domain	Description
tx_analogreset[<n>-1:0]	Output	Synchronous to the Transceiver PHY Reset Controller input clock.	Analog reset for TX channels. The width of this signal depends on the number of TX channels. This signal is asserted when reset is asserted. This signal follows pll_powerdown, which is deasserted after pll_locked goes high.
tx_ready[<n>-1:0]	Output	Synchronous to the Transceiver PHY Reset Controller input clock.	Status signal to indicate when the TX reset sequence is complete. This signal is deasserted while the TX reset is active. It is asserted a few clock cycles after the deassertion of tx_digitalreset. Some protocol implementations may require you to monitor this signal prior to sending data. The width of this signal depends on the number of TX channels.
rx_digital-reset[<n> -1:0]	Output	Synchronous to the Transceiver PHY Reset Controller input clock.	Digital reset for RX. The width of this signal depends on the number of channels. This signal is asserted when any of the following conditions is true: • reset is asserted • rx_analogreset is asserted • rx_cal_busy is asserted • rx_is_lockedtodata is deasserted and rx_manual is deasserted When all of these conditions are false, the reset counter begins its countdown for deassertion of rx_digitalreset.
rx_analogreset[<n>-1:0]	Output	Synchronous to the Transceiver PHY Reset Controller input clock.	Analog reset for RX. When asserted, resets the RX CDR and the RX PMA blocks of the transceiver PHY. This signal is asserted when any of the following conditions is true: • reset is asserted • rx_cal_busy is asserted The width of this signal depends on the number of channels.
rx_ready[<n>-1:0]	Output	Synchronous to the Transceiver PHY Reset Controller input clock.	Status signal to indicate when the RX reset sequence is complete. This signal is deasserted while the RX reset is active. It is asserted a few clock cycles after the deassertion of rx_digitalreset. Some protocol implementations may require you to monitor this signal prior to sending data. The width of this signal depends on the number of RX channels.

Signal Name	Direction	Clock Domain	Description
pll_powerdown[<p>-1:0]	Output	Synchronous to the Transceiver PHY Reset Controller input clock.	Asserted to power down a transceiver PLL circuit. When asserted, the selected TX PLL is reset.

Usage Examples for pll_select

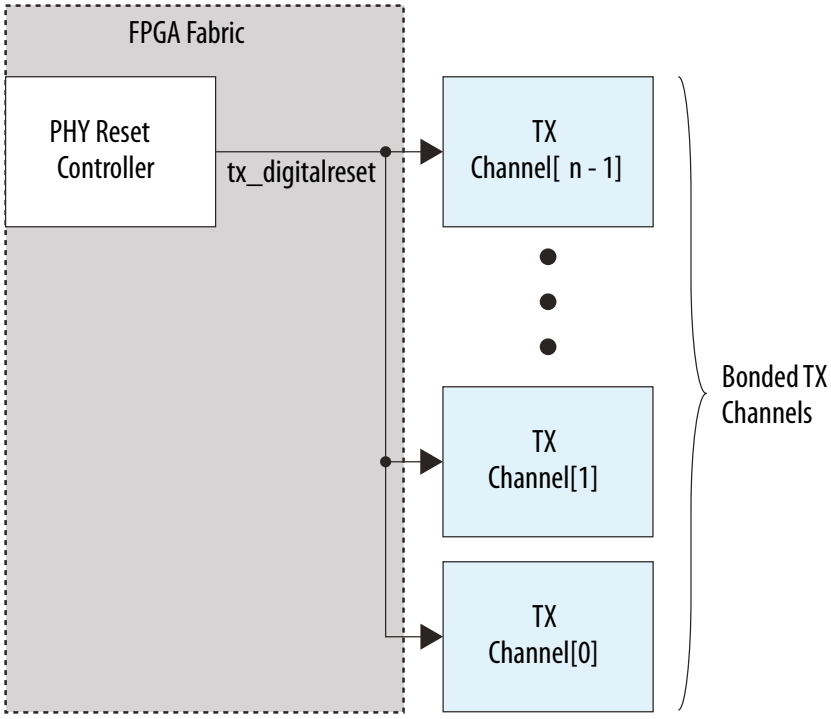
- If a single channel can switch between three TX PLLs, the pll_select signal indicates which one of the selected three TX PLL's pll_locked signal is used to communicate the PLL lock status to the TX reset sequence. In this case, to select the 3-bits wide pll_locked port, the pll_select port is 2-bits wide.
- If three channels are instantiated with three TX PLLs and with a separate TX reset sequence per channel, the pll_select field is 6-bits wide (2-bits per channel). In this case, pll_select [1:0] represents channel 0, pll_select[3:2] represents channel 1, and pll_select[5:4] represents channel 2. For each channel, a separate pll_locked signal indicates the PLL lock status.
- If three channels are instantiated with three TX PLLs and with a single TX reset sequence for all three channels, then pll_select field is 2-bits wide. In this case, the same pll_locked signal indicates the PLL lock status for all three channels.
- If one channel is instantiated with one TX PLL, pll_select field is 1-bit wide. Connect pll_select to logic 0.
- If three channels are instantiated with only one TX PLL and with a separate TX reset sequence per channel, the pll_select field is 3-bits wide. In this case, pll_select should be set to 0 since there is only one TX PLL available.

Timing Constraints for Bonded PCS and PMA Channels

For designs that use TX **PMA and PCS Bonding**, the digital reset signal (tx_digitalreset) to all TX channels within a bonded group must meet a maximum skew tolerance imposed by physical routing. This skew tolerance is one-half the TX parallel clock cycle (tx_clkout). This requirement is not necessary for TX **PMA Bonding** or for RX PCS channels.

Note: If the design is not able to meet the maximum skew tolerance requirement with a positive margin, Intel recommends reassigning the channels locations that are not adjacent to the PCIe Hard IP block.

Figure 18-3: Physical Routing Delay Skew in Bonded Channels



You must provide a Synopsys Design Constraint (SDC) for the reset signals to guarantee that your design meets timing requirements. The Quartus Prime software generates an `.sdc` file when you generate the Transceiver Native PHY IP core.

This `.sdc` contains basic false paths for most asynchronous signals, including resets. In the case of bonded designs, this file contains examples for maximum skew on bonded designs. This `.sdc` file contains an example `false_path` and an example `max_skew` constraint for the `tx_digitalreset` signals.

All modified IP constraints from a generated `.sdc` file must be moved to the project's main `.sdc` file, because changes are lost if the IP is regenerated.

This skew is present whether you tie all `tx_digitalresets` together, or you control them separately. If your design includes the Transceiver PHY Reset Controller IP core, you can substitute your instance and interface names for the generic names shown in the example.

Example 18-1: SDC Constraint for TX Digital Reset When Bonded Clocks Are Used

```
set_max_skew -from *<IP_INSTANCE_NAME> *tx_digitalreset*r_reset
-to *pld_pcs_interface* <1/2 coreclk period in ps>
```

In the above example, you must make the following substitutions:

- *<IP_INSTANCE_NAME>*—substitute the name of your reset controller IP instance or PHY IP instance
- *<½ coreclk period in ps>*—substitute half of the clock period of your design in picoseconds

If your design has custom reset logic, replace the `*<IP_INSTANCE_NAME>*tx_digitalreset*r_reset` with the source register for the TX PCS reset signal, `tx_digitalreset`.

For more information about the `set_max_skew` constraint, refer to the *SDC and Timing Analyzer API Reference Manual*.

Related Information

SDC and Timing Analyzer API Reference Manual

Transceiver PLL IP Core for Stratix V, Arria V, and Arria V GZ Devices 19

2020.06.02

UG-01080

When a fractional PLL functions as the TX PLL, you must configure the Native PHY IP Core to use external PLLs. If you also want to use CMU or ATX PLLs, you must use the device-specific Transceiver PLL to instantiate them.

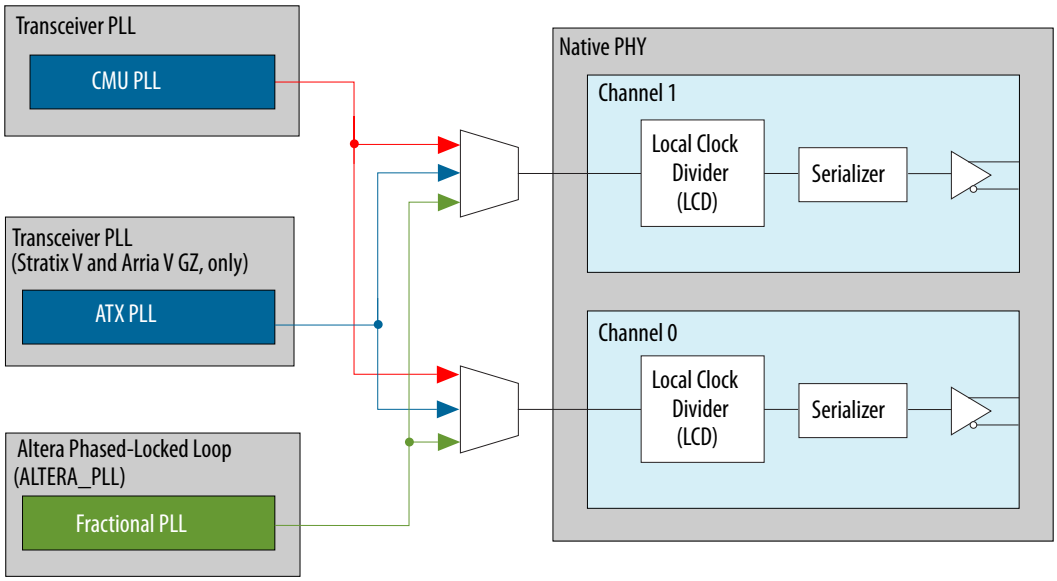
The MegaCore Library includes the following IP cores to instantiate external CMU and ATX PLLs:

- Stratix V Transceiver PLL
- Arria V Transceiver PLL
- Arria V GZ Transceiver PLL

You instantiate the Altera Phase-Locked Loop (ALTERA_ PLL) IP Core to specify the fractional PLL and the Native PHY IP Core to specify the PMA and PCS settings. In the Native PHY GUI, select the **Use external TX PLL** option under the **TX PLL Options** heading. When you choose this option, the Native PHY includes a top-level bus, `ext_pll_clk[<p>-1:0]` that you can connect to the external CMU, ATX, and fractional PLLs. To achieve different TX channel data rates, you create point-to-point connections between `ext_pll_clk[<p>-1:0]` and the CMU, ATX, and fractional PLLs required.

Figure 19-1: IP Cores Required for Designs Using the Fractional PLL

The following figure show the IP Cores you can instantiate to create designs that use a fractional PLL as the TX PLL. The figure also illustrates the use of Transceiver PLL to instantiate CMU and ATX PLLs. The MegaCore Library includes separate Transceiver PLL and Native PHY IP Cores for each V-Series Device Family. This figure shows logical connectivity between IP Cores and does not reflect the physical location of hardware in V-Series devices.



Designs that dynamically reconfigure the TX PLL between the CMU PLL and fractional PLL, must also select **Use external TX PLL** in the Native PHY GUI and instantiate all PLLs externally as shown in the figure above. Dynamic reconfiguration is only supported for non-bonded configurations. Dynamic reconfiguration allows you to implement the following features:

- TX PLL reconfiguration between up to 5 input reference clocks
- PLL switching using the x1 clock lines within a transceiver triplet
- PLL switching using the x6 and xN clock lines when the TX channels are not in the same transceiver bank

Note: It is not recommended to use fractional PLL in fractional mode for transceiver applications as a TX PLL or for PLL cascading.

Related Information

- **Analog Settings for Arria V Devices** on page 20-2
- **Analog Settings for Arria V GZ Devices** on page 20-11
- **Analog Settings for Stratix V Devices** on page 20-35
- **Altera Phase-Locked Loop (ALTERA_PLL) Megafunction User Guide**

Parameterizing the Transceiver PLL PHY

The IP Catalog provides the following Transceiver PLL IP Cores: Arria V Transceiver, Arria V GZ Transceiver PLL, and Stratix V Transceiver PLL to be used with the Arria V, Arria V GZ and Stratix V Native PHYs, respectively.

Complete the following steps to configure a Transceiver PLL IP Core:

1. Under **Tools** > **IP Catalog**, select the device familyof your choice.
2. Under **Tools** > **IP Catalog** > **Interface Protocols** > **Transceiver PHY**> **<Device> Transceiver PLL <ver>**.
3. Specify the options required for the PLL.
4. Click **Finish** to generate your parameterize Transceiver PLL IP Core.

Transceiver PLL Parameters

Table 19-1: PLL Reconfigurations

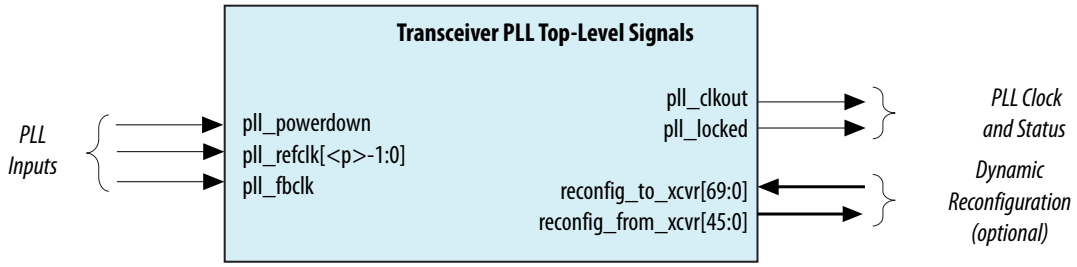
Name	Value	Description
Enable PLL Reconfiguration	**On/Off**	You must enable this option if you plan to reconfigure the PLLs in your design. This option is also required to simulate PLL reconfiguration.
Number of TX reference clocks	**1-5**	Specifies the number of reference clocks inputs to the Transceiver PLL.
PLL feedback path	**Internal** **External**	Select the **External** feedback path for the CPRI protocol to improve clock jitter by using an external voltage controlled crystal oscillator (VCXO). Select **Internal** for all other protocols.
PLL Type	**CMU** **ATX**	Specifies the PLL type. You must select the **CMU** PLL for designs that also include a fractional PLL. The ATX pll is available for Stratix V and Arria V GZ devices.
PLL base data rate	**1 × Lane rate** **2 × Lane rate** **4 × Lane rate** **8 × Lane rate**	Specifies **Base data rate**. This value should match the value specified in the Native PHY.

Name	Value	Description
Reference clock frequency	Variable	Specifies the frequency of the PLL input reference clock. The PLL must generate an output frequency that equals the **Base data rate**/2. You can use any **Input clock frequency** that allows the PLLs to generate this output frequency.
Selected reference clock source	**0-4**	Specifies the index of the **TX reference clock** for the initial configuration of the TX PLL. Logical index 0 corresponds to **TX reference clock** 1, and so on.

Transceiver PLL Signals

Figure 19-2: Transceiver PLL Top-Level Signals

The following figure illustrates the top-level signals of the Transceiver PLL which are defined in the table below.



Signal Name	Direction	Description
`pll_powerdown`	Input	When asserted, powers down the PLL.
`pll_refclk`	Input	Input reference clock for the CMU PLL.
`pll_fbclk`	Input	The feedback input port for the PLL.
`pll_clkout`	Output	Output clock from the PLL.
`pll_locked`	Output	When asserted, indicates that the PLL has locked to the input reference clock.
`reconfig_to_xcvr[69:0]`	Input	Reconfiguration signals from the Transceiver Reconfiguration Controller. When you enable the reconfiguration bus, the simulation model for the TX PLL supports dynamic reconfiguration. When you enable this bus, the Intel Quartus Prime software does not merge TX PLL by default; however, you can merge TX PLLs using QSF settings.

Signal Name	Direction	Description
reconfig_from_xcvr[45:0]	Output	Reconfiguration signals to the Transceiver Reconfiguration Controller.

Related Information

Component Interface Tcl Reference

Analog Parameters Set Using QSF Assignments 20

2020.06.02

UG-01080

You specify the analog parameters using the Intel Quartus Prime Assignment Editor, the Pin Planner, or through the Intel Quartus Prime Settings File (**.qsf**). The default values for analog options fall into three categories:

- *Global*— These parameters have default values that are independent of other parameter settings.
- *Computed*—These parameters have an initial default value that is recomputed based on other parameter settings.
- *Proxy*—These parameters have default values that are place holders. The Intel Quartus Prime software selects these initial default values based on your design; however, Altera recommends that you replace these defaults with values that match your electrical board specification.

For more information about the Pin Planner, refer to About the Pin Planner in Intel Quartus Prime Help. For more information about the Assignment Editor, refer to About the Assignment Editor in Intel Quartus Prime Help.

For more information about Intel Quartus Prime Settings, refer to *Intel Quartus Prime Settings File Manual*.

Related Information

- **About the Pin Planner**
- **About the Assignment Editor**
- **Intel Quartus Prime Settings File Manual**

Making QSF Assignments Using the Assignment Editor

The Intel Quartus Prime software provides default values for analog parameters. You can change the default values using the Assignment Editor. For example, complete the following steps to specify a 3.0V supply for the VCCA voltage and a 1.0V supply to the VCCR voltages.

1. On the Assignments menu, select **Assignment Editor**. The Assignment Editor appears.
2. Complete the following steps for each pin requiring the VCCA voltage:
   a. Double-click in the **Assignment Name** column and scroll to the bottom of the available assignments.
   b. Select **VCCA_GXB Voltage**.
   c. In the **Value** column, select **3_0V** from the list.
3. Complete the following steps for each pin requiring the VCCR voltage:

a. Double-click in the **Assignment Name** column and scroll to the bottom of the available assignments.
b. Select **VCCR_GXB/VCCT_GXB Voltage**.
c. In the **Value** column, select **1_0V** from the list.

The Intel Quartus Prime software adds these instance assignments commands to the **.qsf** file for your project.

Analog Settings for Arria V Devices

Analog Settings for Arria V Devices

This section lists the analog parameters for Arria V devices whose original values are place holders for the values that match your electrical board specification. In the following table, the default value of an analog parameter is shown in **bold** type. The parameters are listed in alphabetical order.

The following table lists the analog parameters with *global* or *computed* default values. You may want to optimize some of these settings. The default value is shown in **bold** type. For computed analog parameters, the default value listed is for the initial setting, not the recomputed setting. The parameters are listed in alphabetical order.

For more information about the Pin Planner, refer to About the Pin Planner in Intel Quartus Prime Help. For more information about the Assignment Editor, refer to About the Assignment Editor in Intel Quartus Prime Help.

For more information about Intel Quartus Prime Settings, refer to *Intel Quartus Prime Settings File Manual*.

Related Information

- **PCI Express Card Electromechanical Specification Rev. 2.0**
- **About the Pin Planner**
- **About the Assignment Editor**
- **Intel Quartus Prime Settings File Manual**

XCVR_IO_PIN_TERMINATION

Pin Planner and Assignment Editor Name

Transceiver I/O Pin Termination

Description

Specifies the intended on-chip termination value for the specified transceiver pin. Use External Resistor if you intend to use off-chip termination.

Options

- 85_Ohms
- **100_Ohms**
- 120_Ohms
- 150_Ohms
- External_Resistor

Assign To

Pin - TX & RX serial data

XCVR_REFCLK_PIN_TERMINATION

Pin Planner and Assignment Editor Name

Transceiver Dedicated Refclk Pin Termination

Description

Specifies the intended termination value for the specified refclk pin. The following 3 settings are available:

- AC_COUPLING: Altera recommends this setting for all transceiver designs. Use it for AC coupled signals. This setting implements on-chip termination and on-chip signal biasing.
- DC_COUPLING_ INTERNAL_100_OHMS: Used this setting when the dedicated transceiver reference clock pins are fed by a DC coupled signal whose V_{cm} meets the device specification. This assignment implements internal on-chip termination but not on-chip signal biasing.
- DC_COUPLING_EXTERNAL_RESISTOR: Use this assignment when the dedicated transceiver reference clock pins are fed by a DC coupled signal. This option does not implement internal on-chip termination or signal biasing. You must implement termination and signal biasing outside of the FPGA. This assignment is recommended for compliance with the *PCI Express Card Electromechanical Specification Rev. 2.0* and the HCSL IO Standard.

Options

- **AC_COUPLING**
- DC_COUPLING_INTERNAL_100_OHMS
- DC_COUPLING_EXTERNAL_RESISTOR

Assign To

Pin - PLL refclk pin

XCVR_TX_SLEW_RATE_CTRL

Pin Planner and Assignment Editor Name

Transmitter Slew Rate Control

Description

Specifies the slew rate of the output signal. The valid values span from the slowest rate to fastest rate with 1 representing the slowest rate.

Options

1-5

4

Assign To

Pin - TX serial data

XCVR_VCCR_ VCCT_VOLTAGE

Pin Planner and Assignment Editor Name

VCCR_GXB

VCCT_GXB Voltage

Description

Configures the VCCR_GXB and VCCT_GXB voltage for an GXB I/O pin by specifying the intended supply voltages for a GXB I/O pin.

Options

1_1V

1_2V

Assign To

Pin - TX & RX serial data

Analog Settings Having Global or Computed Values for Arria V Devices

The following analog parameters have *global* or *computed* default values. You may want to optimize some of these settings. The default value is shown in **bold** type. For computed analog parameters, the default value listed is for the initial setting, not the recomputed setting. The parameters are listed in alphabetical order.

CDR_BANDWIDTH_PRESET

Pin Planner and Assignment Editor Name

CDR Bandwidth Preset

Description

Specifies the CDR bandwidth preset setting

Options

- **Auto**
- Low
- Medium
- High

Assign To

PLL instance

PLL_BANDWIDTH_PRESET

Pin Planner and Assignment Editor Name

PLL Bandwidth Preset

Description

Specifies the PLL bandwidth preset setting

Options

- **Auto**
- Low
- Medium
- High

Assign To

PLL instance

XCVR_RX_DC_GAIN

Pin Planner and Assignment Editor Name

Receiver Buffer DC Gain Control

Description

Controls the amount of a stage receive-buffer DC gain.

Options

0 –1

Assign To

Pin - RX serial data

XCVR_ANALOG_SETTINGS_PROTOCOL

Pin Planner and Assignment Editor Name

Transceiver Analog Settings Protocol

Description

Specifies the protocol that a transceiver implements. When you use this setting for fully characterized devices, the Intel Quartus Prime software automatically sets the optimal values for analog settings, including the V_{OD}, pre-emphasis, and slew rate. For devices that are not fully characterized, the Intel Quartus Prime software specifies these settings using preliminary data. If you assign a value to `XCVR_ANALOG_SETTINGS_PROTOCOL`, you cannot assign a value for any settings that this parameter controls. For example, for PCIe, the `XCVR_ANALOG_SETTINGS_PROTOCOL` assigns a value to

XCVR_RX_BYPASS_EQ_STAGES_234. If you also assign a value to this parameter, a Intel Quartus Prime Fitter error results as shown in the following example:

Example 20-1: Error (21215)

```
Error resolving parameter "pm_rx_sd_bypass_eqz_stages_234" value
on instance "pci_interface_ddf2:u_pci_interface_2|
PCIE_8x8Gb_HARDIP_2:PCIe2_Interface.U_PCIE_CORE|
altpcie_sv_hip_ast_hwtcl:pcie_8x8gb_hardip_2_inst|
altpcie_hip_256_pipen1b:altpcie_hip_256_pipen1b
|sv_xcvr_pipe_native:g_xcvr.sv_xcvr_pipe_native|sv_xcvr_native:
inst_sv_xcvr_native|sv_pma:inst_sv_pma|sv_rx_pma:rx_pma.
sv_rx_pma_inst|rx_pmas[8].rx_pma.rx_pma_buf": Only one QSF
setting for the parameter is allowed.
```

Options

The following protocol values are defined:

- BASIC
- CPRI
- PCIE_GEN1
- PCIE_GEN2
- SATA1_I
- SATA1_M
- SATA2_1
- SATA2_M
- SATA2_X
- SRIO
- XAUI

Assign To

Pin - TX and RX serial data

XCVR_RX_COMMON_MODE_VOLTAGE

Pin Planner and Assignment Editor Name

Receiver Buffer Common Mode Voltage

Description

Receiver buffer common-mode voltage.

Note: Contact Altera for using this assignment.

Related Information

How to Contact Altera on page 22-46

XCVR_RX_LINEAR_EQUALIZER_CONTROL

Pin Planner and Assignment Editor Name

Receiver Linear Equalizer Control

Description

Static control for the continuous time equalizer in the receiver buffer. The equalizer has 3 settings from 0–2 corresponding to the increasing AC gain.

Options

0-2

Assign To

Pin - RX serial data

XCVR_RX_SD_ENABLE

Pin Planner and Assignment Editor Name

Receiver Signal Detection Unit Enable/Disable

Description

Enables or disables the receiver signal detection unit. During normal operation `NORMAL_SD_ON=FALSE`, otherwise `POWER_DOWN_SD=TRUE`.

Used for the PCIe PIPE PHY, SATA and SAS protocols.

Options

FALSE

TRUE

Assign To

Pin - RX serial data

XCVR_RX_SD_OFF

Pin Planner and Assignment Editor Name

Receiver Cycle Count Before Signal Detect Block Declares Loss Of Signal

Description

Number of parallel cycles to wait before the signal detect block declares loss of signal. Only used for the PCIe PIPE PHY, SATA, and SAS protocols.

Options

0–29

1

Assign To

Pin - RX serial data

XCVR_RX_SD_ON

Pin Planner and Assignment Editor Name

Receiver Cycle Count Before Signal Detect Block Declares Presence Of Signal

Description

Number of parallel cycles to wait before the signal detect block declares presence of signal. Only used for the PCIe PIPE PHY, SATA, and SAS protocols.

Options

0–16

1

Assign To

Pin - RX serial data

XCVR_RX_SD_THRESHOLD

Pin Planner and Assignment Editor Name

Receiver Signal Detection Voltage Threshold

Description

Specifies signal detection voltage threshold level, V_{th}. The following encodings are defined:

- SDLV_50MV=7
- SDLV_45MV=6
- SDLV_40MV=5
- SDLV_35MV=4
- SDLV_30MV=3
- SDLV_25MV=2
- SDLV_20MV=1
- SDLV_15MV=0

Only used for the PCIe PIPE PHY, SATA, and SAS protocols.

The signal detect output is high when the receiver peak-to-peak differential voltage (diff p-p) > V_{th} x 4. For example, a setting of 6 translates to peak-to-peak differential voltage of 180mV (4*45mV). The $V_{diff\ p\text{-}p}$ must be > 180mV to turn on the signal detect circuit.

Options

- 0-7
- **3**

Assign To

Pin - RX serial data

XCVR_TX_COMMON_MODE_VOLTAGE

Pin Planner and Assignment Editor Name

Transmitter Common Mode Driver Voltage

Description

Transmitter common-mode driver voltage.

Note: Contact Intel for using this assignment.

Related Information

How to Contact Altera on page 22-46

XCVR_TX_PRE_EMP_1ST_POST_TAP

Pin Planner and Assignment Editor Name

Transmitter Pre-emphasis First Post-Tap

Description

Specifies the first post-tap setting value.

Note: Legal values for this parameter vary with the data pattern and data rate. Refer to the *Arria V Device Datasheet* for more information.

Options

0–31

Assign To

Pin - TX serial data

Related Information

Arria V Device Datasheet

XCVR_TX_RX_DET_ENABLE

Pin Planner and Assignment Editor Name

Transmitter Receiver Detect Block Enable

Description

Enables or disables the receiver detector circuit at the transmitter.

Options

- TRUE
- **FALSE**

Assign To

Pin - TX serial data

XCVR_TX_RX_DET_MODE

Pin Planner and Assignment Editor Name

Transmitter Receiver Detect Block Mode

Description

Sets the mode for receiver detect block.

Options

0–15

Assign To

Pin - TX serial data

XCVR_TX_VOD

Pin Planner and Assignment Editor Name

Transmitter Differential Output Voltage

Description

Differential output voltage setting. The values are monotonically increasing with the driver main tap current strength.

Options

- 0–63
- **10**

Assign To

Pin - TX serial data

XCVR_TX_VOD_PRE_EMP_CTRL_SRC

Pin Planner and Assignment Editor Name

Transmitter V_{OD} Pre-emphasis Control Source

Description

When set to `DYNAMIC_CTL`, the PCS block controls the V_{OD} and pre-emphasis coefficients for PCI Express. When this assignment is set to `RAM_CTL` the V_{OD} and pre-emphasis are controlled by other assignments, such as `XCVR_TX_PRE_EMP_1ST_POST_TAP`.

Options

- **DYNAMIC_CTL**: for PCI Express
- **RAM_CTL**: for all other protocols

Assign To

Pin - TX serial data

Analog Settings for Arria V GZ Devices

Analog Settings for Arria V GZ Devices

This section lists the analog parameters for Arria V GZ devices whose original values are place holders for the values that match your electrical board specification. In the following table, the default value of an analog parameter is shown in **bold** type. The parameters are listed in alphabetical order.

The following table lists the analog parameters with *global* or *computed* default values. You may want to optimize some of these settings. The default value is shown in **bold** type. For computed analog parameters, the default value listed is for the initial setting, not the recomputed setting. The parameters are listed in alphabetical order.

For more information about the Pin Planner, refer to About the Pin Planner in Intel Quartus Prime Help. For more information about the Assignment Editor, refer to About the Assignment Editor in Intel Quartus Prime Help.

For more information about Intel Quartus Prime Settings, refer to *Intel Quartus Prime Settings File Manual*.

Related Information

- **PCI Express Card Electromechanical Specification Rev. 2.0**
- **Device Datasheet for Arria V Devices**
- **About the Pin Planner**
- **About the Assignment Editor**
- **Intel Quartus Prime Settings File Manual**

XCVR_IO_PIN_TERMINATION

Pin Planner and Assignment Editor Name

Transceiver I/O Pin Termination

Description

Specifies the intended on-chip termination value for the specified transceiver pin. Use External Resistor if you intend to use off-chip termination.

Options

- 85_Ohms
- **100_Ohms**
- 120_Ohms
- 150_Ohms
- External_Resistor

Assign To

Pin - TX & RX serial data

XCVR_REFCLK_PIN_TERMINATION

Pin Planner and Assignment Editor Name

Transceiver Dedicated Refclk Pin Termination

Description

Specifies the intended termination value for the specified refclk pin. The following 3 settings are available:

- AC_COUPLING: Altera recommends this setting for all transceiver designs. Use it for AC coupled signals. This setting implements on-chip termination and on-chip signal biasing.
- DC_COUPLING_ INTERNAL_100_OHMS: Used this setting when the dedicated transceiver reference clock pins are fed by a DC coupled signal whose V_{cm} meets the device specification. This assignment implements internal on-chip termination but not on-chip signal biasing.
- DC_COUPLING_EXTERNAL_RESISTOR: Use this assignment when the dedicated transceiver reference clock pins are fed by a DC coupled signal. This option does not implement internal on-chip termination or signal biasing. You must implement termination and signal biasing outside of the FPGA. This assignment is recommended for compliance with the *PCI Express Card Electromechanical Specification Rev. 2.0* and the HCSL IO Standard.

Options

- **AC_COUPLING**
- DC_COUPLING_INTERNAL_100_OHMS
- DC_COUPLING_EXTERNAL_RESISTOR

Assign To

Pin - PLL refclk pin

XCVR_RX_BYPASS_EQ_STAGES_234

Pin Planner and Assignment Editor Name

Receiver Equalizer Stage 2, 3, 4 Bypass

Description

Bypass continuous time equalizer stages 2, 3, and 4 to save power. This setting eliminates significant AC gain on the equalizer and is appropriate for chip-to-chip short range communication on a PCB. Assigning

a value to this setting and XCVR_ANALOG_SETTINGS_PROTOCOL results in a Intel Quartus Prime Fitter error as shown in the following example:

Error (21215)

```
Error resolving parameter "pm_rx_sd_bypass_eqz_stages_234" value
on instance "pci_interface_ddf2:u_pci_interface_2|
PCIE_8x8Gb_HARDIP_2:PCIe2_Interface.U_PCIE_CORE|
altpcie_sv_hip_ast_hwtcl:pcie_8x8gb_hardip_2_inst|
altpcie_hip_256_pipen1b:altpcie_hip_256_pipen1b
|sv_xcvr_pipe_native:g_xcvr.sv_xcvr_pipe_native|sv_xcvr_native:
inst_sv_xcvr_native|sv_pma:inst_sv_pma|sv_rx_pma:rx_pma.
sv_rx_pma_inst|rx_pmas[8].rx_pma.rx_pma_buf": Only one QSF
setting for the parameter is allowed.
```

Options

- **All_Stages_Enabled**
- Bypass_Stages

Assign To

Pin - RX serial data

Note: This setting can be used for data rates upto 5 Gbps for backplane applications, and 8 Gbps for chip-to-chip applications.

XCVR_TX_SLEW_RATE_CTRL

Pin Planner and Assignment Editor Name

Transmitter Slew Rate Control

Description

Specifies the slew rate of the output signal. The valid values span from the slowest rate to fastest rate with 1 representing the slowest rate.

Options

1-5

4

Assign To

Pin - TX serial data

XCVR_VCCA_VOLTAGE

Pin Planner and Assignment Editor Name

VCCA_GXB Voltage

Description

Configure the VCCA_GXB voltage for a GXB I/O pin by specifying the intended VCCA_GXB voltage for a GXB I/O pin. If you do not make this assignment the compiler automatically sets the correct VCCA_GXB voltage depending on the configured data rate, as follows:

- Data rate <= 6.5 Gbps: 2_5V
- Data rate > 6.5 Gbps: 3_0V.

Options

- 2_5V
- 3_0V

Assign To

Pin - TX & RX serial data

XCVR_VCCR_VCCT_VOLTAGE

Pin Planner and Assignment Editor Name

VCCR_GXB

VCCT_GXB Voltage

Description

Refer to the *Arria V GX, GT, GZ, SX, and ST Device Datasheet* for guidance on selecting a value.

Options

- 0_85V
- 1_0V

Assign To

Pin - TX & RX serial data

Related Information

Arria V GX, GT, GZ, SX, and ST Device Datasheet

Analog Settings Having Global or Computed Default Values for Arria V GZ Devices

The following analog parameters have *global* or *computed* default values. You may want to optimize some of these settings. The default value is shown in **bold** type. For computed analog parameters, the default value listed is for the initial setting, not the recomputed setting. The parameters are listed in alphabetical order.

CDR_BANDWIDTH_PRESET

Pin Planner and Assignment Editor Name

CDR Bandwidth Preset

Description

Specifies the CDR bandwidth preset setting

Options

- **Auto**
- Low
- Medium
- High

Assign To

PLL instance

master_ch_number

Pin Planner and Assignment Editor Name

Parameter (Assignment Editor Only)

Description

For the PHY IP Core for PCI Express (PIPE), specifies the channel number of the channel acting as the master channel for a single transceiver bank or 2 adjacent banks. This setting allows you to override the default master channel assignment for the PCS and PMA. The master channel must use a TX PLL that is in the same transceiver bank. Available for Gen1, Gen2, and Gen3 variants.

```
Example: set_parameter -name master_ch_number 4 -to
"<design>.pcie_i|altera_xcvr_pipe:<design>_inst|
sv_xcvr_pipe_nr:pipe_nr_inst|sv_xcvr_pipe_native:
transceiver_core"
```

Options

1, 4

Assign To

Include in **.qsf** file

Related Information

Transceiver Configurations in Arria V GZ Devices
Refer to *Advance [SIC] Channel Placement Guidelines for PIPE Configurations* in this document.

PLL_BANDWIDTH_PRESET

Pin Planner and Assignment Editor Name

PLL Bandwidth Preset

Description

Specifies the PLL bandwidth preset setting

Options

- **Auto**
- Low
- Medium
- High

Assign To

PLL instance

reserved_channel

Pin Planner and Assignment Editor Name

Parameter (Assignment Editor Only)

Description

Allows you to override the default channel placement of x8 variants. For the PHY IP Core for PCI Express (PIPE), you can use this QSF assignment in conjunction with the `master_ch_number` assignment to specify channel 4 as the master channel. Available for Gen1, Gen2, and Gen3 variants.

```
Example: set_parameter -name reserved_channel true
 -to "<design>.pcie_i|altera_xcvr_pipe:<design>
_inst|sv_xcvr_pipe_nr:pipe_nr_inst|sv_xcvr_pipe_native:
transceiver_core"
```

Options

TRUE

Assign To

Include in **.qsf** file

Related Information

Transceiver Configurations in Arria V GZ Devices

Refer to *Advance [SIC] Channel Placement Guidelines for PIPE Configurations* in this document.

XCVR_ANALOG_SETTINGS_PROTOCOL

Pin Planner and Assignment Editor Name

Transceiver Analog Settings Protocol

Description

Specifies the protocol that a transceiver implements. When you use this setting for fully characterized devices, the Intel Quartus Prime software automatically sets the optimal values for analog settings, including the V_{OD}, pre-emphasis, and slew rate. For devices that are not fully characterized, the Intel Quartus Prime software specifies these settings using preliminary data. If you assign a value to `XCVR_ANALOG_SETTINGS_PROTOCOL`, you cannot assign a value for any settings that this parameter controls. For example, for PCIe, the `XCVR_ANALOG_SETTINGS_PROTOCOL` assigns a value to

XCVR_RX_BYPASS_EQ_STAGES_234. If you also assign a value to this parameter, a Intel Quartus Prime Fitter error results as shown in the following example:

Example 20-2: Error (21215)

```
Error resolving parameter "pm_rx_sd_bypass_eqz_stages_234" value
on instance "pci_interface_ddf2:u_pci_interface_2|
PCIE_8x8Gb_HARDIP_2:PCIe2_Interface.U_PCIE_CORE|
altpcie_sv_hip_ast_hwtcl:pcie_8x8gb_hardip_2_inst|
altpcie_hip_256_pipen1b:altpcie_hip_256_pipen1b
|sv_xcvr_pipe_native:g_xcvr.sv_xcvr_pipe_native|sv_xcvr_native:
inst_sv_xcvr_native|sv_pma:inst_sv_pma|sv_rx_pma:rx_pma.
sv_rx_pma_inst|rx_pmas[8].rx_pma.rx_pma_buf": Only one QSF
setting for the parameter is allowed.
```

Options

The following protocol values are defined:

- BASIC
- CEI
- CPRI
- INTERLAKEN
- PCIE_GEN1
- PCIE_GEN2
- PCIE_GEN3
- QPI
- SFIS
- SONET
- SRIO
- TENG_1588
- TENG_BASER
- TENG_SDI
- XAUI

Assign To

Pin - TX and RX serial data

XCVR_RX_DC_GAIN

Pin Planner and Assignment Editor Name

Receiver Buffer DC Gain Control

Description

Controls the RX buffer DC gain for GX channels.

Options

1 –4

Assign To

Pin - RX serial data

XCVR_RX_LINEAR_EQUALIZER_CONTROL

Pin Planner and Assignment Editor Name

Receiver Linear Equalizer Control

Description

Static control for the continuous time equalizer in the receiver buffer. The equalizer has 16 settings from 0–15 corresponding to the increasing AC gain.

Options

1 –16

Assign To

Pin - RX serial data

XCVR_RX_COMMON_MODE_VOLTAGE

Pin Planner and Assignment Editor Name

Receiver Buffer Common Mode Voltage

Description

Receiver buffer common-mode voltage.

Note: Contact Altera for using this assignment.

Related Information

How to Contact Altera on page 22-46

XCVR_RX_ENABLE_LINEAR_EQUALIZER_PCIEMODE

Pin Planner and Assignment Editor Name

Receiver Linear Equalizer Control (PCI Express)

Description

If enabled equalizer gain control is driven by the PCS block for PCI Express. If disabled equalizer gain control is determined by the `XCVR_RX_LINEAR_EQUALIZER_SETTING`

Options

TRUE

FALSE

Assign To

Pin - RX serial data

XCVR_RX_SD_ENABLE

Pin Planner and Assignment Editor Name

Receiver Signal Detection Unit Enable/Disable

Description

Enables or disables the receiver signal detection unit. During normal operation `NORMAL_SD_ON=FALSE`, otherwise `POWER_DOWN_SD=TRUE`.

Used for the PCIe PIPE PHY, SATA and SAS protocols.

Options

FALSE

TRUE

Assign To

Pin - RX serial data

XCVR_RX_SD_OFF

Pin Planner and Assignment Editor Name

Receiver Cycle Count Before Signal Detect Block Declares Loss Of Signal

Description

Number of parallel cycles to wait before the signal detect block declares loss of signal. Only used for the PCIe PIPE PHY, SATA, and SAS protocols.

Options

0–29

Assign To

Pin - RX serial data

XCVR_RX_SD_ON

Pin Planner and Assignment Editor Name

Receiver Cycle Count Before Signal Detect Block Declares Presence Of Signal

Description

Number of parallel cycles to wait before the signal detect block declares presence of signal. Only used for the PCIe PIPE PHY, SATA, and SAS protocols.

Options

0–16

Assign To

Pin - RX serial data

XCVR_RX_SD_THRESHOLD

Pin Planner and Assignment Editor Name

Receiver Signal Detection Voltage Threshold

Description

Specifies signal detection voltage threshold level, V_{th}. The following encodings are defined:

- SDLV_50MV=7
- SDLV_45MV=6
- SDLV_40MV=5
- SDLV_35MV=4
- SDLV_30MV=3
- SDLV_25MV=2
- SDLV_20MV=1
- SDLV_15MV=0

For the PCIe PIPE PHY, SATA, and SAS.

The signal detect output is high when the receiver peak-to-peak differential voltage (diff p-p) > V_{th} x 4. For example, a setting of 6 translates to peak-to-peak differential voltage of 180mV (4*45mV). The $V_{diff\ p\text{-}p}$ must be > 180mV to turn on the signal detect circuit.

Options

- **0**-7

Assign To

Pin - RX serial data

XCVR_TX_COMMON_MODE_VOLTAGE

Pin Planner and Assignment Editor Name

Transmitter Common Mode Driver Voltage

Description

Transmitter common-mode driver voltage.

Note: Contact Intel for using this assignment.

Related Information

How to Contact Altera on page 22-46

XCVR_TX_PRE_EMP_PRE_TAP_USER

Pin Planner and Assignment Editor Name

Transmitter Pre-emphasis Pre-Tap user

Description

Specifies the TX pre-emphasis pretap setting value, including inversion.

Note: This parameter must be set in conjunction with `XCVR_TX_VOD`, `XCVR_TX_PRE_EMP_1ST_POST_TAP`, and `XCVR_TX_PRE_EMP_2ND_POST_TAP`. All combinations of these settings are not legal. Refer to the *Stratix V Device Datasheet* for more information.

Options

0–31

Assign To

Pin - TX serial data

Related Information

- **Solution rd02262013_691**
  This solution provides the mapping of the Transceiver Toolkit pretap settings to the Intel Quartus Prime transceiver QSF assignment.
- **Stratix V Device Datasheet**

XCVR_TX_PRE_EMP_2ND_POST_TAP_USER

Pin Planner and Assignment Editor Name

Transmitter Preemphasis Second Post-Tap user

Description

Specifies the transmitter pre-emphasis second post-tap setting value, including inversion.

Options

0–31

- For option value 1-15, the setting value is -15 to -1.
- For option value 17-31, the setting value is 1 to 15.
- For option value 0/16, the setting value is 0.

Assign To

Pin - TX serial data

Related Information

- **Solution rd02272013_545**
  This solution provides the mapping of the Transceiver Toolkit pretap settings to the Intel Quartus Prime transceiver QSF assignment.

- **Stratix V Device Datasheet**

XCVR_TX_PRE_EMP_1ST_POST_TAP

Pin Planner and Assignment Editor Name

Transmitter Pre-emphasis First Post-Tap

Description

Specifies the first post-tap setting value.

Note: This parameter must be set in conjunction with `XCVR_TX_VOD`, `XCVR_TX_PRE_EMP_2ND_POST_TAP`, and `XCVR_TX_PRE_EMP_PRE_TAP`. All combinations of these settings are not legal. Refer to the *Arria V GZ Device Datasheet* for more information.

Options

0–31

Assign To

Pin - TX serial data

Related Information

Arria V GZ Device Datasheet

XCVR_TX_PRE_EMP_2ND_POST_TAP

Pin Planner and Assignment Editor Name

Transmitter Pre-emphasis Second Post-Tap

Description

Specifies the second post-tap setting value.

Note: This parameter must be set in conjunction with `XCVR_TX_VOD`, `XCVR_TX_PRE_EMP_1ST_POST_TAP`, and `XCVR_TX_PRE_EMP_PRE_TAP`. All combinations of these settings are not legal. Refer to the *Arria V GZ Device Datasheet* for more information.

Options

0–15

Assign To

Pin - TX serial data

Related Information

Arria V GZ Device Datasheet

XCVR_TX_PRE_EMP_INV_2ND_TAP

Pin Planner and Assignment Editor Name

Transmitter Preemphasis Second Tap Invert

Description

Inverts the transmitter pre-emphasis 2nd post tap.

Options

- TRUE
- **FALSE**

Assign To

Pin - TX serial data

Related Information

Solution rd02262013_691
This solution provides the mapping of the Transceiver Toolkit pretap settings to the Intel Quartus Prime transceiver QSF assignment.

XCVR_TX_PRE_EMP_INV_PRE_TAP

Pin Planner and Assignment Editor Name

Transmitter Preemphasis Pre Tap Invert

Description

Inverts the transmitter pre-emphasis pretap. Specifies the TX pre-emphasis pretap setting value, including inversion.

Options

- TRUE
- **FALSE**

Assign To

Pin - TX serial data

Related Information

Solution rd02262013_691
This solution provides the mapping of the Transceiver Toolkit pretap settings to the Intel Quartus Prime transceiver QSF assignment.

XCVR_TX_PRE_EMP_PRE_TAP

Pin Planner and Assignment Editor Name

Transmitter Pre-emphasis Pre Tap

Description

Specifies the pre-tap pre-emphasis setting.

Note: This parameter must be set in conjunction with `XCVR_TX_VOD`, `XCVR_TX_PRE_EMP_1ST_POST_TAP`, and `XCVR_TX_PRE_EMP_2ND_POST_TAP`. All combinations of these settings are not legal. Refer to the *Arria V GZ Device Datasheet* for more information.

Options

0–15

Assign To

Pin - TX serial data

Related Information

Arria V GZ Device Datasheet

XCVR_TX_RX_DET_ENABLE

Pin Planner and Assignment Editor Name

Transmitter Receiver Detect Block Enable

Description

Enables or disables the receiver detector circuit at the transmitter.

Options

- TRUE
- **FALSE**

Assign To

Pin - TX serial data

XCVR_TX_RX_DET_MODE

Pin Planner and Assignment Editor Name

Transmitter Receiver Detect Block Mode

Description

Sets the mode for receiver detect block.

Options

0–15

Assign To

Pin - TX serial data

XCVR_TX_RX_DET_OUTPUT_SEL

Pin Planner and Assignment Editor Name

Transmitter's Receiver Detect Block QPI/PCI Express Control

Description

Determines QPI or PCI Express mode for the Receiver Detect block.

Options

- RX_DET_QPI_ OUT
- **RX_DET_PCIE_ OUT**

Assign To

Pin - TX serial data

XCVR_TX_VOD

Pin Planner and Assignment Editor Name

Transmitter Differential Output Voltage

Description

Differential output voltage setting. The values are monotonically increasing with the driver main tap current strength.

Note: This parameter must be set in conjunction with `XCVR_TX_PRE_EMP_1ST_POST_TAP`, `XCVR_TX_PRE_EMP_2ND_POST_TAP`, and `XCVR_TX_PRE_EMP_PRE_TAP`. All combinations of these settings are not legal. Refer to the *Arria V GZ Device Datasheet* for more information.

Options

- 0–63
- **50**

Assign To

Pin - TX serial data

Related Information

Arria V GZ Device Datasheet

XCVR_TX_VOD_PRE_EMP_CTRL_SRC

Pin Planner and Assignment Editor Name

Transmitter V_{OD} Pre-emphasis Control Source

Description

When set to `DYNAMIC_CTL`, the PCS block controls the V_{OD} and pre-emphasis coefficients for PCI Express. When this assignment is set to `RAM_CTL` the V_{OD} and pre-emphasis are controlled by other assignments, such as `XCVR_TX_PRE_EMP_1ST_POST_TAP`.

Options

- **DYNAMIC_CTL**: for PCI Express
- **RAM_CTL**: for all other protocols

Assign To

Pin - TX serial data

Analog Settings for Cyclone V Devices

XCVR_IO_PIN_TERMINATION

Pin Planner and Assignment Editor Name

Transceiver I/O Pin Termination

Description

Specifies the intended on-chip termination value for the specified transceiver pin. Use External Resistor if you intend to use off-chip termination.

Options

- 85_Ohms
- **100_Ohms**
- 120_Ohms
- 150_Ohms
- External_Resistor

Assign To

Pin - TX & RX serial data

XCVR_REFCLK_PIN_TERMINATION

Pin Planner and Assignment Editor Name

Transceiver Dedicated Refclk Pin Termination

Description

Specifies the intended termination value for the specified refclk pin. The following 3 settings are available:

- AC_COUPLING: Altera recommends this setting for all transceiver designs. Use it for AC coupled signals. This setting implements on-chip termination and on-chip signal biasing.
- DC_COUPLING_ INTERNAL_100_OHMS: Used this setting when the dedicated transceiver reference clock pins are fed by a DC coupled signal whose V_{cm} meets the device specification. This assignment implements internal on-chip termination but not on-chip signal biasing.
- DC_COUPLING_EXTERNAL_RESISTOR: Use this assignment when the dedicated transceiver reference clock pins are fed by a DC coupled signal. This option does not implement internal on-chip termination or signal biasing. You must implement termination and signal biasing outside of the FPGA. This assignment is recommended for compliance with the *PCI Express Card Electromechanical Specification Rev. 2.0* and the HCSL IO Standard.

Options

- **AC_COUPLING**
- DC_COUPLING_INTERNAL_100_OHMS
- DC_COUPLING_EXTERNAL_RESISTOR

Assign To

Pin - PLL refclk pin

XCVR_TX_SLEW_RATE_CTRL

Pin Planner and Assignment Editor Name

Transmitter Slew Rate Control

Description

Specifies the slew rate of the output signal. The valid values span from the slowest rate to fastest rate with 1 representing the slowest rate.

Options

1-5

4

Assign To

Pin - TX serial data

XCVR_VCCR_ VCCT_VOLTAGE

Pin Planner and Assignment Editor Name

VCCR_GXB

VCCT_GXB Voltage

Description

Configures the VCCR_GXB and VCCT_GXB voltage for an GXB I/O pin by specifying the intended supply voltages for a GXB I/O pin.

Options

1_1V

1_2V

Assign To

Pin - TX & RX serial data

Analog Settings Having Global or Computed Values for Cyclone V Devices

The following analog parameters have *global* or *computed* default values. You may want to optimize some of these settings. The default value is shown in **bold** type. For computed analog parameters, the default value listed is for the initial setting, not the recomputed setting. The parameters are listed in alphabetical order.

CDR_BANDWIDTH_PRESET

Pin Planner and Assignment Editor Name

CDR Bandwidth Preset

Description

Specifies the CDR bandwidth preset setting

Options

- **Auto**
- Low
- Medium
- High

Assign To

PLL instance

PLL_BANDWIDTH_PRESET

Pin Planner and Assignment Editor Name

PLL Bandwidth Preset

Description

Specifies the PLL bandwidth preset setting

Options

- **Auto**
- Low
- Medium
- High

Assign To

PLL instance

XCVR_ANALOG_SETTINGS_PROTOCOL

Pin Planner and Assignment Editor Name

Transceiver Analog Settings Protocol

Description

Specifies the protocol that a transceiver implements. When you use this setting for fully characterized devices, the Intel Quartus Prime software automatically sets the optimal values for analog settings, including the V_{OD}, pre-emphasis, and slew rate. For devices that are not fully characterized, the Intel Quartus Prime software specifies these settings using preliminary data. If you assign a value to `XCVR_ANALOG_SETTINGS_PROTOCOL`, you cannot assign a value for any settings that this parameter controls. For example, for PCIe, the `XCVR_ANALOG_SETTINGS_PROTOCOL` assigns a value to `XCVR_RX_BYPASS_EQ_STAGES_234`. If you also assign a value to this parameter, a Intel Quartus Prime Fitter error results as shown in the following example:

Example 20-3: Error (21215)

```
Error resolving parameter "pm_rx_sd_bypass_eqz_stages_234" value
on instance "pci_interface_ddf2:u_pci_interface_2|
PCIE_8x8Gb_HARDIP_2:PCIe2_Interface.U_PCIE_CORE|
altpcie_sv_hip_ast_hwtcl:pcie_8x8gb_hardip_2_inst|
altpcie_hip_256_pipen1b:altpcie_hip_256_pipen1b
|sv_xcvr_pipe_native:g_xcvr.sv_xcvr_pipe_native|sv_xcvr_native:
inst_sv_xcvr_native|sv_pma:inst_sv_pma|sv_rx_pma:rx_pma.
sv_rx_pma_inst|rx_pmas[8].rx_pma.rx_pma_buf": Only one QSF
setting for the parameter is allowed.
```

Options

The following protocol values are defined:

- BASIC
- CPRI
- PCIE_GEN1
- PCIE_GEN2
- SRIO
- XAUI

Assign To

Pin - TX and RX serial data

XCVR_RX_DC_GAIN

Pin Planner and Assignment Editor Name

Receiver Buffer DC Gain Control

Description

Controls the amount of a stage receive-buffer DC gain.

Options

0 –1

Assign To

Pin - RX serial data

XCVR_RX_LINEAR_EQUALIZER_CONTROL

Pin Planner and Assignment Editor Name

Receiver Linear Equalizer Control

Description

Static control for the continuous time equalizer in the receiver buffer. The equalizer has 3 settings from 0–2 corresponding to the increasing AC gain.

Options

0-2

Assign To

Pin - RX serial data

XCVR_RX_COMMON_MODE_VOLTAGE

Pin Planner and Assignment Editor Name

Receiver Buffer Common Mode Voltage

Description

Receiver buffer common-mode voltage.

Note: Contact Altera for using this assignment.

Related Information

How to Contact Altera on page 22-46

XCVR_RX_SD_ENABLE

Pin Planner and Assignment Editor Name

Receiver Signal Detection Unit Enable/Disable

Description

Enables or disables the receiver signal detection unit. During normal operation `NORMAL_SD_ON=FALSE`, otherwise `POWER_DOWN_SD=TRUE`.

Used for the PCIe PIPE PHY, SATA and SAS protocols.

Options

FALSE

TRUE

Assign To

Pin - RX serial data

XCVR_RX_SD_OFF

Pin Planner and Assignment Editor Name

Receiver Cycle Count Before Signal Detect Block Declares Loss Of Signal

Description

Number of parallel cycles to wait before the signal detect block declares loss of signal. Only used for the PCIe PIPE PHY, SATA, and SAS protocols.

Options

0–29

1

Assign To

Pin - RX serial data

XCVR_RX_SD_ON

Pin Planner and Assignment Editor Name

Receiver Cycle Count Before Signal Detect Block Declares Presence Of Signal

Description

Number of parallel cycles to wait before the signal detect block declares presence of signal. Only used for the PCIe PIPE PHY, SATA, and SAS protocols.

Options

0–16

1

Assign To

Pin - RX serial data

XCVR_RX_SD_THRESHOLD

Pin Planner and Assignment Editor Name

Receiver Signal Detection Voltage Threshold

Description

Specifies signal detection voltage threshold level, V_{th}. The following encodings are defined:

- SDLV_50MV=7
- SDLV_45MV=6
- SDLV_40MV=5
- SDLV_35MV=4
- SDLV_30MV=3
- SDLV_25MV=2
- SDLV_20MV=1
- SDLV_15MV=0

For the PCIe PIPE PHY, SATA, and SAS protocols.

The signal detect output is high when the receiver peak-to-peak differential voltage (diff p-p) > V_{th} x 4. For example, a setting of 6 translates to peak-to-peak differential voltage of 180mV (4*45mV). The $V_{diff\ p\text{-}p}$ must be > 180mV to turn on the signal detect circuit.

Options

- 0-7
- **3**

Assign To

Pin - RX serial data

XCVR_TX_COMMON_MODE_VOLTAGE

Pin Planner and Assignment Editor Name

Transmitter Common Mode Driver Voltage

Description

Transmitter common-mode driver voltage.

Note: Contact Intel for using this assignment.

Related Information

How to Contact Altera on page 22-46

XCVR_TX_PRE_EMP_1ST_POST_TAP

Pin Planner and Assignment Editor Name

Transmitter Pre-emphasis First Post-Tap

Description

Specifies the first post-tap setting value.

Note: Legal values for this parameter vary with the data pattern and data rate. Refer to the *Cyclone V Device Datasheet* for more information.

Options

0–31

Assign To

Pin - TX serial data

Related Information

Cyclone V Device Datasheet

XCVR_TX_RX_DET_ENABLE

Pin Planner and Assignment Editor Name

Transmitter Receiver Detect Block Enable

Description

Enables or disables the receiver detector circuit at the transmitter.

Options

- TRUE
- **FALSE**

Assign To

Pin - TX serial data

XCVR_TX_RX_DET_MODE

Pin Planner and Assignment Editor Name

Transmitter Receiver Detect Block Mode

Description

Sets the mode for receiver detect block.

Options

0–15

Assign To

Pin - TX serial data

XCVR_TX_VOD

Pin Planner and Assignment Editor Name

Transmitter Differential Output Voltage

Description

Differential output voltage setting. The values are monotonically increasing with the driver main tap current strength.

Options

- 0–63
- **10**

Assign To

Pin - TX serial data

XCVR_TX_VOD_PRE_EMP_CTRL_SRC

Pin Planner and Assignment Editor Name

Transmitter V_{OD} Pre-emphasis Control Source

Description

When set to `DYNAMIC_CTL`, the PCS block controls the V_{OD} and pre-emphasis coefficients for PCI Express. When this assignment is set to `RAM_CTL` the V_{OD} and pre-emphasis are controlled by other assignments, such as `XCVR_TX_PRE_EMP_1ST_POST_TAP`.

Options

- **DYNAMIC_CTL**: for PCI Express
- **RAM_CTL**: for all other protocols

Assign To

Pin - TX serial data

Analog Settings for Stratix V Devices

Analog PCB Settings for Stratix V Devices

This section lists the analog parameters for Stratix V devices whose original values are *place holders* for the values that match your electrical board specification. The default value of an analog parameter is shown in **bold** type. The parameters are listed in alphabetical order.

Related Information

- **PCI Express Card Electromechanical Specification Rev. 2.0**
- **Stratix V Device Datasheet**
- **About the Pin Planner**
- **About the Assignment Editor**
- **Intel Quartus Prime Settings File Manual**

XCVR_GT_IO_PIN_TERMINATION

Pin Planner and Assignment Editor Name

GT Transceiver I/O Pin Termination

Description

Fine tunes the target 100-ohm on-chip termination for the specified transceiver pin. This parameter is only for GT transceivers. It is available for both TX and RX pins.

Options

- 0-15
- **12 (TX)**
- **9 (RX)**

Assign To

Pin - TX & RX serial data

XCVR_IO_PIN_TERMINATION

Pin Planner and Assignment Editor Name

Transceiver I/O Pin Termination

Description

Specifies the intended on-chip termination value for the specified transceiver pin. Use External Resistor if you intend to use off-chip termination.

Options

- 85_Ohms
- **100_Ohms**
- 120_Ohms
- 150_Ohms
- External_Resistor

Assign To

Pin - TX & RX serial data

XCVR_REFCLK_PIN_TERMINATION

Pin Planner and Assignment Editor Name

Transceiver Dedicated Refclk Pin Termination

Description

Specifies the intended termination value for the specified refclk pin. The following 3 settings are available:

- AC_COUPLING: Altera recommends this setting for all transceiver designs. Use it for AC coupled signals. This setting implements on-chip termination and on-chip signal biasing.
- DC_COUPLING_ INTERNAL_100_OHMS: Used this setting when the dedicated transceiver reference clock pins are fed by a DC coupled signal whose V_{cm} meets the device specification. This assignment implements internal on-chip termination but not on-chip signal biasing.
- DC_COUPLING_EXTERNAL_RESISTOR: Use this assignment when the dedicated transceiver reference clock pins are fed by a DC coupled signal. This option does not implement internal on-chip termination or signal biasing. You must implement termination and signal biasing outside of the FPGA. This assignment is recommended for compliance with the *PCI Express Card Electromechanical Specification Rev. 2.0* and the HCSL IO Standard.

Options

- **AC_COUPLING**
- DC_COUPLING_INTERNAL_100_OHMS
- DC_COUPLING_EXTERNAL_RESISTOR

Assign To

Pin - PLL refclk pin

XCVR_RX_BYPASS_EQ_STAGES_234

Pin Planner and Assignment Editor Name

Receiver Equalizer Stage 2, 3, 4 Bypass

Description

Bypass continuous time equalizer stages 2, 3, and 4 to save power. This setting eliminates significant AC gain on the equalizer and is appropriate for chip-to-chip short range communication on a PCB. Assigning

a value to this setting and XCVR_ANALOG_SETTINGS_PROTOCOL results in a Intel Quartus Prime Fitter error as shown in the following example:

Error (21215)

```
Error resolving parameter "pm_rx_sd_bypass_eqz_stages_234" value
on instance "pci_interface_ddf2:u_pci_interface_2|
PCIE_8x8Gb_HARDIP_2:PCIe2_Interface.U_PCIE_CORE|
altpcie_sv_hip_ast_hwtcl:pcie_8x8gb_hardip_2_inst|
altpcie_hip_256_pipen1b:altpcie_hip_256_pipen1b
|sv_xcvr_pipe_native:g_xcvr.sv_xcvr_pipe_native|sv_xcvr_native:
inst_sv_xcvr_native|sv_pma:inst_sv_pma|sv_rx_pma:rx_pma.
sv_rx_pma_inst|rx_pmas[8].rx_pma.rx_pma_buf": Only one QSF
setting for the parameter is allowed.
```

Options

- **All_Stages_Enabled**
- Bypass_Stages

Assign To

Pin - RX serial data

Note: This setting can be used for data rates upto 5 Gbps for backplane applications, and 8 Gbps for chip-to-chip applications.

XCVR_TX_SLEW_RATE_CTRL

Pin Planner and Assignment Editor Name

Transmitter Slew Rate Control

Description

Specifies the slew rate of the output signal. The valid values span from the slowest rate to fastest rate with 1 representing the slowest rate.

Options

1-5

4

Assign To

Pin - TX serial data

XCVR_VCCA_VOLTAGE

Pin Planner and Assignment Editor Name

VCCA_GXB Voltage

Description

Configure the VCCA_GXB voltage for a GXB I/O pin by specifying the intended VCCA_GXB voltage for a GXB I/O pin. If you do not make this assignment the compiler automatically sets the correct VCCA_GXB voltage depending on the configured data rate, as follows:

- Data rate <= 6.5 Gbps: 2_5V
- Data rate > 6.5 Gbps: 3_0V.

Options

- 2_5V
- 3_0V

Assign To

Pin - TX & RX serial data

XCVR_VCCR_VCCT_VOLTAGE

Pin Planner and Assignment Editor Name

VCCR_GXB

VCCT_GXB Voltage

Description

Refer to the *Device Datasheet for Stratix V Devices* for guidance on selecting a value.

Options

- 0_85V
- 1_0V

Assign To

Pin - TX & RX serial data

Related Information

Stratix V Device Datasheet

Analog Settings Having Global or Computed Default Values for Stratix V Devices

The following analog parameters have *global* or *computed* default values. You may want to optimize some of these settings. The default value is shown in **bold** type. For computed analog parameters, the default value listed is for the initial setting, not the recomputed setting. The parameters are listed in alphabetical order.

CDR_BANDWIDTH_PRESET

Pin Planner and Assignment Editor Name

CDR Bandwidth Preset

Description

Specifies the CDR bandwidth preset setting

Options

- **Auto**
- Low
- Medium
- High

Assign To

PLL instance

master_ch_number

Pin Planner and Assignment Editor Name

Parameter (Assignment Editor Only)

Description

For the PHY IP Core for PCI Express (PIPE), specifies the channel number of the channel acting as the master channel for a single transceiver bank or 2 adjacent banks. This setting allows you to override the default master channel assignment for the PCS and PMA. The master channel must use a TX PLL that is in the same transceiver bank. Available for Gen1, Gen2, and Gen3 variants. The following example shows how to override the default master channel for a Stratix V design. You must apply the `pma_bonding_master` override parameter on the Stratix V Transceiver Native PHY instance name. You can use the same procedure for other devices.

Example 20-4: Overriding Default Master Channel

```
Example: set_parameter -name master_ch_number 4 -to
"<design>:inst|altera_xcvr_native_sv:testx8_inst|
sv_xcvr_native:gen_native_inst.xcvr_native_insts[0].
gen_bonded_group_native.xcvr_native_inst".
```

Options

1, 4

Assign To

Include in **.qsf** file

Related Information

Transceiver Configurations in Stratix V Devices
Refer to *Advance [SIC] Channel Placement Guidelines for PIPE Configurations* in this document.

PLL_BANDWIDTH_PRESET

Pin Planner and Assignment Editor Name

PLL Bandwidth Preset

Description

Specifies the PLL bandwidth preset setting

Options

- **Auto**
- Low
- Medium
- High

Assign To

PLL instance

reserved_channel

Pin Planner and Assignment Editor Name

Parameter (Assignment Editor Only)

Description

Allows you to override the default channel placement of x8 variants. For the PHY IP Core for PCI Express (PIPE), you can use this QSF assignment in conjunction with the `master_ch_number` assignment to specify channel 4 as the master channel. Available for Gen1, Gen2, and Gen3 variants.

```
Example: set_parameter -name reserved_channel true
 -to "<design>.pcie_i|altera_xcvr_pipe:<design>
_inst|sv_xcvr_pipe_nr:pipe_nr_inst|sv_xcvr_pipe_native:
transceiver_core"
```

Options

TRUE

Assign To

Include in **.qsf** file

Related Information

Transceiver Configurations in Stratix V Devices
Refer to *Advance [SIC] Channel Placement Guidelines for PIPE Configurations* in this document.

XCVR_ANALOG_SETTINGS_PROTOCOL

Pin Planner and Assignment Editor Name

Transceiver Analog Settings Protocol

Description

Specifies the protocol that a transceiver implements. When you use this setting for fully characterized devices, the Intel Quartus Prime software automatically sets the optimal values for analog settings, including the V_{OD}, pre-emphasis, and slew rate. For devices that are not fully characterized, the Intel Quartus Prime software specifies these settings using preliminary data. If you assign a value to `XCVR_ANALOG_SETTINGS_PROTOCOL`, you cannot assign a value for any settings that this parameter controls. For example, for PCIe, the `XCVR_ANALOG_SETTINGS_PROTOCOL` assigns a value to `XCVR_RX_BYPASS_EQ_STAGES_234`. If you also assign a value to this parameter, a Intel Quartus Prime Fitter error results as shown in the following example:

Example 20-5: Error (21215)

```
Error resolving parameter "pm_rx_sd_bypass_eqz_stages_234" value
on instance "pci_interface_ddf2:u_pci_interface_2|
PCIE_8x8Gb_HARDIP_2:PCIe2_Interface.U_PCIE_CORE|
altpcie_sv_hip_ast_hwtcl:pcie_8x8gb_hardip_2_inst|
altpcie_hip_256_pipen1b:altpcie_hip_256_pipen1b
|sv_xcvr_pipe_native:g_xcvr.sv_xcvr_pipe_native|sv_xcvr_native:
inst_sv_xcvr_native|sv_pma:inst_sv_pma|sv_rx_pma:rx_pma.
sv_rx_pma_inst|rx_pmas[8].rx_pma.rx_pma_buf": Only one QSF
setting for the parameter is allowed.
```

Options

The following protocol values are defined:

- BASIC
- CEI
- CPRI
- INTERLAKEN
- PCIE_GEN1
- PCIE_GEN2
- PCIE_GEN3
- QPI
- SFIS
- SONET
- SRIO
- TENG_1588
- TENG_BASER
- TENG_SDI
- XAUI

Assign To

Pin - TX and RX serial data

XCVR_GT_RX_DC_GAIN

Pin Planner and Assignment Editor Name

Receiver Buffer DC Gain Control

Description

Controls the RX buffer DC gain for GTchannels.

Options

- 0-19
- **8**

Assign To

Pin - RX serial data

XCVR_RX_DC_GAIN

Pin Planner and Assignment Editor Name

Receiver Buffer DC Gain Control

Description

Controls the RX buffer DC gain for GX channels.

Options

1 –4

Assign To

Pin - RX serial data

XCVR_RX_LINEAR_EQUALIZER_CONTROL

Pin Planner and Assignment Editor Name

Receiver Linear Equalizer Control

Description

Static control for the continuous time equalizer in the receiver buffer. The equalizer has 16 settings from 0–15 corresponding to the increasing AC gain.

Options

1 –16

Assign To

Pin - RX serial data

XCVR_GT_RX_COMMON_ MODE_VOLTAGE

Pin Planner and Assignment Editor Name

GT receiver Buffer Common Mode Voltage

Description

Receiver buffer common-mode voltage. This parameter is only for GT transceivers.

Note: Contact Altera for using this assignment.

Related Information

How to Contact Altera on page 22-46

XCVR_GT_RX_CTLE

Pin Planner and Assignment Editor Name

GT Receiver Linear Equalizer Control

Description

Static control for the continuous time equalizer in the receiver buffer. The equalizer has 9 distinct settings from 0-8 corresponding to increasing AC gain. This parameter is only for GT transceivers.

Options

0-8

Assign To

Pin - RX serial data

XCVR_GT_TX_COMMON_ MODE_VOLTAGE

Pin Planner and Assignment Editor Name

GT Transmitter Common Mode Driver Voltage

Description

Transmitter common-mode driver voltage. This parameter is only for GT transceivers.

Note: Contact Altera for using this assignment.

Related Information

How to Contact Altera on page 22-46

XCVR_GT_TX_PRE_EMP_1ST_POST_TAP

Pin Planner and Assignment Editor Name

GT Transmitter Preemphasis First Post-Tap

Description

Specifies the first post-tap setting value. This parameter is only for GT transceivers.

Note: This parameter must be set in conjunction with `XCVR_GT_TX_PRE_EMP_PRE_TAP`. All combinations of these settings are not legal. Refer to the *Stratix V Device Datasheet* for more information.

Options

- 0-31
- **5**

Assign To

Pin - TX serial data

Related Information

Stratix V Device Datasheet

XCVR_GT_TX_PRE_EMP_ INV_PRE_TAP

Pin Planner and Assignment Editor Name

GT Transmitter Pre-emphasis Pre Tap Invert

Description

Inverts the transmitter pre-emphasis pre-tap. This parameter is only for GT transceivers.

Note: This parameter must be set in conjunction with `XCVR_GT_TX_PRE_EMP_PRE_TAP`. All combinations of these settings are not legal. Refer to the *Stratix V Device Datasheet* for more information.

Options

- ON
- **OFF**

Assign To

Pin - TX serial data

Related Information

Stratix V Device Datasheet

XCVR_GT_TX_PRE_EMP_ PRE_TAP

Pin Planner and Assignment Editor Name

GT Transmitter Preemphasis Pre-Tap

Description

Specifies the pre-tap pre-emphasis setting. This parameter is only for GT transceivers.

Options

0-15

Assign To

Pin - TX serial data

XCVR_GT_TX_VOD_MAIN_TAP

Pin Planner and Assignment Editor Name

GT Transmitter Differential Output Voltage

Description

Differential output voltage setting. The values are monotonically increasing with the driver main tap current strength.

Options

- 0-5
- **3**

Assign To

Pin - TX serial data

XCVR_RX_COMMON_MODE_VOLTAGE

Pin Planner and Assignment Editor Name

Receiver Buffer Common Mode Voltage

Description

Receiver buffer common-mode voltage.

Note: Contact Altera for using this assignment.

Related Information

How to Contact Altera on page 22-46

XCVR_RX_ENABLE_LINEAR_EQUALIZER_PCIEMODE

Pin Planner and Assignment Editor Name

Receiver Linear Equalizer Control (PCI Express)

Description

If enabled equalizer gain control is driven by the PCS block for PCI Express. If disabled equalizer gain control is determined by the `XCVR_RX_LINEAR_EQUALIZER_SETTING`

Options

TRUE

FALSE

Assign To

Pin - RX serial data

XCVR_RX_SD_ENABLE

Pin Planner and Assignment Editor Name

Receiver Signal Detection Unit Enable/Disable

Description

Enables or disables the receiver signal detection unit. During normal operation `NORMAL_SD_ON=FALSE`, otherwise `POWER_DOWN_SD=TRUE`.

Used for the PCIe PIPE PHY, SATA and SAS protocols.

Options

FALSE

TRUE

Assign To

Pin - RX serial data

XCVR_RX_SD_OFF

Pin Planner and Assignment Editor Name

Receiver Cycle Count Before Signal Detect Block Declares Loss Of Signal

Description

Number of parallel cycles to wait before the signal detect block declares loss of signal. Only used for the PCIe PIPE PHY, SATA, and SAS protocols.

Options

0–29

Assign To

Pin - RX serial data

XCVR_RX_SD_ON

Pin Planner and Assignment Editor Name

Receiver Cycle Count Before Signal Detect Block Declares Presence Of Signal

Description

Number of parallel cycles to wait before the signal detect block declares presence of signal. Only used for the PCIe PIPE PHY, SATA, and SAS protocols.

Options

0–16

Assign To

Pin - RX serial data

XCVR_RX_SD_THRESHOLD

Pin Planner and Assignment Editor Name

Receiver Signal Detection Voltage Threshold

Description

Specifies signal detection voltage threshold level, V_{th}. The following encodings are defined:

- SDLV_50MV=7
- SDLV_45MV=6
- SDLV_40MV=5
- SDLV_35MV=4
- SDLV_30MV=3
- SDLV_25MV=2
- SDLV_20MV=1
- SDLV_15MV=0

For the PCIe PIPE PHY, SATA, and SAS.

The signal detect output is high when the receiver peak-to-peak differential voltage (diff p-p) > V_{th} x 4. For example, a setting of 6 translates to peak-to-peak differential voltage of 180mV (4*45mV). The $V_{diff\ p\text{-}p}$ must be > 180mV to turn on the signal detect circuit.

Options

- **0**-7

Assign To

Pin - RX serial data

XCVR_TX_COMMON_MODE_VOLTAGE

Pin Planner and Assignment Editor Name

Transmitter Common Mode Driver Voltage

Description

Transmitter common-mode driver voltage.

Note: Contact Intel for using this assignment.

Related Information

How to Contact Altera on page 22-46

XCVR_TX_PRE_EMP_PRE_TAP_USER

Pin Planner and Assignment Editor Name

Transmitter Pre-emphasis Pre-Tap user

Description

Specifies the TX pre-emphasis pretap setting value, including inversion.

Note: This parameter must be set in conjunction with `XCVR_TX_VOD`, `XCVR_TX_PRE_EMP_1ST_POST_TAP`, and `XCVR_TX_PRE_EMP_2ND_POST_TAP`. All combinations of these settings are not legal. Refer to the *Stratix V Device Datasheet* for more information.

Options

0–31

Assign To

Pin - TX serial data

Related Information

- **Solution rd02262013_691**
  This solution provides the mapping of the Transceiver Toolkit pretap settings to the Intel Quartus Prime transceiver QSF assignment.
- **Stratix V Device Datasheet**

XCVR_TX_PRE_EMP_2ND_POST_TAP_USER

Pin Planner and Assignment Editor Name

Transmitter Preemphasis Second Post-Tap user

Description

Specifies the transmitter pre-emphasis second post-tap setting value, including inversion.

Options

0–31

- For option value 1-15, the setting value is -15 to -1.
- For option value 17-31, the setting value is 1 to 15.
- For option value 0/16, the setting value is 0.

Assign To

Pin - TX serial data

Related Information

- **Solution rd02272013_545**
  This solution provides the mapping of the Transceiver Toolkit pretap settings to the Intel Quartus Prime transceiver QSF assignment.

- **Stratix V Device Datasheet**

XCVR_TX_PRE_EMP_1ST_POST_TAP

Pin Planner and Assignment Editor Name

Transmitter pre-emphasis First Post-Tap

Description

Specifies the first post-tap setting value.

Note: This parameter must be set in conjunction with `XCVR_TX_VOD`, `XCVR_TX_PRE_EMP_2ND_POST_TAP`, and `XCVR_TX_PRE_EMP_PRE_TAP`. All combinations of these settings are not legal. Refer to the *Stratix V Device Datasheet* for more information.

Options

0–31

Assign To

Pin - TX serial data

Related Information

Stratix V Device Datasheet

XCVR_TX_PRE_EMP_2ND_POST_TAP

Pin Planner and Assignment Editor Name

Transmitter pre-emphasis Second Post-Tap

Description

Specifies the second post-tap setting value.

Note: This parameter must be set in conjunction with `XCVR_TX_VOD`, `XCVR_TX_PRE_EMP_1ST_POST_TAP`, and `XCVR_TX_PRE_EMP_PRE_TAP`. All combinations of these settings are not legal. Refer to the *Stratix V Device Datasheet* for more information.

Options

0–15

Assign To

Pin - TX serial data

Related Information

- **Solution rd02262013_691**
  This solution provides the mapping of the Transceiver Toolkit pretap settings to the Intel Quartus Prime transceiver QSF assignment.
- **Stratix V Device Datasheet**

XCVR_TX_PRE_EMP_INV_2ND_TAP

Pin Planner and Assignment Editor Name

Transmitter Preemphasis Second Tap Invert

Description

Inverts the transmitter pre-emphasis 2nd post tap.

Options

- TRUE
- **FALSE**

Assign To

Pin - TX serial data

Related Information

Solution rd02262013_691
This solution provides the mapping of the Transceiver Toolkit pretap settings to the Intel Quartus Prime transceiver QSF assignment.

XCVR_TX_PRE_EMP_INV_PRE_TAP

Pin Planner and Assignment Editor Name

Transmitter Preemphasis Pre Tap Invert

Description

Inverts the transmitter pre-emphasis pretap. Specifies the TX pre-emphasis pretap setting value, including inversion.

Options

- TRUE
- **FALSE**

Assign To

Pin - TX serial data

Related Information

Solution rd02262013_691
This solution provides the mapping of the Transceiver Toolkit pretap settings to the Intel Quartus Prime transceiver QSF assignment.

XCVR_TX_PRE_EMP_PRE_TAP

Pin Planner and Assignment Editor Name

Transmitter Pre-emphasis Pre Tap

Description

Specifies the pre-tap pre-emphasis setting.

Note: This parameter must be set in conjunction with `XCVR_TX_VOD`, `XCVR_TX_PRE_EMP_1ST_POST_TAP`, and `XCVR_TX_PRE_EMP_2ND_POST_TAP`. All combinations of these settings are not legal. Refer to the *Stratix V Device Datasheet* for more information.

Options

0–15

Assign To

Pin - TX serial data

Related Information

Stratix V Device Datasheet

XCVR_TX_RX_DET_ENABLE

Pin Planner and Assignment Editor Name

Transmitter Receiver Detect Block Enable

Description

Enables or disables the receiver detector circuit at the transmitter.

Options

- TRUE
- **FALSE**

Assign To

Pin - TX serial data

XCVR_TX_RX_DET_MODE

Pin Planner and Assignment Editor Name

Transmitter Receiver Detect Block Mode

Description

Sets the mode for receiver detect block.

Options

0–15

Assign To

Pin - TX serial data

XCVR_TX_RX_DET_OUTPUT_SEL

Pin Planner and Assignment Editor Name

Transmitter's Receiver Detect Block QPI/PCI Express Control

Description

Determines QPI or PCI Express mode for the Receiver Detect block.

Options

- RX_DET_QPI_ OUT
- **RX_DET_PCIE_ OUT**

Assign To

Pin - TX serial data

XCVR_TX_VOD

Pin Planner and Assignment Editor Name

Transmitter Differential Output Voltage

Description

Differential output voltage setting. The values are monotonically increasing with the driver main tap current strength.

Note: This parameter must be set in conjunction with `XCVR_TX_PRE_EMP_1ST_POST_TAP`, `XCVR_TX_PRE_EMP_2ND_POST_TAP`, and `XCVR_TX_PRE_EMP_PRE_TAP`. All combinations of these settings are not legal. Refer to the *Stratix V Device Datasheet* for more information.

Options

- 0–63
- **50**

Assign To

Pin - TX serial data

Related Information

Stratix V Device Datasheet

XCVR_TX_VOD_PRE_EMP_CTRL_SRC

Pin Planner and Assignment Editor Name

Transmitter V_{OD} Pre-emphasis Control Source

Description

When set to `DYNAMIC_CTL`, the PCS block controls the V_{OD} and pre-emphasis coefficients for PCI Express. When this assignment is set to `RAM_CTL` the V_{OD} and pre-emphasis are controlled by other assignments, such as `XCVR_TX_PRE_EMP_1ST_POST_TAP`.

Options

- **DYNAMIC_CTL**: for PCI Express
- **RAM_CTL**: for all other protocols

Assign To

Pin - TX serial data

Migrating from Stratix IV to Stratix V Devices Overview 21

2020.06.02

UG-01080

Previously, Altera provided the ALTGX megafunction as a general purpose transceiver PHY solution. The current release of the Intel Quartus Prime software includes protocol-specific PHY IP cores that simplify the parameterization process.

The design of these protocol-specific transceiver PHYs is modular and uses standard interfaces. An Avalon-MM interface provides access to control and status registers that record the status of the PCS and PMA modules. Consequently, you no longer must include signals in the top level of your transceiver PHY to determine the status of the serial RX and TX interfaces. Using standard interfaces to access this device-dependent information should ease future migrations to other device families and reduce the overall design complexity. However, to facilitate debugging, you may still choose to include some device-dependent signals in the top level of your design during the initial simulations or even permanently. All protocol-specific PHY IP in Stratix V devices also include embedded controls for post-reset initialization which are available through the Avalon-MM interface.

For Stratix IV devices, the location of the transceiver dynamic reconfiguration logic is design dependent. In general, reconfiguration logic is integrated with the transceiver channels for simple configurations and is separately instantiated for more complex designs that use a large number of channels or instantiate more than one protocol in a single transceiver quad. For Stratix V devices, transceiver dynamic reconfiguration is always performed using the separately instantiated Transceiver Reconfiguration Controller.

Control of loopback modes is also different in Stratix IV and Stratix V devices. For Stratix IV devices, you must select loopback options in the using the MegaWizard Plug-In Manager. For Stratix V devices, you control loopback modes through Avalon-MM registers.

Table 21-1: Controlling Loopback Modes in Stratix IV and Stratix V Devices

Loopback Mode	Stratix IV	Stratix V
Serial loopback	On the **Loopback** tab of the ALTGX MegaWizard Plug-In Manager, Instantiate the `rx_seriallpbken` signal by selecting the **Serial loopback** option. Drive this signal to 1 to put the transceiver in serial loopback mode.	Use the Avalon-MM PHY management interface to set the appropriate bit in the `phy_serial_loopback` register (0x061).

Loopback Mode	Stratix IV	Stratix V
Reverse serial loopback (pre- and post-CDR)	On the **Loopback** tab of the ALTGX MegaWizard Plug-In Manager, select either pre-CDR or post-CDR loopback and regenerate the ALTGX IP core.	Update the appropriate bits of the Transceiver Reconfiguration Controller `tx_rx_word_offset` register to enable the pre- or post-CDR reverse serial loopback mode. Refer to PMA Analog Registers for more information.

Related Information

Transceiver Reconfiguration Controller PMA Analog Control Registers on page 17-14

Differences in Dynamic Reconfiguration for Stratix IV and Stratix V Transceivers

Dynamic reconfiguration interface is completely new in Stratix V devices. You cannot automatically migrate a dynamic reconfiguration solution from Stratix IV to Stratix V devices.

Stratix V devices that include transceivers must use the Altera Transceiver Reconfiguration Controller that contains the offset cancellation logic to compensate for variations due to PVT. Initially, each transceiver channel and each TX PLL has its own parallel, dynamic reconfiguration bus, named reconfig_from_xcvr[45:0] and reconfig_to_xcvr[69:0]. The reconfiguration bus includes Avalon-MM signals to read and write registers and memory and test bus signals. When you instantiate a transceiver PHY in a Stratix V device, the transceiver PHY IP core provides informational messages specifying the number of required reconfiguration interfaces in the message pane.

Example 21-1: Informational Messages for the Transceiver Reconfiguration Interface

```
PHY IP will require 5 reconfiguration interfaces for connection to the
external reconfiguration controller.

Reconfiguration interface offsets 0-3 are connected to the transceiver
channels.

Reconfiguration interface offset 4 is connected to the transmit PLL.
```

Although you must initially create a separate reconfiguration interface for each channel and TX PLL in your design, when the Intel Quartus Prime software compiles your design, it reduces the number of reconfiguration interfaces by merging reconfigurations interfaces. The synthesized design typically includes a reconfiguration interface for three channels. Allowing the Intel Quartus Prime software to merge reconfiguration interfaces gives the Fitter more flexibility in placing transceiver channels.

Stratix IV devices that include transceivers must use the ALTGX_RECONFIG IP Core to implement dynamic reconfiguration. The ALTGX_RECONFIG IP Core always includes the following two serial buses:

- `reconfig_from[<n>16:0]`— this bus connects to all the channels in a single quad. <n> is the number of quads connected to the ALTGX_RECONFIG IP Core.
- `reconfig_togxb[3:0]`—this single bus connects to all transceiver channels.

If you select additional functionality in the MegaWizard Plug-In Manager for the ALTGX_RECONFIG IP Core, the IP core adds signals to support that functionality. For more information about the ALTGX_RECONFIG IP Core, refer to *ALTGX_RECONFIG Megafunction User Guide for Stratix IV Devices* in volume 3 of the *Stratix IV Device Handbook*.

Related Information

ALTGX_RECONFIG Megafunction User Guide for Stratix IV Devices

Differences Between XAUI PHY Parameters for Stratix IV and Stratix V Devices

Table 21-2: Comparison of ALTGX Megafunction and XAUI PHY Parameters

ALTGX Parameter Name (Default Value)	XAUI PHY Parameter Name	Comments
Number of channels	**Number of XAUI interfaces**	In Stratix V devices, this parameter is locked to 1 (for 4 channels). You cannot change it in the current release.
Train receiver clock and data recover (CDR) from pll_inclk (On)	Not available as parameters in the MegaWizard Plug-In Manager interface	Use assignment editor to make these assignment
TX PLL bandwidth mode (Auto)		
RX CDR bandwidth mode (Auto)		

ALTGX Parameter Name (Default Value)	XAUI PHY Parameter Name	Comments
Acceptable PPM threshold between receiver CDR VCO and receiver input reference clock (±1000)	Not available as parameters in the MegaWizard Plug-In Manager interface	Use assignment editor to make these assignments
Analog power (Auto)		
Loopback option (No loopback)		
Enable static equalizer control (Off)		
DC gain (0)		
Receiver common mode voltage (0.82v)		
Use external receiver termination (Off)		
Receiver termination resistance (100 ohms)		
Transmitter buffer power (1.5v)		
Transmitter common mode voltage (0.65v)		
Use external transmitter termination (Off)		
Transmitter termination resistance (100 ohms)		
VOD setting (4)		
Preemphasis 1st post-tap (0)		
Preemphasis pre-tap setting (0)		
Preemphasis second post-tap setting (0)		
Analog controls (Off)		
Enable ADCE (Off)	Not available as parameters in the MegaWizard Plug-In Manager interface	Not available in 10.0
Enable channel and transmitter PLL reconfig (Off)		
Starting channel number (0)	No longer required	Automatically set to 0. The Intel Quartus Prime software handles lane assignments

ALTGX Parameter Name (Default Value)	XAUI PHY Parameter Name	Comments
Enable run length violation checking with run length of (40)	Not available as parameters in the MegaWizard Plug-In Manager interface	Use assignment editor
Enable transmitter bit reversal (Off)		
Word alignment pattern length (10)		

Differences Between XAUI PHY Ports in Stratix IV and Stratix V Devices

This section lists the differences between the top-level signals in Stratix IV GX and Stratix V GX/GS devices.

Table 21-3: Correspondences between XAUI PHY Stratix IV GX and Stratix V Device Signals

Stratix IV GX Devices[21]		Stratix V Devices	
Signal Name	Width	Signal Name	Width
Reference Clocks and Resets			
`pll_inclk`	1	`refclk`	1
`rx_cruclk`	[<*n*> -1:0]	Not available	—
`coreclkout`	1	`xgmii_rx_clk`	1
`rx_coreclk`	[<*n*> -1:0]	Not available	—
`tx_coreclk`	[<*n*> -1:0]	`xgmii_tx_clk`	1
Not available	—	`rx_pma_ready`	1
Not available	—	`tx_pma_ready`	1
Data Ports			
`rx_datain`	[<*n*> -1:0]	`xaui_rx_serial`	[3:0]
`tx_datain`	[16<*n*> -1:0]	`xgmii_tx_dc`	[63:0]
`rx_dataout`	[16<*n*> -1:0]	`xgmii_rx_dc`	[63:0]
`tx_dataout`	[<*n*> -1:0]	`xaui_tx_serial`	[3:0]
Optional TX and RX Status Ports			
`gxb_powerdown`	[<*n*>/4 -1:0]	Not available, however you can access them through the Avalon-MM PHY management interface.	—
`pll_locked`	[<*n*> -1:0]	Not available	—

(21) <*n*> = the number of lanes. <*d*> = the total deserialization factor from the pin to the FPGA fabric.

Stratix IV GX Devices[21]		Stratix V Devices	
Signal Name	Width	Signal Name	Width
rx_locktorefclk	[<*n*> -1:0]	Not available	—
rx_locktodata	[<*n*> -1:0]	Not available	—
rx_pll_locked	[<*n*>/4 -1:0]	Not available	—
rx_freqlocked	[<*n*>/4 -1:0]	Not available	—
rx_phase_comp_fifo_error	[<*n*>/4 -1:0]	Not available	—
tx_phase_comp_fifo_error	[<*n*>/4 -1:0]	Not available	—
cal_blk_powerdown	—	Not available	—
rx_syncstatus	[2<*n*> -1:0]	rx_syncstatus	[<*n*> *2 -1:0]
rx_patterndetect	[2<*n*> -1:0]	Not available	—
rx_invpolarity	[2<*n*> -1:0]	Not available	—
rx_ctrldetect	[2<*n*> -1:0]	Not available	—
rx_errdetect	[2<*n*> -1:0]	rx_errdetect	[<*n*> *2 -1:0]
rx_disperr	[2<*n*> -1:0]	rx_disperr	[<*n*> *2 -1:0]
tx_invpolarity	[2<*n*> -1:0]	Not available	—
rx_runningdisp	[2<*n*> -1:0]	Not available	—
rx_rmfifofull	[2<*n*> -1:0]	Not available	—
rx_rmfifoempty	[2<*n*> -1:0]	Not available	—
rx_rmfifodatainserted	[2<*n*> -1:0]	Not available	—
rx_rmfifodatadeleted	[2<*n*> -1:0]	Not available	—
Transceiver Reconfiguration			
cal_blk_clk	1	These signals are included in the reconfig_to_xcvr bus.	—
reconfig_clk	1		—
reconfig_togxb	[3:0]	reconfig_to_xcvr	Variable
reconfig_fromgxb	[16:0]	reconfig_from_xcvr	Variable
Avalon MM Management Interface			

(21) <*n*> = the number of lanes. <*d*> = the total deserialization factor from the pin to the FPGA fabric.

Stratix IV GX Devices[21]		Stratix V Devices	
Signal Name	Width	Signal Name	Width
Not Available	—	phy_mgmt_clk_rst	1
		phy_mgmt_clk	1
		phy_mgmt_address	[8:0]
		phy_mgmt_read	1
		phy_mgmt_readdata	[31:0]
		phy_mgmt_write	1
		phy_mgmt_writedat	[31:0]

Differences Between PHY IP Core for PCIe PHY (PIPE) Parameters in Stratix IV and Stratix V Devices

This section lists the PHY IP Core for PCI Express PHY (PIPE) parameters and the corresponding ALTGX megafunction parameters.

Table 21-4: Comparison of ALTGX Megafunction and PHY IP Core for PCI Express PHY (PIPE) Parameters

ALTGX Parameter Name (Default Value)	CI Express PHY (PIPE) Parameter Name	Comments
Number of channels	**Number of Lanes**	—
Channel width	**Deserialization factor**	—
Subprotocol	**Protocol Version**	—
Input clock frequency	**PLL reference clock frequency**	—
Starting Channel Number	—	Automatically set to 0. Intel Quartus Prime software handles lane assignments.
Enable low latency sync	pipe_low_latency_syncronous_mode	—
Enable RLV with run length of	pipe_run_length_violation_checking	Always on
Enable electrical idle inference functionality	**Enable electrical idle inferencing**	—

[21] <*n*> = the number of lanes. <*d*> = the total deserialization factor from the pin to the FPGA fabric.

ALTGX Parameter Name (Default Value)	CI Express PHY (PIPE) Parameter Name	Comments
—	phy_mgmt_clk_in_mhz	For embedded reset controller to calculate delays
Train receiver CDR from pll_inclk (false)	Not available in MegaWizard Interface	Use assignment editor to make these assignments
TX PLL bandwidth mode (Auto)		
RX CDR bandwidth mode (Auto)		
Acceptable PPM threshold (±300)		
Analog Power(VCCA_L/R) (Auto)		
Reverse loopback option (No loopback)		
Enable static equalizer control (false)		
DC gain (1)		
RX Vcm (0.82)		
Force signal detection (Off)		
Signal Detect threshold (4)		
Use external receiver termination (Off)		
RX term (100)		
Transmitter buffer power(VCCH) (1.5)		
TX Vcm (0.65)		
Use external transmitter termination (Off)		
TX Rterm (100)		
VCO control setting (5)		
Pre-emphasis 1st post tap (18)	Not available in MegaWizard Interface	Use assignment editor to make these assignments
Pre-tap (0)		
2nd post tap (0)		
DPRIO - VOD, Pre-em, Eq and EyeQ (Off)		
DPRIO - Channel and TX PLL Reconfig (Off)		

Differences Between PHY IP Core for PCIe PHY (PIPE) Ports for Stratix IV and Stratix V Devices

This section lists the differences between the top-level signals in Stratix IV GX and Stratix V GX/GS devices. PIPE standard ports remain, but are now prefixed with pipe_. Clocking options are simplified to match the PIPE 2.0 specification.

Table 21-5: PCIe PHY (PIPE) Correspondence between Stratix IV GX Device and Stratix V Device Signals

Stratix IV GX Device Signal Name[22]	Stratix V GX Device Signal Name	Width
Reference Clocks and Resets		
pll_inclk	pll_ref_clk	1
rx_cruclk	Not available	[<*n*> -1:0]
tx_coreclk	Not available	[<*n*> -1:0]
rx_coreclk	Not available	[<*n*> -1:0]
tx_clkout/coreclkout	pipe_pclk	1
pll_powerdown	These signals are now available as control and status registers. Refer to the "Avalon-MM PHY Management Interface and PCI Express PHY (PIPE) IP Core Registers".	1
cal_blk_powerdown		1
Not available	tx_ready (reset control status)	1
Not available	rx_ready (reset curl status)	1
PIPE interface Ports		
tx_datain	pipe_txdata	[<*n*><*d*>-1:0]
tx_ctrlenable	pipe_txdatak	[(<*d*>/8)*<*n*>-1:0]
tx_detectrxloop	pipe_txdetectrx_loopback	[<*n*> -1:0]
tx_forcedispcompliance	pipe_txcompliance	[<*n*> -1:0]
tx_forceelecidle	pipe_txelecidle	[<*n*> -1:0]
txswing	pipe_txswing	[<*n*> -1:0]
tx_pipedeemph[0]	pipe_txdeemph	[<*n*> -1:0]
tx_pipemargin[2:0]	pipe_txmargin	[3<*n*>-1:0]
rateswitch[0]	pipe_rate[1:0]	[<*n*>-1:0]
powerdn	pipe_powerdown	[2<*n*>-1:0]
rx_elecidleinfersel	pipe_eidleinfersel	[3<*n*>-1:0]

[22] <*n*> = the number of lanes. <*d*> = the total deserialization factor from the pin to the FPGA fabric.

Stratix IV GX Device Signal Name[22]	Stratix V GX Device Signal Name	Width
rx_dataout	pipe_rxdata	[<*n*>-*<*d*>-1:0]
rx_ctrldetect	pipe_rxdatak	[(<*d*>/8)*<*n*>-1:0]
pipedatavalid	pipe_rxvalid	[<*n*>-1:0]
pipe8b10binvpolarity	pipe_rxpolarity	[<*n*>-1:0]
pipeelecidle	pipe_rxelecidle	[<*n*>-1:0]
pipephydonestatus	pipe_phystatus	[<*n*>-1:0]
pipestatus	pipe_rxstatus	[3<*n*>-1:0]
Non-PIPE Ports		
rx_pll_locked	rx_is_lockedtoref	[<*n*>--1:0]
rx_freqlocked	rx_is_lockedtodata	[<*n*>--1:0]
pll_locked	pll_locked	1
rx_syncstatus	rx_syncstatus (also management interface)	[(<*d*>/8)*<*n*>-1:0]
rx_locktodata	These signals are now available as control and status registers. Refer to the "Register Interface and Register Descriptions".	[<*n*>-1:0]
rx_locktorefclk		[<*n*>-1:0]
tx_invpolarity		[<*n*>-1:0]
rx_errdetect		[(<*d*>/8)*<*n*>-1:0]
rx_disperr		[(<*d*>/8)*<*n*>-1:0]
rx_patterndetect		[(<*d*>/8)*<*n*>-1:0]
tx_phase_comp_fifo_error		[<*n*>-1:0]
rx_phase_comp_fifo_error		[<*n*>-1:0]
rx_signaldetect		[<*n*>-1:0]
rx_rlv		[<*n*>-1:0]
rx_datain	rx_serial_data	[<*n*>-1:0]
tx_dataout	tx_serial_data	[<*n*>-1:0]
Reconfiguration		
cal_blk_clk	These signals are included in the reconfig_to_xcvr bus	1
reconfig_clk		1
fixedclk		1
reconfig_togxb	reconfig_to_xcvr	Variable

(22) <*n*> = the number of lanes. <*d*> = the total deserialization factor from the pin to the FPGA fabric.

Stratix IV GX Device Signal Name[22]	Stratix V GX Device Signal Name	Width
reconfig_fromgxb	reconfig_from_xcvr	Variable
Avalon MM Management Interface		
Not available	phy_mgmt_clk_reset	1
	phy_mgmt_clk	1
	phy_mgmt_address	[8:0]
	phy_mgmt_read	1
	phy_mgmt_readdata	[31:0]
	phy_mgmt_write	1
	phy_mgmt_writedata	[31:0]

Related Information

PHY for PCIe (PIPE) Register Interface and Register Descriptions on page 9-16

Differences Between Custom PHY Parameters for Stratix IV and Stratix V Devices

This section lists the Custom PHY parameters and the corresponding ALTGX megafunction parameters.

Table 21-6: Comparison of ALTGX Megafunction and Custom PHY Parameters

ALTGX Parameter Name (Default Value)	Custom PHY Parameter Name
General	
Not available	**Device family**
	Transceiver protocol
	Mode of operation
	Enable bonding
What is the number of channels?	**Number of lanes**
Which subprotocol will you be using? (×4, ×8)	Not available
What is the channel width?	**Serialization factor**
What is the effective data rate?	**Data rate**
What is the input clock frequency?	**Input clock frequency**
tx/rx_8b_10b_mode	**Enable 8B/10B encoder/decoder**

[22] <*n*> = the number of lanes. <*d*> = the total deserialization factor from the pin to the FPGA fabric.

ALTGX Parameter Name (Default Value)	Custom PHY Parameter Name
Not available	**Enable manual disparity control**
	Create optional 8B10B status ports
What is the deserializer block width? **Single** **Double**	**Deserializer block width:** [23] **Auto** **Single** **Double**
Additional Options	
Not available	**Enable TX Bitslip**
	Create rx_coreclkin port
	Create tx_coreclkin port
	Create rx_recovered_clk port
	Create optional ports
	Avalon data interfaces
	Force manual reset control
Protocol Settings-Word Aligner	**Word Aligner**
Use manual word alignment mode **Use manual bitslipping mode** **Use the built-in 'synchronization state machine'**	**Word alignment mode**
Enable run length violation checking with a run length of	**Run length**
What is the word alignment pattern	**Word alignment pattern**
What is the word alignment pattern length	**Word aligner pattern length**
Protocol Settings-Rate match/Byte order	**Rate Match**
What is the 20-bit rate match pattern1 **(usually used for +ve disparity pattern)**	**Rate match insertion/deletion +ve disparity pattern**
What is the 20-bit rate match pattern1 **(usually used for -ve disparity pattern)**	**Rate match insertion/deletion -ve disparity pattern**
Protocol Settings—Rate match/Byte order	**Byte Order**

[23] This parameter is on the **Datapath** tab.

ALTGX Parameter Name (Default Value)	Custom PHY Parameter Name
What is the byte ordering pattern	**Byte ordering pattern**

Differences Between Custom PHY Ports in Stratix IV and Stratix V Devices

This section lists the differences between the top-level signals in Stratix IV GX and Stratix V GX/GS devices.

Table 21-7: Custom PHY Correspondences between Stratix IV GX Device and Stratix V Device Signals

ALTGX[24]	Custom PHY	Width
Avalon-MM Management Interface		
Not available	phy_mgmt_clk_reset	1
	phy_mgmt_clk	1
	phy_mgmt_address	8
	phy_mgmt_read	1
	phy_mgmt_readdata	32
	phy_mgmt_write	1
	phy_mgmt_writedata	32
Clocks		
cal_blk_clk	These signals are included in the reconfig_to_xcvr bus	—
reconfig_clk		—
pll_inclk	pll_ref_clk	[<*p*>-1:0]
rx_coreclk	rx_coreclkin	—
tx_coreclk	tx_coreclkin	—
Avalon-ST TX Interface		
tx_datain	tx_parallel_data	[<*d*><*n*>-1:0]
tx_ctrlenable	tx_datak	[<*d*><*n*>-1:0]
rx_ctrldetect	rx_datak	[<*d*><*n*>-1:0]
Avalon-ST RX Interface		
rx_dataout	rx_parallel_data	[<*d*><*n*>-1:0]
rx_runningdisp	rx_runningdisp	[<*d*/8><*n*>-1:0]
rx_enabyteord	rx_enabyteord	[<*n*>-1:0]

[24] <*n*> = the number of lanes. <*d*> = the total deserialization factor from the pin to the FPGA fabric.

ALTGX[24]	Custom PHY	Width
High Speed Serial I/O		
rx_datain	rx_serial_data	[<*n*>-1:0]
tx_dataout	tx_serial_data	[<*n*>-1:0]
rx_freqlocked	rx_is_lockedtodata	[<*n*>-1:0]
Transceiver Control and Status Signals		
gxb_powerdown	phy_mgmt_clk_reset	—
rx_dataoutfull	—	—
tx_dataoutfull	—	—
rx_pll_locked	There are both pll_locked and rx_pll_clocked in Stratix IV. Stratix V only has pll_locked.	—
rx_clkout	These signals are now available as control and status registers. Refer to Register Descriptions.	—
rx_phase_comp_fifo_error		—
rx_seriallpbken		—
tx_phase_comp_fifo_error		—
tx_invpolarity		—
Transceiver Reconfiguration		
reconfig_togxb[3:0]	reconfig_to_xcvr	Variable
reconfig_fromgxb[16:0]	reconfig_from_xcvr	Variable

Related Information

Register Interface and Register Descriptions on page 10-27

[24] <*n*> = the number of lanes. <*d*> = the total deserialization factor from the pin to the FPGA fabric.

Additional Information for the Transceiver PHY IP Core 22

2020.06.02

UG-01080 Subscribe Send Feedback

This section provides the revision history for the chapters in this user guide.

Document Version	Changes
2020.06.02	Updated *Transceiver Reconfiguration Controller DFE Registers* to state that DFE is supported by Arria V GZ and Stratix V devices.

Revision History for Previous Releases of the Transceiver PHY IP Core

This section provides the revision history for all versions of this user guide.

Chapter	Document Version	Changes Made
Stratix V Transceiver Native PHY IP Core	3.2	Corrected the definition for `tx_10g_control [9<n>-1:0]` in the Basic mode, 66-bit word width in Table: *10G PCS Interface Signals*.
1G/2.5G/5G/10G Multi-rate Ethernet PHY Intel FPGA IP Core	3.2	Made the following changes: • Updated *Timing Constraints*. • Removed a footnote from the description of the *Register Map*. • Updated Table: *Register Map*.
10GBASE-R PHY IP Core	3.1	Corrected Figure: *Stratix V Clock Generation and Distribution*.

Chapter	Document Version	Changes Made
Backplane Ethernet 10GBASE-KR PHY IP Core	3.1	• Added the section *MII Interface Signals* and also listed them in Figure: *10GBASE-KR Top-Level Signals*. • Clarified the descriptions for `tx_pcfifo_error_1g` and `rx_pcfifo_error_1g` signals. • Corrected the default value for `ber_time_k_frames` register. • Added the default value for `ber_time_m_frames` register. • Changed the 1GbE parameter name from **Enable SGMII bridge logic** to **Expose MII Interface**.
1G/10Gbps Ethernet PHY IP Core	3.1	• Added the section *MII Interface Signals* and also listed them in Figure: *1G/10GbE PHY Top-Level Signals*. • Added 10G PCS Pattern Generators support for 1G/10G Ethernet PHY IP core. • Changed the 1GbE parameter name from **Enable SGMII bridge logic** to **Expose MII Interface**.
Interlaken PHY IP Core	3.1	Clarified the `rx_parallel_data<n> [65]` signal description.
Custom PHY IP Core	3.1	Removed the `reset_control` (write) register from Table: *Reset Control Registers–Automatic Reset Controller*.
Stratix V Transceiver Native PHY IP Core	3.1	Corrected the definition for `rx_10g_control [3]` in Table: *10G PCS Interface Signals*.
Arria V GZ Transceiver Native PHY IP Core	3.1	Corrected the definition for `rx_10g_control [3]` in Table: *10G PCS Interface Signals*.
Cyclone V Transceiver Native PHY IP Core Overview	3.1	Corrected the maximum data rate in Table: *PMA Options*.
Analog Parameters Set Using QSF Assignments	3.1	Clarified the analog setting options for `XCVR_TX_PRE_EMP_2ND_POST_TAP_USER`.
Backplane Ethernet 10GBASE-KR PHY IP Core	3.0	• Corrected the default values for the 10GBASE-KR Link Training Parameters **VODMINRULE**, **VPOSTRULE**, **INITMAINVAL**, **INITPOSTVAL**, and **INITPREVAL** in the *Table: Link Training Settings*

Chapter	Document Version	Changes Made
1G/10Gbps Ethernet PHY IP Core	3.0	• Added the status signal `led_panel_link` to the *Table: SGMII and GMII Signals*. • Clarified the signal description for `led_link`, `rx_block_lock`, and `COPPER_LINK_STATUS`. • Corrected the register list for address 0x94/95 (SGMII mode) in *Table: GMII PCS Registers*.
Stratix V Transceiver Native PHY IP Core	3.0	• Corrected the Slew Rate Settings.
Arria V GZ Transceiver Native PHY IP Core	3.0	• Corrected the Slew Rate Settings.
1G/2.5G/5G/10G Multi-rate Ethernet PHY IP Core	2.9	Added this chapter.
Stratix V Transceiver Native PHY IP Core	2.9	Made the following changes: • Added the PRBS Verifier Inversion Offset to the *Table: Offsets for the Standard PCS Pattern Generator and Verifier.* • Added the slew rate settings for SATA sub-protocol.
Arria V GZ Transceiver Native PHY IP Core	2.9	Made the following changes: • Added the PRBS Verifier Inversion Offset to the *Table: Offsets for the Standard PCS Pattern Generator and Verifier.* • Added the slew rate settings for SATA sub-protocol.
Analog Parameters Set Using QSF Assignments	2.9	Corrected the `XCVR_RX_DC_GAIN` default setting for Stratix V devices.
10GBASE-R PHY IP Core	2.8	Corrected the Product ID for 10GBASE-R PHY IP Core in *Table: 10GBASE-R Release Information*.
10GBASE-KR PHY IP Core	2.8	Removed Early Access FEC Option.
1G/10Gbps Ethernet PHY IP Core	2.8	Added the device ability and partner ability registers for SGMII mode to *Table: GMII PCS Registers*.

Chapter	Document Version	Changes Made
PHY IP Core for PCI Express (PIPE)	2.8	Made the following changes: • Updated *Table: Device Family Support* for PHY IP Core for PCI Express with the support information for Cyclone V devices. • Clarified the description of `rx_polarity` signal in *Table: Avalon-ST TX Inputs*.
Custom PHY IP Core	2.8	Made the following changes: • Clarified the description for `tx_dispval` signal in *Table: Avalon-ST TX Interface Signals*. • Corrected the pattern length for PMA-PCS Interface Width (20-bit manual alignment) in *Table: More Information about Word Aligner Functions*.
Deterministic Latency PHY IP Core	2.8	Made the following changes: • Added the latency values for Arria V GZ in the *Table: PMA Datapath Total Latency*. • Clarified the information in *Table: Signal Definitions for `tx_parallel_data` with and without 8B/10B Encoding*. • Clarified the information in *Table: Signal Definitions for `rx_parallel_data` with and without 8B/10B Encoding*.
Stratix V Transceiver Native PHY IP Core	2.8	Made the following changes: • Clarified the information in *Table: Signal Definitions for `tx_parallel_data` with and without 8B/10B Encoding*. • Clarified the information in *Table: Signal Definitions for `rx_parallel_data` with and without 8B/10B Encoding*. • Corrected the description of Offset `0x135[3]` in *Table: Pattern Generator Registers*.
Arria V Transceiver Native PHY IP Core	2.8	Made the following changes: • Clarified the information in *Table: Signal Definitions for `tx_parallel_data` with and without 8B/10B Encoding*. • Clarified the information in *Table: Signal Definitions for `rx_parallel_data` with and without 8B/10B Encoding*. • Added a new topic called *Slew Rate Setting*.

Chapter	Document Version	Changes Made
Arria V GZ Transceiver Native PHY IP Core	2.8	Made the following changes: • Clarified the information in *Table: Signal Definitions for `tx_parallel_data` with and without 8B/10B Encoding*. • Clarified the information in *Table: Signal Definitions for `rx_parallel_data` with and without 8B/10B Encoding*. • Corrected the description of Offset `0x135[3]` in *Table: Pattern Generator Registers*. • Added a new topic called *Slew Rate Setting*.
Cyclone V Transceiver Native PHY IP Core	2.8	Made the following changes: • Clarified the information in *Table: Signal Definitions for `tx_parallel_data` with and without 8B/10B Encoding*. • Clarified the information in *Table: Signal Definitions for `rx_parallel_data` with and without 8B/10B Encoding*. • Added a new topic called *Slew Rate Setting*.
Transceiver PHY Reset Controller IP Core	2.8	Added the *Usage Examples for `pll_select`* section.
Analog Parameters Set Using QSF Assignments	2.8	Added the default value for `XCVR_TX_VOD` (Analog settings for Arria V Devices).
Getting Started Overview	2.7	Updated the chapter to indicate new IP instantiation flow using the IP Catalog.
10GBASE-R PHY IP Core	2.7	Made the following changes: • Updated the chapter to indicate new IP instantiation flow using the IP Catalog. • Changed the device family support to final for this IP core in *Table 3-4 Device Family Support*. • Changed the description of **Starting Channel Number** parameter in *General Option Parameters* section. • Changed the description of `phy_mgmt_clk_reset signal` in *Table 3-15: Avalon-MM PHY Management Interface*. • Changed the TX and RX Latency numbers for Stratix V devices in *Table 3-2: Latency for TX and RX PCS and PMA in Stratix V Devices*.

Chapter	Document Version	Changes Made
10GBASE-KR PHY IP Core	2.7	Made the following changes: • Updated the chapter to indicate new IP instantiation flow using the IP Catalog. • Changed the device family support to final for this IP core in *Table 4-2: Device Family Support*. • Added resource utilization numbers when FEC is used in *Table 4-3: 10GBASE KR PHY Performance and Resource Utilization*. • Changed the description of `tx_invpolarity` register and register address `0x22` in *PMA Registers* section. • Changed the descriptions of `main_rc`, `post_rc`, and `pre_rc` signals with example mappings in *Dynamic Reconfiguration Interface Signals* section.
1G/10Gbps Ethernet PHY IP Core	2.7	Made the following changes: • Updated the chapter to indicate new IP instantiation flow using the IP Catalog. • Changed the device family support to final for this IP core in *Table 5-2: Device Family Support*. • Removed erroneous references to 10GBBASE-KR PHY IP Core from this chapter. • Updated the description of *rx_clkslip* signal in *1G/10GbE Control and Status Interfaces*.
XAUI PHY IP Core	2.7	Made the following changes: • Updated the description of `rx_digitalreset` and `tx_digitalreset` signals in *Table 6-10: Optional Clock and Reset Signals*. • Updated *Figure 6-4: XAUI Top-Level Signals - Soft PCS and PMA*. • Added the description of `xgmii_tx_clk` and `xgmii_rx_clk` in *Table 6-10: Optional Clock and Reset Signals*.
Interlaken PHY IP Core	2.7	Made the following changes: • Updated the chapter to indicate new IP instantiation flow using the IP Catalog. • Changed the device family support to final for this IP core in the *Table 7-1 Device Family Support*. • Updated the descriptions of `rx_dataout_bp<n>` and `tx_user_clkout` signals in *Table 7-5: Avalon-ST RX Signals*.

Chapter	Document Version	Changes Made
PHY IP Core for PCI Express	2.7	Made the following changes: • Updated the chapter to indicate new IP instantiation flow using the IP Catalog. • Changed the device family support to final for this IP core in *Table 8-1: Device Family Support*. • Updated *Table 8-4: Preset Mappings to TX De-Emphasis.*
Custom PHY IP Core	2.7	Made the following changes: • Updated the chapter to indicate new IP instantiation flow using the IP Catalog. • Changed the device family support to final for this IP core in *Table 9-1: Device Family Support*. • Changed the description of `tx_bitslipboundaryselect` signal in *Optional Status Interfaces* section. • Changed the word alignment pattern for Ethernet in *Table 9-11: Presets for Ethernet Protocol.* • Added a note related to compile warning (12020) in the description of `tx_analogreset` signal. • Corrected byte ordering pattern length for configuration 4 in *Byte Order Parameters* section.
Low Latency PHY IP Core	2.7	Made the following changes: • Updated the chapter to indicate new IP instantiation flow using the IP Catalog. • Changed the device family support fo final for this IP core in *Table 10-1: Device Family Support*.

Deterministic Latency PHY IP Core	2.7	Made the following changes:
		• Updated the chapter to indicate new IP instantiation flow using the IP Catalog.
		• Changed the device family support fo final for this IP core in *Table 11-4: Device Family Support*.
		• Updated Table: PMA Datapath Total Latency with actual hardware delays.
		• Removed description and figure related to using PLL feedback method to align the TX core clock with the RX core clock.
		• In *Deterministic Latency PHY Delay Estimation Logic* section:
		  • Added description of `rx_std_bitslipboundaryselect` signal.
		  • Added footnotes related to latency calculations in Table 11-2 and Table 11-3.

Chapter	Document Version	Changes Made
Stratix V Transceiver Native PHY IP Core	2.7	Made the following changes: • Updated the chapter to indicate new IP instantiation flow using the IP Catalog. • Changed the device family support fo final for this IP core in *Table 12-1: Device Family Support*. • Added a new topic called *Slew Rate Settings*. • Changed **Enable rx_pma_bitslip port** parameter to **Enable rx_clkslip port** in *Table 12-7: RX PMA Parameters*. • Changed the description of `rx_clkslip` port in *Table 12-38: Native PHY Common Interfaces*. • Changed the range of **PPM detector threshold** parameter to +/- 1000 in *Table 12-6: RX PMA Parameters*. • Updated the description of `rx_10g_blk_sh_err` signal in *10G PCS Interface* section. • Updated the description of **Enable rx_std_signaldetect port** parameter with details for implementing SATA/SAS applications. • Updated the *10G PCS Interface* section to indicate that 10G PCS interface signals are used when phase compensation FIFO is in FIFO mode. • Added a note to Table: *Status Flag Mappings for Simplified Native PHY Interface* regarding EDB and PAD characters.
Arria V Transceiver Native PHY IP Core	2.7	Made the following changes: • Updated the chapter to indicate new IP instantiation flow using the IP Catalog. • Changed the device family support to final for this IP core in *Table 13-1: Device Family Support*. • Added note related to PMA Direct mode support for Arria V devices. • Changed the range of **PPM detector threshold** parameter to +/- 1000 in *Table 13-6: RX PMA Parameters*. • Added a note related to compile warning (12020) in the description of `tx_analogreset` signal. • Added Table: *Status Flap Mappings for Simplified Native PHY Interface* in *Rate Match FIFO Parameters* section.

Chapter	Document Version	Changes Made
Arria V GZ Transceiver Native PHY IP Core	2.7	Made the following changes: • Updated the chapter to indicate new IP instantiation flow using the IP Catalog. • Changed the device family support to final for this IP core in *Table 14-1: Device Family Support*. • Changed the range of **PPM detector threshold** parameter to +/- 1000 in *Table 14-6: RX PMA Parameters*. • Updated the description of `rx_10g_blk_sh_err` signal in *10G PCS Interfaces* section. • Updated the description of **Enable rx_std_signaldetect port** parameter with details for implementing SATA/SAS applications.
Cyclone V Transceiver Native PHY IP Core	2.7	Made the following changes: • Updated the chapter to indicate new IP instantiation flow using the IP Catalog. • Changed the device family support to final for this IP core in *Table 15-1: Device Family Support*. • Changed the range of **PPM detector threshold** parameter to +/- 1000 in *Table 15-6: RX PMA Parameters*. • Added Table: *Status Flag Mappings for Simplified Native PHY Interface* under *Rate Match FIFO Parameters* section.
Transceiver Reconfiguration Controller IP Core	2.7	Made the following changes: • Updated the chapter to indicate new IP instantiation flow using the IP Catalog. • Added an exception for Native PHY IP in the *Loopback Modes* section. • Changed the introductory sentence of *MIF Reconfiguration Manage Avalon-MM Master Interface* section to better describe MIF Reconfiguration Management interface. • Updated *Table 16-18: ATX PLL Tuning Offsets and Values.* • Updated the description of Streamer Offset Register in *Table 16-24: Streamer Module Registers.* • Created a new topic *Sharing Reconfiguration Interface for Multi-Channel Transceiver Designs.* • Modified the description of `cal_busy_in` signal in *Table 16-5: MIF Reconfiguration Management Avalon-MM Master Interface.* • Enhanced the verbiage of *Register Based Read* and *Register Based Write* sections. • Added a new section called *EyeQ Usage Example*.

Chapter	Document Version	Changes Made
Transceiver PHY Reset Controller IP Core	2.7	Made the following changes: • Updated the chapter to indicate new IP instantiation flow using the IP Catalog. • Changed the device family support to final in *Table 17-1: Device Family Support*. • Added description for `rx_analogreset` signal in *Transceiver PHY Reset Controller Interfaces* section.
Transceiver PLL IP Core for Stratix V, Arria V, and Arria V GZ Devices	2.7	Made the following changes: • Updated the chapter to indicate new IP instantiation flow using the IP Catalog. • Added a note to indicate that it is recommended to use fractional PLL in fractional mode as a TX PLL or for PLL cascading.
Analog Parameters Set Using QSF Assignments	2.7	Made the following changes: • Removed references to SATA protocol from `XCVR_ANALOG_PROTOCOL QSF` assignment. • Added a note related to data rate restriction for `XCVR_RX_BYPASS_EQ_STAGES_234` assignment.

Chapter	Document Version	Changes Made
10GBASE-R PHY IP Core	2.6	Updated the descriptions of `tx_cal_busy` and `rx_cal_busy` interface signals.

Chapter	Document Version	Changes Made
Backplane Ethernet 10GBASE-KR PHY	2.6	Made the following changes: • Corrected an error in the description of `pcs_mode_rc[5:0]` in *Table 4-17: Dynamic Reconfiguration Interface Signals*. Added back the option for GigE data mode and 10G data mode with FEC. • Updated the descriptions of `tx_cal_busy` and `rx_cal_busy` interface signals. • Updated the descriptions of `tm_in_trigger[3:0]` and `tm_out_trigger [3:0]` signals in *Table 4-14: Control and Status Signals*. • Updated the descriptions of `xgmii_tx_clk` and `xgmii_rx_clk` signals in *Table 4-11: XGMII and GMII Signals*. • Updated the description of `en_lcl_rxeq` and `rxeq_done` signals in *Table 4-17: Dynamic Reconfiguration Interface Signals*. • Added a note about performing read-modify-writes for all registers in *10GBASE-KR PHY Register Definitions* section. • Added a clarification about reset sequencer in the *10GBASE-KR PHY Clock and Reset Interfaces* section on page 4-18. • Updated `tx_clkout_1g`, `rx_clkout_1g`, `tx_coreclkin_1g`, and `rx_coreclkin_1g` connections in *Figure 4-11: Clocks for Standard and 10G PCS and TX PLLs*.

Chapter	Document Version	Changes Made
1G/10GbE Ethernet PHY IP Core	2.6	Made the following changes: • Corrected an error in the description of `pcs_mode_rc[5:0]` in *Table 5-15: Dynamic Reconfiguration Interface Signals*. Added back the option for GigE data mode and 10G data mode with FEC. • Updated the descriptions of `tx_cal_busy` and `rx_cal_busy` interface signals. • Updated the descriptions of `tm_in_trigger[3:0]` and `tm_out_trigger [3:0]` signals in *Table 5-10: Control and Status Signals*. • Updated the descriptions of `xgmii_tx_clk` and `xgmii_rx_clk` signals in *Table 5-7: SGMII and GMII signals*. • Updated the description of `en_lcl_rxeq` and `rxeq_done` signals in *Table 5-15: Dynamic Reconfiguration Interface Signals*. • Updated the description of `tx_clkout_1g` signal in *Table 5-6: Clock and Reset Signals*. • Added a note about performing read-modify-writes for all registers in *1G/10GbE PHY Register Definitions* section. • Added a clarification about reset sequencer in the *1G/10GbE PHY Clock and Reset Interfaces* section on page 5-7. • Updated `tx_clkout_1g`, `rx_clkout_1g`, `tx_coreclkin_1g`, and `rx_coreclkin_1g` connections in *Figure 5-3: Clocks for Standard and 10G PCS and TX PLLs*.
XAUI	2.6	Added the statement "This register is only available in the hard XAUI implementation" for 0x82 and 0x83, polarity inversion" for 0x082 and 0x083, polarity inversion registers.
Custom PHY IP Core	2.6	Made the following changes: • Corrected the description of `tx_datak` signal in *Table 9-12: Avalon -ST TX Interface Signals*. • Corrected the available word alignment pattern lengths for 20 bit PMA-PCS interface width in manual mode in *Table 9-6: More Information About Word Aligner Functions*. • Updated the descriptions of `tx_cal_busy` and `rx_cal_busy` interface signals.
Low Latency PHY IP Core	2.6	Updated the descriptions of `tx_cal_busy` and `rx_cal_busy` interface signals.

Chapter	Document Version	Changes Made
Deterministic Latency PHY IP Core	2.6	Made the following changes: • Corrected the description of `tx_datak` signal in *Table 11-8: Avalon-ST TX Interface*. • Updated the descriptions of `tx_cal_busy` and `rx_cal_busy` interface signals.
Stratix V Transceiver Native PHY IP Core	2.6	Made the following changes: • Removed the description for `rx_clklow` and `rx_fref` ports from *Table 12-38: Native PHY Common Interfaces*. • Removed the ports `rx_clklow` and `rx_fref` from *Figure 12-5: Stratix V Native PHY Common Interfaces*. • Updated the description of `rx_10g_clk33out` clock signal in *Table 12-44: Name Dir Synchronous to tx_10g_coreclkin/rx_10g_coreclkin Description*. • Updated the description of `tx_pma_qpipullup` ,`tx_pma_qpipulldn` , and `rx_pma_qpipulldn` signals in *Table 12:38 - Native PHY Common Interfaces*. • Updated the descriptions of `tx_cal_busy` and `rx_cal_busy` interface signals. • Added`ext_pll_clk` signal to *Figure 12-5: Stratix V Native PHY Common Interfaces* and added its description in *Table 12-38: Native PHY Common Interfaces*.
Arria V Transceiver Native PHY IP Core	2.6	Made the following changes: • Removed the description for `rx_clklow` and `rx_fref` ports from *Table 13-31: Native PHY Common Interfaces*. • Removed the ports `rx_clklow` and `rx_fref` from *Figure 13-3: Native PHY Common Interface Ports*. • Updated the descriptions of `tx_cal_busy` and `rx_cal_busy` interface signals. • Added`ext_pll_clk` signal to *Figure 13-3: Common Interface ports* and added its description in *Table 13-18: Native PHY Common Interfaces*.

Chapter	Document Version	Changes Made
Arria V GZ Transceiver Native PHY IP Core	2.6	Made the following changes: • Removed the description for `rx_clklow` and `rx_fref` ports from *Table 14-38: Native PHY Common Interfaces*. • Removed the ports `rx_clklow` and `rx_fref` from *Figure 14-5: Arria V GZ Native PHY Common Interfaces*. • Updated the description of `rx_10g_clk33out` clock signal in *Table 14-44: Name Dir Synchronous to tx_10g_coreclkin/rx_10g_coreclkin Description*. • Updated the description of `tx_pma_qpipullup`, `tx_pma_qpipulldn`, and `rx_pma_qpipulldn` signals in *Table 14:38 - Native PHY Common Interfaces*. • Updated the descriptions of `tx_cal_busy` and `rx_cal_busy` interface signals. • Added`ext_pll_clk` signal to *Figure 14-5: Common Interfaces Ports* and added its description in *Table 14-38: Native PHY Common Interfaces*.
Cyclone V Transceiver Native PHY IP Core	2.6	Made the following changes: • Removed the description for `rx_clklow` and `rx_fref` ports from *Table 15-15: Native PHY Common Interfaces*. • Removed the ports `rx_clklow` and `rx_fref` from *Figure 15-3: Common Interfaces Ports*. • Updated the descriptions of `tx_cal_busy` and `rx_cal_busy` interface signals. • Added `ext_pll_clk` signal to *Figure 15-3: Common Interface Ports* and added its description in *Table 15-15: Native PHY Common Interfaces*.
Transceiver Reconfiguration Controller IP Core Overview	2.6	Made the following changes: • Added a footnote for the `Polarity` register in *Table 16-12: EyeQ Offsets and Values*. • Updated the description of `Control` register in *Table 16-18: ATX PLL Tuning Offsets and Values* to clarify the conditions when `tx_cal_busy` gets asserted. • Removed `physical channel address` register description from all the tables describing reconfiguration registers for different blocks. • Updated all instances of the note about undefined register bits. • Updated the description of `tx_cal_busy` and `rx_cal_busy` interface signals.

Chapter	Document Version	Changes Made
Analog Parameters Set Using QSF Assignments	2.6	Made the following changes: • Corrected values for `XCVR_REFCLK_PIN_TERMINATION`. `DC_COUPLING_INTERNAL_100_OHM` should be `DC_COUPLING_INTERNAL_100_OHMS`. • Removed the options for `XCVR_TX_COMMON_MODE_VOLTAGE` and `XCVR_RX_COMMON_MODE_VOLTAGE` assignments and added a note to use these assignments for Arria V, Arria V GZ, Cyclone V, and Stratix V devices. • Removed the options for `XCVR_GT_TX_COMMON_MODE_VOLTAGE` and `XCVR_GT_RX_COMMON_MODE_VOLTAGE` and added a note to use these assigments for Stratix V GT devices.

Chapter	Document Version	Changes Made
Introduction	2.5	Added information on running `ip-make-simscript` for designs including multiple transceiver PHYs.
10GBASE-R PHY	2.5	Made the following changes: • Corrected description of *Table 3-2 Latency for TX and RX PCS and PMA in Stratix V Devices*. The FPGA fabric to PCS interface width is 64 bits. • Added the description for a new parameter - **PCS / PMA interface width** in *General Option Parameters* section. • Added frequency for `rx_recovered_clk[<n>:0]` . It's 257.8 MHz. • Updated the descriptions of `rx_latency_adj_10g` and `tx_latency_adj_10g`. Changed the width of these signals for all references. • Added description for **Enable embedded reset controller** parameter *General Option Parameters* section. • Added a new section *Optional Reset Control and Status Interface*.
Backplane Ethernet 10GBASE-KR PHY	2.5	Made the following changes: • Updated the descriptions of `xgmii_tx_clk` and `xgmii_rx_clk` in *10GBASE-KR PHY Data Interfaces* section. • Updated the descriptions of `rx_latency_adj_1g` and `tx_latency_adj_1g`. Changed the width of these signals for all references. • Added *SDC Timing Constraints* topic. • Added parameter description for **Use M20K for FEC Buffer (if available)**.

Chapter	Document Version	Changes Made
1G/10GbE Ethernet PHY IP Core	2.5	Made the following changes: • Corrected definition of `gxmii_rx_d`. This signal is synchronous to `tx_clkout_1g`. • Added frequency for `rx_recovered_clk[<n>:0]` . It's 257.8 MHz. • Updated the descriptions of `rx_latency_adj_1g` and `tx_latency_adj_1g`. Changed the width of these signals for all references.
XAUI	2.5	Made the following changes: • Added fact that both bits of the `reset_control` register at 0x042 self-clear. • Added *SDC Timing Constraints* topic.
Interlaken	2.5	Added additional information about SDC timing constraints.
PHY IP Core for PCI Express	2.5	Made the following changes: • Removed the reference and description for `tx_invpolarity` and `rx_invpolarity` registers from *Register Interface and Register Descriptions* section.
Custom PHY IP Core	2.5	Made the following changes: • Added information on bit mapping for `tx_parallel_data` and `rx_parallel_data`. • Changed the introduction of *Optional Status Interfaces* section. This section applies for both TX and RX. • Added a note related to auto-negotiation state machine in *Rate Match FIFO Parameters* section. • Updated the description of `rx_bitslip` signal. • Added *SDC Timing Constraints* topic.
Low Latency PHY IP Core	2.5	Added *SDC Timing Constraints* topic.
Deterministic Latency PHY IP Core	2.5	Made the following changes: • Updated the *Channel Placement and Utilization for Deterministic Latency PHY* with details for Arria V and Arria V GZ devices. • Updated the table for "Signal Definitions for rx_parallel_data with and without 8B/10B Encoding". • Added *SDC Timing Constraints* topic.

Chapter	Document Version	Changes Made
Stratix V Transceiver Native PHY IP Core	2.5	Made the following changes: • Corrected Figure 12-4 showing the 10G PCS datapath. This datapath does not include hard IP blocks to implement KR-FEC. • Corrected errors in *Standard PCS Pattern Generators* section. • Updated the description of **Number of TX PLLs** parameter in "TX PMA Parameters" table. • Updated the description of **Selected Clock Network** parameter in "TX PLL Parameters" table. • Added a note related to auto-negotiation state machine in *Rate Match FIFO Parameters* section. • Updated the description of `rx_std_bitslip` signal. • Added information on PRBS-8 Standard PCS pattern generator.
Arria V Transceiver Native PHY IP Core	2.5	Made the following changes: • Updated the description of **Number of TX PLLs** parameter in "TX PMA Parameters" table. • Updated the description of **Selected Clock Network** parameter in "TX PLL Parameters" table. • Added a note related to auto-negotiation state machine in *Rate Match FIFO Parameters* section. • Updated the description of `rx_std_bitslip` signal. • Updated the table for "Signal Definitions for rx_parallel_data with and without 8B/10B Encoding".
Arria V GZ Transceiver Native PHY IP Core	2.5	Made the following changes: • Updated the description of **Number of TX PLLs** parameter in "TX PMA Parameters" table. • Updated the description of **Selected Clock Network** parameter in "TX PLL Parameters" table. • Updated the table for "Signal Definitions for rx_parallel_data with and without 8B/10B Encoding". • Added a note related to auto-negotiation state machine in *Rate Match FIFO Parameters* section. • Updated the description of `rx_std_bitslip` signal.
Cyclone V Transceiver Native PHY IP Core	2.5	Made the following changes: • Updated the description of **Number of TX PLLs** parameter in "TX PMA Parameters" table. • Updated the description of **Selected Clock Network** parameter in "TX PLL Parameters" table. • Added a note related to auto-negotiation state machine in *Rate Match FIFO Parameters* section.

Chapter	Document Version	Changes Made
Transceiver Reconfiguration Controller IP Core Overview	2.5	Made the following changes: • Updated table for "Device Support for Dynamic Reconfiguration" to indicate that Arria V and Cyclone ® V devices support TX PLL switching. • Added a note in table for "PMA Offsets and Values" to indicate possible methods to modify RX linear equalization settings. • Added a note related to PLL reference clock status in *Transceiver Reconfiguration Controller DFE Registers* section and *ATX PLL Calibration* section. • Updated the description of `mgmt_clk_clk` signal in *Reconfiguration Management Interfaces* section. • Changed the name of "one time adaptation mode" to "triggered dfe mode" and updated the *Controlling DFE Using Register-Based Reconfiguration* section. • Added a note in *Transceiver Reconfiguration Controller EyeQ Registers* section. • Added a note related to default values in *Transceiver Reconfiguration Controller EyeQ Registers* section.
Transceiver Reset Controller IP Core Overview	2.5	Made the following changes: • Updated the description of `rx_manual` signal in *Interfaces for Transceiver PHY Reset Controller* section. • Updated the description of `pll_select` signal. • Added a new section "Usage Examples for `pll_select`".
Analog Parameters Set Using QSF Assignments	2.5	Made the following changes: • Updated definitions of `XCVR_RX_SD_ENABLE`, `XCVR_RX_SD_OFF`, `XCVR_RX_SD_ON`, and `XCVR_RX_SD_THRESHOLD`.These settings are now available for SATA and SAS in addition to PCIe PIPE. • Added documentation for `XCVR_ANALOG_SETTINGS_PROTOCOL` setting. • Added warnings that there are restrictions `XCVR_TX_VOD`, `XCVR_TX_PRE_EMP_1ST_POST_TAP`, `XCVR_TX_PRE_EMP_2ND_POST_TAP`, and `XCVR_TX_PRE_EMP_PRE_TAP` for Stratix V and Arria V GZ devices. • Added warnings for restrictions on `XCVR_TX_PRE_EMP_1ST_POST_TAP` for Arria V and Cyclone V devices. • Changed default value for `XCVR_TX_VOD_PRE_EMP_CTRL_SRC` It's `DYNAMIC_CTL` for PCIe and `RAM_CTL` for other protocols. • Corrected example showing how to override the `master_ch_number`. The override must be applied to the transceiver instance, not the top-level PHY wrapper.

Date	Document Version	Changes Made
1G/10Gbps Ethernet PHY IP Core	2.4	
Backplane Ethernet 10GBASE-KR PHY IP Core	2.4	Added descriptions of FEC-related bits: C2[8], CB[26:25].
PHY IP Core for PCI Express (PIPE)	2.4	

Date	Document Version	Changes Made
1G/10Gbps Ethernet PHY IP Core	2.3	Changed speed of `rx_recovered_clk` from 125 MHz or 156.25 MHz to 125 MHz or 257.8125 MHz .
Backplane Ethernet 10GBASE-KR PHY IP Core	2.3	Changed speed of `rx_recovered_clk` from 125 MHz or 156.25 MHz to 125 MHz or 257.8125 MHz .
PHY IP Core for PCI Express (PIPE)	2.3	Added definition for `pipe_tx_data_valid`

Date	Document Version	Changes Made
PHY IP Core for PCI Express	2.2	Corrected SDC timing constraint for 62.5 MHz. Clock name is clk_g1.
Stratix V Native PHY	2.2	Correction: You can specify PLL merging using the `XCVR_TX_PLL_RECONFIG_GROUP` QSF setting, not the `FORCE_MERGE_PLL` QSF setting.
Arria V Native PHY	2.2	Correction: You can specify PLL merging using the `XCVR_TX_PLL_RECONFIG_GROUP` QSF setting, not the `FORCE_MERGE_PLL` QSF setting.
Arria V GZ Native PHY	2.2	Correction: You can specify PLL merging using the `XCVR_TX_PLL_RECONFIG_GROUP` QSF setting, not the `FORCE_MERGE_PLL` QSF setting.
Cyclone V Native PHY	2.2	Correction: You can specify PLL merging using the `XCVR_TX_PLL_RECONFIG_GROUP` QSF setting, not the `FORCE_MERGE_PLL` QSF setting.
Transceiver Reconfiguration Controller		
May 2013	2.2	Update to Transceiver Reconfiguration Controller chapter. Table 16-3 showing resource utilization for Stratix V devices, the timing unit should be us, microseconds, not ms, milliseconds.

Date	Document Version	Changes Made
Introduction		
April 2013	2.1	Update to introduction. Renamed heading "Additional Transceiver PHYs" to "Non-Protocol-Specific Transceiver PHYs."
Getting Started		
April 2013	2.1	No changes from previous release.
10GBASE-R		
April 2013	2.1	No changes from previous release.
10GBASE-KR		
April 2013	2.1	No changes from previous release.
1Gbe/10GbE		
April 2013	2.1	No changes from previous release.
XAUI		
April 2013	2.1	Fixed minor topographical error in heading.
Interlaken		
April 2013	2.1	No changes from previous release.
PHY IP Core for PCI Express		
April 2013	2.1	No changes from previous release.
Custom PHY		
April 2013	2.1	No changes from previous release.
Low Latency PHY		
April 2013	2.1	No changes from previous release.
Deterministic Latency PHY		
April 2013	2.1	No changes from previous release.
Stratix V Native PHY		
April 2013	2.1	Removed Arria V GT sentence on first page.
Arria V Native PHY		
April 2013	2.1	No changes from previous release.
Arria V GZ Native PHY		
April 2013	2.1	Removed Arria V GT sentence on first page.
Cyclone V Native PHY		
April 2013	2.1	No changes from previous release.

Date	Document Version	Changes Made
Transceiver Reconfiguration Controller		
April 2013	2.1	Rename table 16-13 to DFE Registers. Fix typo in Reconfig Addr column changed 7'h11 to 7'h19. In Table 16-8, removed the DCD Calibration registers row.
Transceiver Reset Controller		
April 2013	2.1	No changes from previous release.
Transceiver PLL for Arria V, Arria V GZ, and Stratix V Devices		
April 2013	2.1	No changes from previous release.
Analog Parameters Set Using QSF Assignment		
April 2013	2.1	Fix typo in the "Analog Settings for Arria V GZ Devices" table.
Migrating from Stratix IV to Stratix V Devices		
April 2013	2.1	No changes from previous release.

Date	Document Version	Changes Made
Introduction		
March 2013	2.0	No changes from previous release.
Getting Started		
March 2013	2.0	No changes from previous release.
10GBASE-R		
March 2013	2.0	No changes from previous release.
10GBASE-KR		
March 2013	2.0	Made the following changes: • Improved the description of automatic speed detection. • Updated speed grade information. • Updated definition of `KR AN Link Ready[5:0]` to include 1000BASE-KX. • Added the following registers `SEQ LT timeout` at 0xB1, Bit 2 and `SEQ Reconfig Mode[5:0]` 0xB1, Bits[13:8] registers • Revised Functional Description section. • Corrected typos in specifications of address bits `Ovride LP Coef enable` , `Updated TX Coef new`and `Updated RX coef new` . • Corrected encodings for `ber_time_k_frames`, `ber_time_frames` and `ber_time_m_frames`.

Date	Document Version	Changes Made
		1Gbe/10GbE
March 2013	2.0	Made the following changes: • Updated speed grade information. • Removed definition of `Disable AN Timer` bit. It is not used for this variant. • Added fact that `RESTART_AUTO_NEGOTIATION` bit is self-clearing (0x90, bit 9) • Added fact that half-duplex mode is not supported. (0x94, bit 6) • Added fact that the next page bit is not supported. (0x94, bit 15)
		XAUI
March 2013	2.0	Added Arria V, Arria V GZ and Cyclone V to the list of devices that do not support the `pma_tx_pll_is_locked` register in **Table 6-15: XAUI PHY IP Core Registers**.
		Interlaken
March 2013	2.0	No changes from previous release.
		PHY IP Core for PCI Express
March 2013	2.0	Added SDC constraints for Gen3 clocking.
		Custom PHY
March 2013	2.0	No changes from previous release.
		Low Latency PHY
March 2013	2.0	No changes from previous release.
		Deterministic Latency PHY
March 2013	2.0	No changes from previous release.
		Stratix V Native PHY
March 2013	2.0	Updated definition of **User external TX PLL** to include information on how to instantiate an external PLL.
		Arria V Native PHY
March 2013	2.0	Updated definition of **User external TX PLL** to include information on how to instantiate an external PLL.
		Arria V GZ Native PHY
March 2013	2.0	Updated definition of **User external TX PLL** to include information on how to instantiate an external PLL.
		Cyclone V Native PHY

Date	Document Version	Changes Made
March 2013	2.0	Updated definition of **User external TX PLL** to include information on how to instantiate an external PLL.
Transceiver Reconfiguration Controller		
March 2013	2.0	Made the following changes: • Updated definition of`TX PLL select` at 0x4 in **Table 16-21: PLL Reconfiguration Offsets and Values.** • Changed **Figure 16-5: MIF File Format**to match data in **Table 16-26**. • Changed **Example 16-11** to read the `busy` bit after each command, `read_32 0x3a`. • Added SDC timing constraints. • Changed `RX Equalization Control`0x11 in **Table 16-10: PMA Offsets and Values** to `RW`. This change is available starting with Quartus II 12.1 SP1. • Clarified encodings for `RX equalization DC gain` in **Table 16-10: PMA Offsets and Values**. • Clarified encodings for pre-emphasis pre-tap and pre-emphasis second post-tap in **Table 16-10: PMA Offsets and Values**. • Clarified fact that you can only connect a single Transceiver Reconfiguration Controller to a single transceiver PHY. • Updated recommendations for use of DCD in Arria V and Cyclone V devices. You should use DCD if the data rate is greater than 4.9152 Gbps or if there is dynamic TX PLL switching and the data rate is greater than 4.9152 Gbps. Updated address map and illustration to include DCD function.
Transceiver Reset Controller		
March 2013	2.0	Added `tx_ready` and `rx_ready` to **Figure 17-1**.
Transceiver PLL for Arria V, Arria V GZ, and Stratix V Devices		
March 2013	2.0	Initial Release.
Analog Parameters Set Using QSF Assignment		

Date	Document Version	Changes Made
March 2013	2.0	Made the following changes. • Changed choices for `XCVR_RX_SD_ENABLE` from **TRUE/FALSE** to **On/Off** • Corrected definitions of `XCVR_IO_PIN_TERMINATION` and `XCVR_GT_IO_PIN_TERMINATION` which were reversed. • Added references to Knowledge Base Solution showing the mapping of Transceiver Toolkit settings to `XCVR_TX_PRE_EMP_PRE_TAP`, `XCVR_TX_PRE_EMP_INV_PRE_TAP` and `XCVR_TX_PRE_EMP_PRE_TAP_USER` for Arria V GZ and Stratix V devices. • Added references to Knowledge Base Solution showing the mapping of Transceiver Toolkit settings to `XCVR_TX_PRE_EMP_2ND_POST_TAP`, `XCVR_TX_PRE_EMP_INV_2ND_TAP`, and `XCVR_TX_PRE_EMP_2ND_POST_TAP_USER` for Arria V GZ and Stratix V devices.
Migrating from Stratix IV to Stratix V Devices		
March 2013	2.0	No changes from previous release.

Date	Document Version	Changes Made
Introduction and Getting Started		
February 2013	1.9	• Reformatted.
10GBASE-R PHY		
February 2013	1.9	• Reformatted. • Corrected definition of the **PLL type** parameter. Altera recommends the ATX PLL for data rates greater than 8 Gbps.
Backplane Ethernet 10GBASE-KR PHY		
February 2013	1.9	• Reformatted. • Removed description of PMA `reset_ch_bitmask` at 0x41 which is not available. Added definition of digital and analog resets at 0x44, bits 1-3. • Removed definitions of `trn_in_trigger` and `trn_out_trigger` buses which are not used. • Corrected direction of `xgmii_rx_clk` in pinout figure.
1G/10GbE PHY		

Date	Document Version	Changes Made
February 2013	1.9	• Reformatted. • Corrected definition of `rx_data_ready`. This signal is used and indicates that the PCS is ready to receive data. • Removed description of PMA `reset_ch_bitmask` at 0x41 and reset_control at 0x42 which are not available. • Removed definitions of `trn_in_trigger` and `trn_out_trigger` buses which are not used.
XAUI PHY		
February 2013	1.9	• Reformatted.
Interlaken PHY		
February 2013	1.9	• Reformatted. • Improved definitions of `rx_parallel_data<n>[68]`, `rx_dataout_bp<n>` and typo in definition of `tx_user_clkout`.
PHY IP Core for PCI Express (PIPE)		
February 2013	1.9	• Reformatted.
Custom PHY		
February 2013	1.9	• Reformatted.
Low Latency PHY		
February 2013	1.9	• Reformatted.
Deterministic Latency PHY		
February 2013	1.9	• Reformatted. • Corrected headings in Table 11-4. The TX PMA Latency in UI and RX PMA Latency in UI were previously reversed. • In Table 11-3, added explanation of a latency uncertainty of 0.5 cycles when the byte serializer/deserializer is turned on. The location of the alignment pattern which can be in the upper or lower symbol.
Stratix V Native PHY		
February 2013	1.9	• Reformatted. • Added missing descriptions of Interlaken parameters to 10G RX FIFO section. • Improved definition of `pll_powerdown` signal.
Arria V Native PHY		

Date	Document Version	Changes Made
February 2013	1.9	• Reformatted. • Removed QPI signals from Figure showing Arria V Native PHY Common Interfaces. These signals are not available for Arria V devices. • Removed SDC constraints for 10G signals which are not available for Arria V.
Arria V GZ Native PHY		
February 2013	1.9	• Reformatted. • Improved definition of `pll_powerdown` signal.
Cyclone V Native PHY		
February 2013	1.9	• Reformatted. • Removed information about PMA direct mode. PMA direct mode is not supported for Cyclone V devices. • Improved definition of `pll_powerdown` signal.
Transceiver Reconfiguration Controller		
February 2013	1.9	• Reformatted. • Expanded definition of *mgmt_clk_clk* to include constraints when CvP is enabled and frequency range for Arria V GZ and Cyclone V devices. • Corrected address for channel 2 in register-based read examples.
Transceiver PHY Reset Controller		
February 2013	1.9	• Reformatted. • Improved definition of `pll_powerdown` and `rx_manual`signals.
Analog Parameters Set Using QSF Assignments		
February 2013	1.9	• Reformatted. • Added the following settings to the Arria V and Cyclone V tables: `XCVR_RX_SD_ON`, `XCVR_RX_SD_OFF`, `XCVR_RX_SD_THRESHOLD`, `CDR_BANDWIDTH_PRESET`, `XCVR_RX_COMMON_MODE_VOLTAGE`, `XCVR_TX_COMMON_MODE_VOLTAGE`, `XCVR_TX_RX_DET_ENABLE`, and `XCVR_RX_DET_MODE`.
Migrating from Stratix IV to Stratix V Devices		
February 2013	1.9	• Reformatted.
Introduction		

Date	Document Version	Changes Made
November 2012	1.8	• Expanded discussion of the Arria V, Arria V GZ, Cyclone V, and Stratix V Transceiver Native PHY IP Cores. • Added Riviera-PRO Aldec simulation directory.
10GBASE-R PHY		
November 2012	1.8	• Added support for IEEE 1588 Precision Time Protocol. • Added Arria V GZ support. • Enabled `RCLR_BER_COUNT` (0x81, bit 3) and `HI_BER` (0x82, bit 1) for Arria V GZ and Stratix V devices. • Moved Analog Options to a separate chapter.
10GBASE-KR PHY		
November 2012	1.8	• Initial release.
1G/10 Gbps Ethernet PHY		
November 2012	1.8	• Initial release.
XAUI PHY		
November 2012	1.8	• Added Arria V GZ support. • Moved Analog Options to a separate chapter. • Added constraint for `tx_digitalreset` when TX PCS uses bonded clocks.
Interlaken PHY		
November 2012	1.8	• Added Arria V GZ support. • Added 12500 Mbps lane rate. • Moved Analog Options to a separate chapter. • Removed recommendation to use /40 for `tx_user_clkout` and `rx_coreclkin`. Data rates between /40 and /67 all work reliably.
PHY IP Core for PCI Express (PIPE)		

Date	Document Version	Changes Made
November 2012	1.8	• Added Gen3 support. • Added Arria V GZ support. • Added ×2 support. • Added discussion of link equalization for Gen3. • Added timing diagram showing rate change to Gen3. • Revised presentation of signals. • Corrected the definition of `rx_eidleinfersel[3<n>-1:0]`. • Moved Analog Options to a separate chapter. • Updated section on Logical Lane Assignment Restrictions. • Removed the following statement from the definition of `pll_powerdown`. Asserting `pll_powerdown` no longer powers down `tx_analogreset`. `tx_analogreset` is a separate signal.
Custom PHY IP Core		
November 2012	1.8	• Added Cyclone V support. • Moved Analog Options to a separate chapter. • Added constraint for tx_digitalreset when TX PCS uses bonded clocks. • Corrected description of manual word alignment mode.
Low Latency PHY IP Core		
November 2012	1.8	• Added Cyclone V support. • Moved Analog Options to a separate chapter. • Added constraint for `tx_digitalreset` when TX PCS uses bonded clocks. • Added RX bitslip option for the word aligner when the 10G PCS is selected. • Added description of `reset_fine_control` register at 0x044. This register is available when not using the embedded reset controller.
Deterministic Latency PHY IP Core		
November 2012	1.8	• Added Cyclone V support. • Moved Analog Options to a separate chapter.
Stratix V Transceiver Native PHY		
November 2012	1.8	• Added support for Standard and 10G datapaths. • Added QPI interface. • Moved Analog Options to a separate chapter. • Added constraint for `tx_digitalreset` when TX PCS uses bonded clocks.

Date	Document Version	Changes Made
Arria V Transceiver Native PHY		
November 2012	1.8	• Added support for Standard datapath. • Added support for multiple PLLs. • Moved Analog Options to a separate chapter. • Added constraint for `tx_digitalreset` when TX PCS uses bonded clocks.
Arria V GZ Transceiver Native PHY		
November 2012	1.8	• Initial release.
Cyclone V Transceiver Native PHY		
November 2012	1.8	• Initial release.
Reconfiguration Controller		
November 2012	1.8	• Added `MIF addressing mode` option. Byte and word (16 bits) addressing are available. • Added ATX PLL reference clock switching and reconfiguration of ATX PLL settings, including counters. • Added support for ATX PLL reconfiguration. • Added statement that if you are using the EyeQ monitor when DFE is enabled, if you must use the EyeQ monitor with a 1D-eye. • Corrected definition of `DFE_control` bit at 0xa. This register is write only. • Removed duty cycle calibration. This function is run automatically during the power-on sequence. • Added DFE support including examples showing how to program this function. • Added DCD for Arria V devices. • Updated data for writes in Streamer Mode 1 Reconfiguration. • Changed data value to write in step 7 of Streamer-Based Reconfiguration. • Changed data value to write to setup streaming in Reconfiguration of Logical Channel 0 Using a MIF.
Transceiver PHY Reset Controller		
November 2012	1.8	• Added Arria V GZ support. • Added SDC constraint for `tx_digitalreset` when TX PCS uses bonded clocks.
Analog Parameters Set Using QSF Assignments		

Date	Document Version	Changes Made
November 2012	1.8	• Created separate chapter for analog parameters that were previously listed in the individual transceiver PHY chapters. • Changed default value for `XCVR_GT_RX_COMMON_MODE_VOLTAGE` to 0.65V.
Introduction and Getting Started		
June 2012	1.7	• Added brief discussion of the Stratix V and Arria V Transceiver Native PHY IP Cores.
Getting Started		
June 2012	1.7	• No changes from the previous release.
10GBASE-R PHY		
June 2012	1.7	• Added the following QSF settings to all transceiver PHY: `XCVR_TX_PRE_EMP_PRE_TAP_USER`, `XCVR_TX_PRE_EMP_2ND_POST_TAP_USER`, and 11 new settings for GT transceivers. • Added Arria V device support. • Changed the default value for `XCVR_REFCLK_PIN_TERMINATION` from DC_coupling_internal_100_Ohm to **AC_coupling**. • Changed references to Stratix IV GX to Stratix IV GT. This IP core only supports Stratix IV GT devices. • Added optional `pll_locked` status signal for Arria V and Stratix V devices. Added optional `rx_coreclkin`. • Added arrows Transceiver Reconfiguration Controller IP Core connection to block diagram. • Changed the maximum frequency of `phy_mgmt_clk` to 150 MHz if the same clock is used for the Transceiver Reconfiguration Controller IP Core. • Added the following restriction in the dynamic reconfiguration section: three channels share an Avalon-MM slave interface which must connect to the same Transceiver Reconfiguration Controller IP Core. • Added example showing how to override the logical channel 0 channel assignment in Stratix V devices. • Added table showing latency through PCS and PMA for Arria V and Stratix V devices.
XAUI PHY		

Date	Document Version	Changes Made
June 2012	1.7	• Added the following QSF settings to all transceiver PHY: `XCVR_TX_PRE_EMP_PRE_TAP_USER`, `XCVR_TX_PRE_EMP_2ND_POST_TAP_USER`, and 11 new settings for GT transceivers. • Added reference Transceiver device handbook chapters for detailed explanation of PCS blocks. • Changed the default value for `XCVR_REFCLK_PIN_TERMINATION` from DC_coupling_internal_100_Ohm to **AC_coupling**. • Changed the maximum frequency of `phy_mgmt_clk` to 150 MHz if the same clock is used for the Transceiver Reconfiguration Controller IP Core. • Added example showing how to override the logical channel 0 channel assignment in Stratix V devices. • Expanded definition of **External PMA control and configuration** parameter. • Added the following restriction in the dynamic reconfiguration section: three channels share an Avalon-MM slave interface which must connect to the same Transceiver Reconfiguration Controller IP Core. • Added note that `cal_blk_powerdown` register is not available for Stratix V devices.
		Interlaken PHY
June 2012	1.7	• Added support for custom, user-defined, data rates. • Added the following QSF settings to all transceiver PHY: `XCVR_TX_PRE_EMP_PRE_TAP_USER`, `XCVR_TX_PRE_EMP_2ND_POST_TAP_USER`, and 11 new settings for GT transceivers. • Changed the default value for `XCVR_REFCLK_PIN_TERMINATION` from DC_coupling_internal_100_Ohm to **AC_coupling**. • Updated the definition of `tx_sync_done`. It is no longer necessary to send pre-fill data before `tx_sync_done` and `tx_ready` are asserted. • Updated definition of `tx_datain_bp<n>`. • Added arrows indicating Transceiver Reconfiguration Controller IP Core connection to block diagram. • Changed the maximum frequency of `phy_mgmt_clk` to 150 MHz if the same clock is used for the Transceiver Reconfiguration Controller IP Core. • Clarified signal definitions. • Added the following restriction in the dynamic reconfiguration section: three channels share an Avalon-MM slave interface which must connect to the same Transceiver Reconfiguration Controller IP Core.

Date	Document Version	Changes Made
PHY IP Core for PCI Express (PIPE)		
June 2012	1.7	• Added the following QSF settings to all transceiver PHY: `XCVR_TX_PRE_EMP_PRE_TAP_USER`, `XCVR_TX_PRE_EMP_2ND_POST_TAP_USER`, and 11 new settings for GT transceivers. • Added reference Stratix V Transceiver Architecture chapter for detailed explanation of PCS blocks. • Changed the default value for `XCVR_REFCLK_PIN_TERMINATION` from DC_coupling_internal_100_Ohm to **AC_coupling**. • Corrected definition of `tx_bitslipboundary_select` register. • Changed `pipe_rate` signal to 2 bits. • Added the following restriction in the dynamic reconfiguration section: three channels share an Avalon-MM slave interface which must connect to the same Transceiver Reconfiguration Controller IP Core.
Custom PHY IP Core		
June 2012	1.7	• Added the following QSF settings to all transceiver PHY: `XCVR_TX_PRE_EMP_PRE_TAP_USER`, `XCVR_TX_PRE_EMP_2ND_POST_TAP_USER`, and 11 new settings for GT transceivers. • Added reference to Stratix V Transceiver Architecture chapter for detailed explanation of the PCS blocks. • Updated definition of `rx_enapatternalign`: It is edge sensitive in most cases; however, if the PMA-PCS interface width is 10 bits, it is level sensitive. • Added definition for `rx_byteordflag` output status signal which is created when you enable the byte ordering block. • Changed the default value for `XCVR_REFCLK_PIN_TERMINATION` from DC_coupling_internal_100_Ohm to **AC_coupling**. • Added arrows indicating Transceiver Reconfiguration Controller IP Core connection to block diagram. • Changed the maximum frequency of `phy_mgmt_clk` to 150 MHz if the same clock is used for the Transceiver Reconfiguration Controller IP Core. • Added the following restriction in the dynamic reconfiguration section: three channels share an Avalon-MM slave interface which must connect to the same Transceiver Reconfiguration Controller IP Core.
Low Latency PHY IP Core		

Date	Document Version	Changes Made
June 2012	1.7	• Added the following QSF settings to all transceiver PHY: `XCVR_TX_PRE_EMP_PRE_TAP_USER`, `XCVR_TX_PRE_EMP_2ND_POST_TAP_USER`, and 11 new settings for GT transceivers. • Changed the default value for `XCVR_REFCLK_PIN_TERMINATION` from DC_coupling_internal_100_Ohm to **AC_coupling**. • Added arrows indicating Transceiver Reconfiguration Controller IP Core connection to block diagram. • Changed the maximum frequency of `phy_mgmt_clk` to 150 MHz if the same clock is used for the Transceiver Reconfiguration Controller IP Core. • Added the following restriction in the dynamic reconfiguration section: three channels share an Avalon-MM slave interface which must connect to the same Transceiver Reconfiguration Controller IP Core.
Deterministic Latency PHY IP Core		
June 2012	1.7	• Added the following QSF settings to all transceiver PHY: `XCVR_TX_PRE_EMP_PRE_TAP_USER`, `XCVR_TX_PRE_EMP_2ND_POST_TAP_USER`, and 11 new settings for GT transceivers. • Added PLL reconfiguration option. • Changed the default value for `XCVR_REFCLK_PIN_TERMINATION` from DC_coupling_internal_100_Ohm to **AC_coupling**. • Removed references to the byte serializer and deserializer which is not included in the datapath. • Added GUI option for `tx_clkout` feedback path for TX PLL to align the TX and RX clock domains and figure illustrating this approach. • Added tables showing the signals in TX and RX parallel data that correspond to data, control, and status signals with and without 8B/10B encoding. • Corrected definition of `rx_runnindisp`. This is a status output. • Added the following restriction in the dynamic reconfiguration section: three channels share an Avalon-MM slave interface which must connect to the same Transceiver Reconfiguration Controller IP Core.
Stratix V Transceiver Native PHY		
June 2012	1.7	• Initial release.
Arria V Transceiver Native PHY		
June 2012	1.7	• Initial release.
Transceiver PHY Reconfiguration Controller		

Date	Document Version	Changes Made
June 2012	1.7	• DFE now automatically runs offset calibration and phase interpolator (PI) phase calibration at power on. • Added section explaining how to generate a reduced MIF file. • Corrected definition of EyeQ control register. Writing a 1 to bit 0 enables the Eye monitor. • Corrected bit-width typos in PMA Analog Registers.
Transceiver PHY Reset Controller		
June 2012	1.7	• Initial release.
Custom		
March 2012	1.6	• Added register definitions for address range 0x080–0x085.
Low Latency PHY		
March 2012	1.6	• Removed register definitions for address range 0x080–0x085.
10GBASE-R		
February 2012	1.5	• Added datapath latency numbers for Stratix V devices. • Corrected bit range for `ERRORED_BLOCK_COUNT`. • Added statement that the `cal_blk_powerdown` (0x021) and `pma_tx_pll_is_locked` (0x022) registers are only available when the **Use external PMA control and reconfig** option is turned **On** on the **Additional Options** tab of the GUI. • Clarified that the BER count functionality is for Stratix IV devices only. • Removed `pma_rx_signaldetect` register. The 10GBASE-R PHY does not support this functionality.
XAUI		
February 2012	1.5	• Removed reset bits at register 0x081. The reset implemented Cat register 0x044 provides more comprehensive functionality. • Removed `pma_rx_signaldetect` register. The XAUI PHY does not support this functionality.
PCI Express (PIPE)		
February 2012	1.5	• Updated definition of `fixedclk`. It can be derived from `pll_ref_clk`.
Custom		
February 2012	1.5	• Removed register definitions for Low Latency PHY.

Date	Document Version	Changes Made
		Low Latency PHY
February 2012	1.5	• Added register definitions for Low Latency PHY.
		Deterministic Latency PHY
February 2012	1.5	• Removed `pma_rx_signaldetect` register. The Deterministic Latency PHY does not support this functionality. • Updated the definition of deterministic latency word alignment mode to include the fact that the word alignment pattern, which is currently forced to K28.5 = 0011111010 is always placed in the least significant byte (LSB) of a word with a fixed latency of 3 cycles.
		Transceiver Reconfiguration Controller
February 2012	1.5	• Added DFE.
		Introduction
December 2011	1.4	• Revised discussion of embedded reset controller to include the fact that this reset controller can be disabled for some transceiver PHYs.
		10GBASE-R
December 2011	1.4	• Removed description of calibration block powerdown register (0x021) which is not available for this transceiver PHY. • Changed definition of `phy_mgmt_clk_reset`. This signal is active high and level sensitive.
		XAUI
December 2011	1.4	• Changed definition of `phy_mgmt_clk_reset`. This signal is active high and level sensitive. • Added Arria II GX to device support table.
		Interlaken
December 2011	1.4	• Changed access mode for RX equalization, pre-CDR reverse serial loopback, and post-CDR reverse serial loopback to write only (WO). • Removed optional `rx_sync_word_err`, `rx_scrm_err`, and `rx_framing_err` status bits. • Changed definition of `phy_mgmt_clk_reset`. This signal is active high and level sensitive.
		PHY IP Core for PCI Express (PIPE)

Date	Document Version	Changes Made
December 2011	1.4	• Changed definition of `phy_mgmt_clk_reset`. This signal is active high and level sensitive.
Custom		
December 2011	1.4	• Added ×N and feedback compensation options for bonded clocks. • Added **Enable Channel Interface** parameter which is required for dynamic reconfiguration of transceivers. • Corrected formulas for signal width in top-level signals figure. • Changed definition of `phy_mgmt_clk_reset`. This signal is active high and level sensitive.
Low Latency PHY		
December 2011	1.4	• Added option to disable the embedded reset controller to allow you to create your own reset sequence. • Added ×N and feedback compensation options for bonded clocks. • Fixed name of `phy_mgmt_reset` signal. Should be `phy_mgmt_clk_reset`. Also, a positive edge on this signal initiates a reset. • Added **Enable Channel Interface** parameter which is required for dynamic reconfiguration of transceivers. • Corrected formulas for signal width in top-level signals figure. • Changed definition of `phy_mgmt_clk_reset`. This signal is active high and level sensitive.
Deterministic Latency PHY		
December 2011	1.4	• Removed **Enable tx_clkout feedback path for TX PLL** from the **General Options** tab of the Deterministic Latency PHY IP Core GUI. This option is unavailable in 11.1 and 11.1 SP1. • Changed definition of `phy_mgmt_clk_reset`. This signal is active high and level sensitive.
Transceiver Reconfiguration Controller		

Date	Document Version	Changes Made
December 2011	1.4	• Added duty cycle distortion (DCD) signal integrity feature. • Added PLL and channel reconfiguration using a memory initialization file (.mif). • Added ability to reconfigure PLLs, including the input reference clock or to change the PLL that supplies the high speed serial clock to the serializer without including logic to reconfigure channels. • Corrected values for RX equalization gain. 0–4 are available. • Corrected logical number in Interface Ordering with Multiple Transceiver PHY Instances. • Increased the number of channels that can share a PLL from 5 to 11 when feedback compensation is used. • Increased the number of channels that can connect to the Transceiver Reconfiguration Controller from 32 to 64. • Added section on requirements for merging PLLs.
Introduction		
November 2011	1.3	• Revised reset section. The 2 options for reset are now the embedded reset controller or user-specified reset controller. • Updated directory names in simulation testbench.
10GBASE-R PHY Transceiver		
November 2011	1.3	• Added support for Stratix V devices. • Added section discussing transceiver reconfiguration in Stratix V devices. • Removed `rx_oc_busy` signal which is included in the reconfiguration bus. • Updated QSF settings to include text strings used to assign values and location of the assignment which is either a pin or PLL.
XAUI Transceiver PHY		
November 2011	1.3	• The `pma_tx_pll_is_locked` is not available in Stratix V devices. • Added **base data rate**, **lane rate**, **input clock frequency**, and **PLL type** parameters. • Updated QSF settings to include text strings used to assign values and location of the assignment which is either a pin or PLL. • Added section on dynamic transceiver reconfiguration in Stratix V devices. • Removed Timing Constraints section. These constraints are included in the HDL code.

Date	Document Version	Changes Made
		Interlaken Transceiver PHY
November 2011	1.3	• Added `tx_sync_done` signal which indicates that all lanes of TX data are synchronized. • `tx_coreclk_in` is required in this release. • Added **base data rate**, **lane rate**, **input clock frequency**, and **PLL type** parameters. • Updated QSF settings to include text strings used to assign values and location of the assignment which is either a pin or PLL.
		PHY IP Core for PCI Express (PIPE)
November 2011	1.3	• Added `pll_powerdown` bit (bit[0] of 0x044) for manual reset control. You must assert this bit for 1 ms for Gen2 operation. • Added **PLL type** and **base data rate** parameters. • Updated QSF settings to include text strings used to assign values and location of the assignment which is either a pin or PLL.
		Custom Transceiver PHY
November 2011	1.3	• Added Arria V and Cyclone V support. • Added**base data rate**, **lane rate**, **input clock frequency**, and **PLL type** parameters. • Revised reset options. The 2 options for reset are now the embedded reset controller or a user-specified reset logic. • Updated QSF settings to include text strings used to assign values and location of the assignment which is either a pin or PLL.
		Low Latency PHY
November 2011	1.3	• Added base data rate, lane rate, input clock frequency, and PLL type parameters. • Updated QSF settings to include text strings used to assign values and location of the assignment which is either a pin or PLL. • Revised reset options. The 2 options for reset are now the embedded reset controller or a user-specified reset logic.
		Deterministic Latency
November 2011	1.3	• Initial release.
		Transceiver Reconfiguration Controller

Date	Document Version	Changes Made
November 2011	1.3	• Added MIF support to allow transceiver reconfiguration from a **.mif** file that may contain updates to multiple settings. • Added support for the following features: • EyeQ • AEQ • ATX tuning • PLL reconfiguration • DC gain and four-stage linear equalization for the RX channels • Removed Stratix IV device support. • Changed frequency range of `phy_mgmt_clk` to 100-125 MHz.
		All Chapters
July 2011	1.2.1	• Restricted frequency range of the `phy_mgmt_clk` to 90–100 MHz for the *Transceiver Reconfiguration Controller IP Core* chapter. There is no restriction on the frequency of `phy_mgmt_clk` for Stratix V devices in the 10GBASE-R, XAUI, Interlaken, PHY IP Core for PCI Express, Custom, and Low Latency PHYs; however, to use the same clock source for both, you must restrict this clock to 90–100 MHz. • Added column specifying availability of read and write access for PMA analog controls in the *Transceiver Reconfiguration Controller IP Core* chapter. • Renamed Avalon-MM bus in for Transceiver Reconfiguration Controller `reconfig_mgmt*`. • Provided frequency range for `phy_mgmt_clk` for the XAUI PHY IP Core in Arria II GX, Cyclone IV GX, HardCopy IV, and Stratix IV GX devices. • Added register descriptions for the automatic reset controller to the *Low Latency PHY IP Core* chapter. • Added two steps to procedure to reconfigure a PMA control in the *Transceiver Reconfiguration Controller* chapter. • Corrected RX equalization DC gain in *Transceiver Reconfiguration Controller* chapter. It should be 0–4. • Corrected serialization factor column in *Low Latency PHY IP Core* chapter.
		Introduction

Date	Document Version	Changes Made
May 2011	1.2	• Added simulation section. • Revised Figure 1–1 on page 1–1 to show the Transceiver Reconfiguration Controller as a separately instantiated IP core. • Added statement saying that the transceiver PHY IP cores do not support the NativeLink feature of the Quartus II software. • Revised reset section.
		Getting Started
May 2011	1.2	• No changes from previous release.
		10GBASE-R PHY Transceiver
May 2011	1.2	• Corrected frequency of `pll_ref_clk`. Should be 644.53125 MHz, not 644.53725 MHz. • Renamed `reconfig_fromgxb` and `reconfig_togxb reconfig_from_xcvr` and `reconfig_to_xcvr`, respectively.
		XAUI PHY Transceiver

May 2011	1.2	• Added support for DDR XAUI
		• Added support for Arria II GZ and HardCopy IV
		• Added example testbench
		• Renamed reconfig_fromgxb and reconfig_togxb reconfig_from_xcvr and reconfig_to_xcvr, respectively.
		• Updated definitions of rx_digital_reset and tx_digital_reset for the soft XAUI implementation in XAUI PHY IP Core Registers.
		• Changed description of rx_syncstatus register and signals to specify 2 bits per channel in hard XAUI and 1 bit per channel in soft XAUI implementations.
		• Corrected bit sequencing for 0x084, 0x085 and 0x088 in XAUI PHY IP Core Registers, as follows:
		  • `patterndetect` = 0x084, bits [15:8]
		  • `syncstatus` = 0x084, bits [7:0]
		  • `errordetect` = 0x085, bits [15:8]
		  • `disperr` = 0x085, bits [7:0]
		  • `rmfifofull` = 0x088, bits [7:4]
		  • `rmfifoempty` = 0x088, bits [3:0]

		Interlaken PHY Transceiver

Date	Document Version	Changes Made
May 2011	1.2	• Added details about the 0 ready latency for `tx_ready`. • Added PLL support to lane rate parameter description in Interlaken PHY General Options. • Moved dynamic reconfiguration for the transceiver outside of the Interlaken PHY IP Core. The reconfiguration signals now connect to a separate Reconfiguration Controller IP Core. • Added a reference to *PHY IP Design Flow with Interlaken for Stratix V Devices* which is a reference design that implements the Interlaken protocol in a Stratix V device. • Changed supported metaframe lengths from 1–8191 to 5–8191. • Added `pll_locked` output port. • Added `indirect_addr` register at 0x080 for use in accessing PCS control and status registers. • Added new **Bonded group size** parameter.
PHY IP Core for PCI Express PHY (PIPE)		
May 2011	1.2	• Renamed to PHY IP Core for PCI Express. • Moved dynamic reconfiguration for the transceiver outside of the PHY IP Core. The reconfiguration signals now connect to a separate Reconfiguration Controller IP Core. • Removed ×2 support.
Custom PHY Transceiver		

Date	Document Version	Changes Made

May 2011	1.2	• Added presets for the 2.50 GIGE and 1.25GIGE protocols.
		• Moved dynamic reconfiguration for the transceiver outside of the Custom PHY IP Core. The reconfiguration signals now connect to a separate Reconfiguration Controller IP Core.
		• Removed device support for Arria II GX, Arria II GZ, HardCopy IV GX, and Stratix IV GX.
		• Added the following parameters on the General tab:
		  • Transceiver protocol
		  • Create rx_recovered_clk port
		  • Force manual reset control
		• Added optional `rx_rmfifoddatainserted`, `rx_rmfifodatadelted`, `rx_rlv`, and `rx_recovered_clk` as output signals.
		• Added `phy_mgmt_waitrequest` to the PHY management interface.
		• Renamed `reconfig_fromgxb` and `reconfig_togxb` `reconfig_from_xcvr` and `reconfig_to_xcvr`, respectively.
		• Corrected address for 8-Gbps RX PCS status register in Table 9–18 on page 9–20.
		• Added special pad requirement for Byte ordering pattern. Refer to Table 9–6 on page 9–8.
		• Clarified behavior of the word alignment mode. Added note explaining how to disable all word alignment functionality.

Low Latency PHY Transceiver		
May 2011	1.2	• Moved dynamic reconfiguration for the transceiver outside of the Low Latency PHY IP Core. The reconfiguration signals now connect to a separate Reconfiguration Controller IP Core. • Moved dynamics reconfiguration for the transceiver outside of the Custom PHY IP Core. The reconfiguration signals now connect to a separate Reconfiguration Controller IP Core. • Renamed the `tx_parallel_clk` signal `tx_clkout`.
Transceiver Reconfiguration Controller		
May 2011	1.2	• Added Stratix V support. The Transceiver Reconfiguration Controller is only available for Stratix IV devices in the Transceiver Toolkit. • Added sections describing the number of reconfiguration interfaces required and restrictions on channel placement. • Added pre- and post-serial loopback controls. • Changed reconfiguration clock source. In 10.1, the Avalon-MM PHY Management clock was used for reconfiguration. In 11.0, the reconfiguration controller supplies this clock.

Date	Document Version	Changes Made
Migrating from Stratix IV to Stratix V		
May 2011	1.2	• Added discussion of dynamic reconfiguration for Stratix IV and Stratix V devices. • Added information on loopback modes for Stratix IV and Stratix V devices. • Added new parameters for Custom PHY IP Core in Stratix V devices.
All Chapters		
December 2010	1.11	• Corrected frequency range for the `phy_mgmt_clk` for the Custom PHY IP Core in Avalon-MM PHY Management Interface. • Added optional `reconfig_from_xcvr[67:0]` to XAUI Top-Level Signals—Soft PCS and PMA. Provided more detail on size of `reconfig_from_xcvr` in Dynamic Reconfiguration Interface Arria II GX, Cyclone IV GX, HardCopy IV GX, and Stratix IV GX devices. • Removed table providing ordering codes for the Interlaken PHY IP Core. Ordering codes are not required for Stratix V devices using the hard implementation of the Interlaken PHY. • Added note to 10GBASE-R release information table stating that "No ordering codes or license files are required for Stratix V devices." • Minor update to the steps to reconfigure a TX or RX PMA setting in the Transceiver Reconfiguration Controller chapter.
Introduction		
December 2010	1.1	• Revised reset diagram. • Added block diagram for reset. • Removed support for SOPC Builder.
Getting Started		
December 2010	1.1	• Removed description of SOPC Builder design flow. SOPC Builder is not supported in this release.
10GBASE-R PHY Transceiver		

Date	Document Version	Changes Made
December 2010	1.1	• • Added Stratix V support • Changed `phy_mgmt_address` from 16 to 9 bits. • Renamed management interface, adding `phy_ prefix` • Renamed `block_lock` and `hi_ber` signals `rx_block_lock` and `rx_hi_ber`, respectively. • Added top-level signals for external PMA and reconfiguration controller in Stratix IV devices. Refer to External PMA and Reconfiguration Signals. • Removed the `mgmt_burstcount` signal. • Changed register map to show word addresses instead of a byte offset from a base address.
XAUI PHY Transceiver		
December 2010	1.1	• Added support for Arria II GX and Cyclone IV GX with hard PCS • Renamed management interface, adding `phy_` prefix • Changed `phy_mgmt_address` from 16 to 9 bits. • Renamed many signals. Refer to XAUI Top-Level Signals—Soft PCS and PMA and "XAUI Top-Level Signals–Hard IP PCS and PMA" as appropriate. • Changed register map to show word addresses instead of a byte offset from a base address. • Removed the `rx_ctrldetect` and `rx_freqlocked` signals.
Interlaken PHY Transceiver		
December 2010	1.1	• Added simulation support in ModelSim SE, Synopsys VCS MX, Cadence NCSim • Changed number of lanes supported from 4–24 to 1–24. • Changed reference clock to be 1/20th rather than 1/10th the lane rate. • Renamed management interface, adding phy_ prefix • Changed phy_mgmt_address from 16 to 9 bits. • Changed many signal names, refer to Top-Level Interlaken PHY Signals.Changed register map to show word addresses instead of a byte offset from a base address.
PCI Express PHY (PIPE)		

Date	Document Version	Changes Made
December 2010	1.1	• Added simulation support in ModelSim SE • Added PIPE low latency configuration option • Changed `phy_mgmt_address` from 16 to 9 bits. • Changed register map to show word addresses instead of a byte offset from a base address. • Added `tx_ready`, `rx_ready`, `pipe_txswing`, and `pipe_rxeleciidle` signals • Added `rx_errdetect`, `rx_disperr`, and `rx_a1a2sizeout` register fields
Custom PHY Transceiver		
December 2010	1.1	• Added support for 8B/10B encoding and decoding in Stratix V devices • Added support for rate matching in Stratix V devices. • Added support for Arria II GX, Arria II GZ, HardCopy IV GX, and Stratix IV GX devices • Changed `phy_mgmt_address` from 8 to 9 bits. • Added many optional status ports and renamed some signals. Refer to Figure 9–2 on page 9–15 and subsequent signal descriptions. • Changed register map to show word addresses instead of a byte offset from a base address.
Low Latency PHY IP Core		
December 2010	1.1	• Renamed management interface, adding `phy_` prefix • Changed `phy_mgmt_address` from 16 to 9 bits. • Changed register map to show word addresses instead of a byte offset from a base address. • Removed `rx_offset_cancellation_done` signal. Internal reset logic determines when offset cancellation has completed. • Removed support for Stratix IV GX devices.
Transceiver Reconfiguration Controller		
December 2010	1.1	• Reconfiguration is now integrated into the XAUI PHY IP Core and 10GBASE-R PHY IP Core. • Revised register map to show word addresses instead of a byte offset from a base address.
Migrating from Stratix IV to Stratix V		
December 2010	1.1	• Changed phy_mgmt_address from 16 to 9 bits.