PCIe 4.0 PCS for the DesignWare® Cores PCIe 4.0 PHY

PHY Subsystem Databook

Preliminary Information
PCS Version:1.41
May 26, 2020

Copyright Notice and Proprietary Information

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at https://www.synopsys.com/company/legal/trademarks-brands.html

All other product or company names may be trademarks of their respective owners.

Free and Open-Source Software Licensing Notices

If applicable, Free and Open-Source Software (FOSS) licensing notices are available in the product installation.

Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

Synopsys, Inc.
www.synopsys.com

Contents

Revision History

PCS Version	Databook Version	Date	Description
1.41	1.30	May 26, 2020	Updated: ■ “Preface” on page 9 ■ “General Product Description” on page 14 ■ “Protocol Support” on page 16 ■ Table “RTL Parameters Available for PCS Configuration” on page 17 ■ “Standards Compliance” on page 20 ■ Table “pipe_laneX_pclk Frequencies” on page 24 ■ Table “pipe_laneX_max_pclk Frequencies” on page 25 ■ “Signal Overview” on page 29 ■ “Signal Descriptions for x4 Configuration” on page 31 ■ Table “PCIe 4.0 PCS Clock Dependency” on page 230 ■ “Reference Clock Input” on page 235 ■ Table “RX Datapath Latency” on page 238 ■ “Power State Transitions” on page 239 ■ Note at “PHY Configuration Overrides” on page 254
1.41		February 27, 2020	Added: ■ “Signal Overview” on page 29 ■ “Signal Descriptions for x1 Configuration” on page 33 ■ “Signal Descriptions for x2 Configuration” on page 33 ■ “Signal Descriptions for x4 Configuration” on page 31 Updated: ■ Copyright Notice and Proprietary Information ■ “Preface” on page 9 ■ “Product Overview” on page 13 ■ Figure “Transmit and Receive Datapaths” on page 26 ■ “Usage Model” on page 227 ■ “Simulation” on page 267 “pcs_*” instances have been updated to “pipe_*” throughout the databook
1.33_1		December 2, 2019	No technical updates

PCS Version	Databook Version	Date	Description (Continued)
1.33_1		August 16, 2019	Updated: ■ “Product Overview” on page 13 ■ Table “Power States Supported by the PCS” on page 15 ■ Table “pipe_laneX_pclk Frequencies” on page 24 ■ Table “pipe_laneX_max_pclk Frequencies” on page 25 ■ “Signal Descriptions” on page 31 ■ “Power State Transition Times” on page 245 ■ “Power-Gating Support” on page 260 ■ Table “Power Supplies” on page 260 ■ Table “Power State Table” on page 261
1.33		May 21, 2019	Added: ■ PCS version ■ “Tie-Off Values for Unused Lanes” on page 264 ■ Table “Tie-Off Values for Unused Lanes” on page 264 Updated: ■ Table “RTL Parameters Available for PCS Configuration” on page 17 ■ “Signal Descriptions” on page 31 ■ “PIPE (PCS and PHY) Constraints” on page 271 ■ Table “Hand-Instantiated Modules in PCIe 4.0 and SATA” on page 272
		March 6, 2019	Updated: ■ Copyright Notice and Proprietary Information ■ “General Product Description” on page 12 ■ Footnotes at table “Power States Supported by the PCS” on page 13 ■ Signal auto-extraction in “Signal Descriptions” on page 29 ■ Figure “Power-up Reset Timing” on page 188 ■ “PHY Bifurcation Support” on page 207 ■ Table “Lane Margining Parameters” on page 216
		April 20, 2018	■ “Product Overview” chapter: updated “Applications” section ■ “Signal Descriptions” chapter: updated “PCS Top-Level I/O Diagram”, “PIPE 4 Per-lane Interface Signals”, and “PIPE 4 Interface (Per-Lane TX/RX)” sections ■ “Usage Model” chapter: updated “Clocks”, “Power State Transition Times”, “PHY Bifurcation Support”, “Lane Disabling and Up/Down-Configuration”, and “Power-Gating Support” sections
		February 22, 2018	■ “Product Overview” chapter: updated “Features” section ■ “Signal Descriptions” chapter: updated “PCS Top-Level I/O Diagram”, “PIPE 4 Interface (Per-Lane TX/RX)”, “PHY Interface Signals (Shared)”, and “Miscellaneous Interface Signals (PHY Configuration)” sections ■ “Usage Model” chapter: updated “PHY Configuration Overrides” section

PCS Version	Databook Version	Date	Description (Continued)
		November 17, 2017	■ Preface: updated “Databook Organization” and “Reference Documentation” sections ■ “Product Overview” chapter: updated introduction, “Features”, “Applications”, and “Standards Compliance” sections ■ “Architecture” chapter: updated “Clocks” section ■ “Signal Descriptions” chapter: updated “PCS Top-Level I/O Diagram”, “PIPE 4 Interface Signals”, “PIPE 4 Per-lane Interface Signals”, “PIPE 4 Interface (Per-Lane TX/RX)”, “PHY Interface Signals (Shared)”, “PHY Interface (Per-Lane TX/RX)”, “Miscellaneous Interface Signals (Power Gating)”, “Miscellaneous Interface Signals (PHY Configuration)”, and “PIPE Wrapper Signal Descriptions” sections ■ “Usage Model” chapter: updated “Clocks”, “Datapath”, “Power State Transitions”, “PHY Bifurcation Support”, “PHY Aggregation Support”, “Transmitter Equalization Settings”, “Lane Margining at Receiver”, and “Power-Gating Support” sections; added “Clock Architecture and Link Selection” section ■ “Simulation” chapter: updated introduction
		June 19, 2017	Preliminary version: ■ “Preface” updated “Customer Support” section ■ “Signal Descriptions” chapter: updated “PCS Top-Level I/O Diagram”, “PIPE 4 Per-lane Interface Signals”, “PIPE 4 Interface (Per-Lane TX/RX)”, and “Miscellaneous Interface Signals (PHY Configuration)” sections ■ “Usage Model” chapter: added “Datapath”, “Power State Transition Times”, and “Lane Margining at Receiver” sections
		February 10, 2017	Preliminary version: ■ Preface: updated “Reference Documentation” section ■ “Product Overview” chapter: updated “Device Configuration”, “Applications”, and “Standards Compliance” sections; added “Device Parameters” section ■ “Signal Descriptions” chapter: updated “PCS Top-Level I/O Diagram”, “PIPE 4 Interface Signals”, “PIPE 4 Interface (Per-Lane TX/RX)”, “PHY Interface Signals (Shared)”, “PHY Interface (Per-Lane Configuration)”, “Miscellaneous Interface Signals (PHY Configuration)”, and “PIPE Wrapper Signal Descriptions” sections ■ “Usage Model” chapter: updated “Reference Clock Input” and “PIPE Lane Disable” sections; added “Lane Margining at Receiver”, “SATA BIST Loopback Mode”, “SATA - COM Align and Misalign Mechanism”, “SATA - Disconnect Detection”, and “SATA - Nominal Empty Buffer Mode Operation” sections
		December 6, 2016	Preliminary version: ■ “Architecture” chapter: updated “Transmit and Receive Datapaths” section ■ “Signal Descriptions” chapter: updated “PIPE 4 Per-lane Interface Signals” and “PIPE Wrapper Signal Descriptions” sections

PCS Version	Databook Version	Date	Description (Continued)
		December 5, 2016	Preliminary version: ■ “Architecture” chapter: updated “Clocks” and “Transmit and Receive Datapaths” sections ■ “Signal Descriptions” chapter: updated “PCS Top-Level I/O Diagram”, “PIPE 4 Per-lane Interface Signals”, and “PIPE 4 Interface (Per-Lane TX/RX)” sections ■ “Usage Model” chapter: updated “Reference Clock Input” section
		September 8, 2016	Preliminary version

Preface

This document describes the Synopsys DesignWare PCIe 4.0 PCS core (PCS), which provides a PIPE 4-compliant interface between a MAC and a Synopsys DesignWare Cores PCIe 4.0 PHY. The PCS, in conjunction with the DesignWare Cores PCIe 4.0 PHY, provides interface connectivity in a System-on-Chip (SoC).

Databook Organization

This databook is organized as follows:

- Chapter 1, "Product Overview", provides an introduction to the PCIe 4.0 PCS core and its features.
- Chapter 2, "Architecture", describes the PCIe 4.0 PCS core's architecture.
- Chapter 3, "Signal Overview", describes the organization of the Signal Description chapters of the PCIe 4.0 PCS core.
- Chapter 4, "Signal Descriptions for x4 Configuration", describes the top-level interface signals for the x4 configuration.
- Chapter 5, "Usage Model", provides simulation requirements and usage information for the PCIe 4.0 PCS core.
- Chapter 6, "Simulation", lists modules that must be processed correctly during simulation to avoid problems on the ASIC interface. This chapter includes ways to ensure that the simulation process is correct.
- Chapter 7, "Synthesis", describes the synthesis database and the PIPE constraints.

Web Resources

- DesignWare IP product information: https://www.synopsys.com/designware-ip.html
- Your custom DesignWare IP page: https://www.synopsys.com/dw/mydesignware.php
- Documentation through SolvNetPlus: https://solvnetplus.synopsys.com (Synopsys password required)
- Synopsys Common Licensing (SCL): https://www.synopsys.com/keys

Reference Documentation

- *PCI Express® Base Specification*, Revision 4.0, Version 1.0, September 27, 2017, PCI-SIG
- *PHY Interface For the PCI Express, SATA, and USB 3.1 Architectures*, Version 4.4.1, January 2017, Intel Corporation
- *PHY Interface For the PCI Express, SATA, and USB 3.1 Architectures*, Version 4.3, January 2014, Intel Corporation
- DesignWare Cores PCIe 4.0 PHY Databook, Synopsys, Inc.

Customer Support

Synopsys provides various methods for contacting Customer Support, as follows:

- *For fastest response*, enter a case through SolvNetPlus:

a. https://solvnetplus.synopsys.com

Note SolvNetPlus does not support Internet Explorer. Use a supported browser such as Microsoft Edge, Google Chrome, Mozilla Firefox, or Apple Safari.

b. Click the **Cases** menu and then click **Create a New Case** (below the list of cases).

c. Complete the mandatory fields that are marked with an asterisk and click **Save**.

Make sure to include the following:

- **Product L1:** DesignWare Cores
- **Product L2:** SERDES_16G_PHY

For more information about general usage information, refer to the following article in SolvNetPlus:

https://solvnetplus.synopsys.com/s/article/SolvNetPlus-Usage-Help-Resources

- Or, send an e-mail message to support_center@synopsys.com (your e-mail will be queued and then, on a first-come, first-served basis, manually routed to the correct support engineer):
  - Include the Product L1 and Product L2 names, process, and Version number in your e-mail so it can be routed correctly.
  - For simulation issues, include the timestamp of any signals or locations in waveforms that are not understood
- Or, telephone your local support center:
  - North America:

    Call 1-800-245-8005 from 7 AM to 5:30 PM Pacific time, Monday through Friday.
  - All other countries:

    https://www.synopsys.com/support/global-support-centers.html

1 Product Overview

The DesignWare PCIe 4.0 PCS core is a configurable Physical Coding Sublayer (PCS) that complies with the PIPE 4.4.1 and 4.3 specifications and supports the PCIe 4.0 and SATA protocol(s).

This chapter includes the following sections:

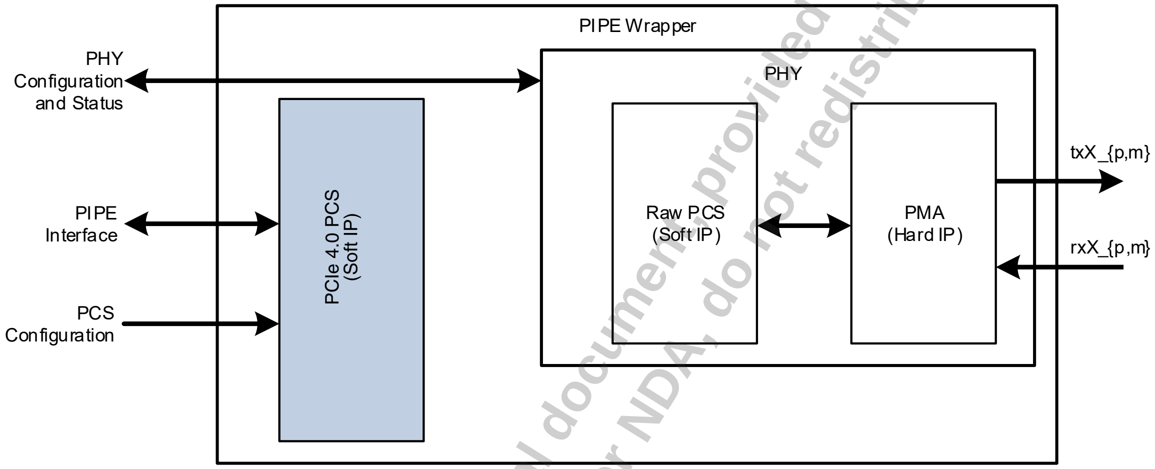
- "General Product Description" on page 14
- "Features" on page 15
- "Applications" on page 20
- "Standards Compliance" on page 20

1.1 General Product Description

The PCIe 4.0 PCS core provides many of the functions outlined in the PIPE 4 specification and comprises the PCS as referenced in the PIPE 4 specification.

The PCIe 4.0 PCS core can be used in conjunction with the Synopsys DesignWare Cores PCIe 4.0 PHY to create a PIPE4-compliant PHY solution. The following figure shows how the PCIe 4.0 PCS core fits in a PIPE4 PHY subsystem.

Figure 1-1 PCIe 4.0 PCS as Part of a PIPE4 PHY Subsystem



The top level—referred to as the PIPE wrapper— includes instantiation of the PCIe 4.0 PCS core and the PCIe 4.0 PHY. This databook describes the soft IP block highlighted (blue) in Figure 1-1.

The top-level inputs and outputs of the PIPE wrapper directly connect to the PCS and the PHY.

"Signal Overview" on page 29 provides the detailed mapping of these connections. The interface between the PCS and PHY is not described in this document.

The PCIe 4.0 PCS provides the following features:

- Standard PIPE 4 interface to upper layers of protocol stack (MAC layer and above)
- PCIe: 8b/10b (Gen 1/2) and 128b/130b (Gen3/4) encoding and decoding
- Translation of PIPE 4 power states (P0, P1, and so on) into the appropriate PHY controls, including any PHY-specific intermediate state sequencing
- Clock-rate compensation via SKP ordered sets and an elasticity buffer
- Error reporting
- Optional bifurcation of multi-lane PHYs into two or four independent PIPE 4 interfaces
- Optional aggregation to connect up to four PHYs.

1.2 Features

This section summarizes the functions that the PCIe 4.0 PCS supports.

1.2.1 Power States

The following table lists the supported power states and the impact of each state.

Table 1-1 Power States Supported by the PCS

Power State	Disabled Elements	Active Elements	Notes
PCIe4 Operation			
P0	N/A	All datapath and control	Fully-powered state
P0s	■ PHY CDR ■ PHY RX clocks	■ PHY TX Electrical Idle ■ PHY reference clock	N/A
P1	■ PHY RX clocks ■ PHY TX clocks	■ MPLLs and MPLL clocks ■ PHY reference clock	N/A
P2	■ PHY RX clocks ■ PHY TX clocks ■ PHY TX common-mode ■ PCLK	■ PHY reference clock	N/A
P1.CPM	■ PHY RX clocks ■ PHY TX clocks ■ PCLK ■ PHY reference clock can be disabled	■ PHY TX common mode	Equivalent to L1 Clock Power Management power state
P1.1[a]	■ PHY RX clocks ■ RX Electrical Idle Exit (EIE) Detector ■ PHY TX clocks ■ PCLK ■ PHY reference clock can be disabled	■ PHY TX common mode	Equivalent to PCIe L1.1 power state
P1.2[b]	■ PHY RX clocks ■ RX Electrical Idle Exit (EIE) Detector ■ PHY TX clocks ■ PHY TX common-mode ■ PCLK (pcs_lane0_pclk) ■ PHY reference clock can be disabled ■ If power gating is enabled, switchable core supply can be disabled in this state (for more information, "Power Gating" on page 16).	N/A	Equivalent to PCIe L1.2 power state

Table 1-1 Power States Supported by the PCS (Continued)

Power State	Disabled Elements	Active Elements	Notes
P2.CPM	■ PHY RX clocks ■ PHY TX clocks ■ PHY TX common-mode ■ PCLK ■ PHY reference clock can be disabled	N/A	Equivalent to PCIe L2 state with reference clock disabled
P2.NOBEACON	■ PHY RX clocks ■ RX Electrical Idle Exit (EiE) Detector ■ PHY TX clocks ■ PHY TX common-mode ■ PCLK ■ PHY reference clock can be disabled	N/A	Equivalent to PCIe L2 state with reference clock disabled and no beacon supported. The PCS disables the RX Electrical Idle Exit (EIE) Detector and PHY TX common-mode. The controller should not assert pipe_rxX_disable/pipe_txX_disable.
SLUMBER.PG	■ PHY RX clocks ■ RX Electrical Idle Exit (EiE) Detector ■ PHY TX clocks ■ PHY TX common-mode ■ PCLK ■ PHY reference clock can be disabled	N/A	The PCS disables the RX Electrical Idle Exit (EIE) Detector and PHY TX common-mode. The controller should not assert pipe_rxX_disable/pipe_txX_disable.

a. For PIPE specification 4.4.1, P1.1 is supported by setting the power state to P1.CPM. For more information about power states, see signal “pipe_laneX_powerdown” description at “Signal Overview” on page 29.

b. For PIPE specification 4.4.1, P1.2 is supported by setting the power state to P1.CPM. For more information about power states, see signal “pipe_laneX_powerdown” description at “Signal Overview” on page 29.

1.2.2 Power Gating

The PCIe 4.0 PCS supports gating of the core supply in a PCIe P1.2 state, where most of the logic is power-gated. A small portion of the design is in an always-on power domain and is needed to enter/exit the power-gated state. The PCIe 4.0 PCS also generates controls for the DesignWare Cores PCIe 4.0 PHY to gate its digital core voltage supplies.

1.2.3 Protocol Support

Each PCS lane can be independently configured to operate as follows:

- PCIe (1.1, 2.1, 3.0, and 4.0) modes.

However, the following restrictions exist:

- All PCS lanes connected to the same PHY must have baud rates that can be generated from the MPLL clock outputs of the same PHY. For example, lanes connected to the same PHY cannot operate in PCIe 1.1 (2.5 Gbps), PCIe 3.0 (8 Gbps), and SATA 6G (6 Gbps) simultaneously. That is, the protocols cannot be mixed.
- The pipe_lane0_protocol[1:0] input for configuring the protocol for each lane can be changed only when the PCIe 4.0 PCS is in Reset mode.

1.2.4 Bifurcation and Aggregation

You can configure the PCIe 4.0 PCS lanes to operate as part of one or multiple PCIe links (bifurcation). In addition, the PCIe 4.0 PCS can support connectivity for up to four PHYs (aggregation). There are multiple options available to configure the PCIe 4.0 PCS for aggregation and PCIe bifurcation. For more information about these features, see DWC_UPCS_PERLINK_CLK_ARCH parameter in "Device Configuration" on page 17, "PHY Aggregation Support" on page 248, and "PHY Bifurcation Support" on page 247.

Note Multiple PCS instances cannot be aggregated together to create a single link.

1.2.5 Device Configuration

This release requires you to manually edit the upcs/include/*e16_defaults.v file to change the configuration of the PCIe 4.0 PCS. The following table describes the RTL parameters available to change the configuration; you are permitted to use only these parameters.

Table 1-2 RTL Parameters Available for PCS Configuration

RTL Parameter	Description
IPNAME_TOP_PG_PINS	If defined, top-level pins for power are included. If not defined, no pins for power are included, but they might be inserted via the UPF flow. Default: Not defined
IPNAME_SEP_RESREF	If defined, a separate resref pin is provided per-phy: phyX_resref. If not defined, only one resref pin, phy_resref. is provided. Default: Not defined
DWC_PIPE_VER_4_4_1	If defined, the PCS supports PIPE specification 4.4.1. If not defined, the PCS supports PIPE specifications 4.2 and 4.3. **Note**: If you are using a Synopsys controller, do not change this parameter. Default: Defined

Table 1-2 RTL Parameters Available for PCS Configuration (Continued)

RTL Parameter	Description
DWC_PCIE_GEN4_32B_MODE	Configures the internal PCS datapath mode for operation at PCIe Gen4 data rate. This parameter enables a reduction of the maximum PCS operating frequency at the expense of a slight increase in the number of datapath registers and a slightly higher latency. When RTL parameter is defined: PCIe Gen4 internal datapath is 32 bits wide with a 500-MHz clock. When RTL parameter is not defined: PCIe Gen4 internal datapath is 16 bits wide with a 1,000-MHz clock. Default: Defined (Datapath is in 32-bit mode at 500 MHz.)
DWC_UPCS_PERLINK_CLK_ARCH	Enables Per-Link clock architecture. Per-Link clock architecture enables more efficient implementation of CTS in the PCS. If defined, the Per-Link clock architecture is implemented. If not defined the Per-Lane clock architecture is implemented. Default: Not defined
DWC_UPCS_NLINKS	Maximum number of links in which the PCS can be bifurcated. Valid values are 1, 2, and 4. Used when DWC_UPCS_PERLINK_CLK_ARCH is defined.
DWC_UPCS_NLINKS_GTR_1	Must be defined if DWC_UPCS_NLINKS = 2 or 4.
DWC_UPCS_NLINKS_GTR_2	Must be defined if DWC_UPCS_NLINKS = 4.
DWC_PCIE_ESTORE_MPS_OPT_EN	If defined, the maximum payload size related to area/latency optimizations in estore is enabled. If not defined, the estore is sized to support a maximum payload packet size of 4k bytes. Default: Not defined
DWC_PCIE_ESTORE_MPS_VALUE	If DWC_PCIE_ESTORE_MPS_OPT_EN is defined, this parameter defines the maximum payload size (in bytes) supported in the estore. If DWC_PCIE_ESTORE_MPS_OPT_EN is not defined, the maximum payload size supported in estore is 4k, regardless of this parameter. Default value: 4,096 Supported values: 128, 256, 512,1,024, 2,048, 4,096

1.2.6 Device Parameters

The following table describes system parameters that are required for configuring controllers.

Table 1-3 Device Parameters

Parameter	Description
NFTS	This parameter is used only for PCI Express and represents the number of Fast Training Sequences (FTSs) that are required to be advertised to the remote partner in order to successfully exit L0s/P0s. For a 100-MHz reference clock, the NFTS = 190. For a 25-MHz reference clock, the NFTS = 200.

1.3 Applications

The PCIe 4.0 PCS is designed to be used with the DesignWare Cores PCIe 4.0 PHY IP and to interface with the following controller(s):

- **PCIe4**: PIPE 4.4.1/4.3/4.2-compliant PCIe 1.1, 2.1, 3.0, or 4.0 controllers

The PCIe 4.0 PCS is designed to maximize scalability and configurability on a per-lane basis while minimizing the integration complexity with PHY and controller IP.

In addition, the PCIe 4.0 PCS supports a power-gating option for PCIe operation that is integrated with the PHY and suitable for extremely low-power applications with low exit latencies.

1.4 Standards Compliance

The PCIe 4.0 PCS is compliant to the PIPE 4.4.1/4.3 specification with the following exceptions:

- PIPE 4.4.1/4.3, Section 7.17: "…To support lane polarity inversion, the PHY must invert received data when RxPolarity is asserted. Inverted data must begin showing up on RxData[] within 20 PCLKs of when RxPolarity is asserted…."

  The PCIe 4.0 PCS requires more than 20 PCLK cycles for certain interface widths and data rates.

- PIPE 4.4.1/4.3, Section 6.1: "…PhyStatus… When this signal transitions during entry and exit from any PHY state where PCLK is not provided, then the signaling is asynchronous…"

  The PCIe 4.0 PCS does not generate the PhyStatus output when transitioning between two power states where the reference clock input to the PHY can be turned off. In PCIe operation for example, when transitioning between P1.CPM and P1.2, PhyStatus is not generated, because the PCS does not have any clock available when the reference clock is disabled.

- PIPE 4.3, Section 6.1: "…AsyncPowerChangeAck…Only used when transitioning between two power states without PCLK..." (supported in PIPE 4.4.1)

  The PCIe 4.0 PCS does not support this handshake, because in states where PCLK is disabled, the reference clock to the PHY can also be disabled. In absence of the reference clock, the PCIe 4.0 PCS cannot implement a handshake sequence.

- PIPE 4.4.1, Section 8.14 states rules for operating in Nominal Empty buffer mode. The rule stating, "All SKP symbols of SOS are removed(8b/10b SKP or 128/130 AA)" is not supported

  The following PIPE 4.4.1 Message Bus Address Space registers are not supported (they are optional):

  - PHY Register Address 2h: Elastic Buffer Control
  - MAC Register Address 3h: Elastic Buffer Status

In addition, note that the PCIe 4.0 PCS is backwards-compatible to the PIPE 4.2 specification with proprietary signal support for PCIe L1 substate entry and exit sequences.

2
Architecture

This chapter contains the following sections:

2.1 Resets

The PCIe 4.0 PCS provides the following reset inputs:

- Global reset: phy_reset
- Per-lane reset: pipe_laneX_reset_n (where X represents lane number)

The active-high phy_reset is a shared reset input that connects to the phy_reset inputs of all the PHYs connected to the PCIe 4.0 PCS. Assertion of phy_reset resets all registers in the PCIe 4.0 PCS.

The per-lane reset, pipe_laneX_reset_n, is an active-low signal that resets all registers in lane X. This reset is equivalent in functionality to the PIPE interface reset signal (Reset#) as defined in the PIPE specification. For lanes belonging to the same PIPE link, pipe_laneX_reset_n for the corresponding lanes should be driven from the common PIPE reset signal from the MAC.

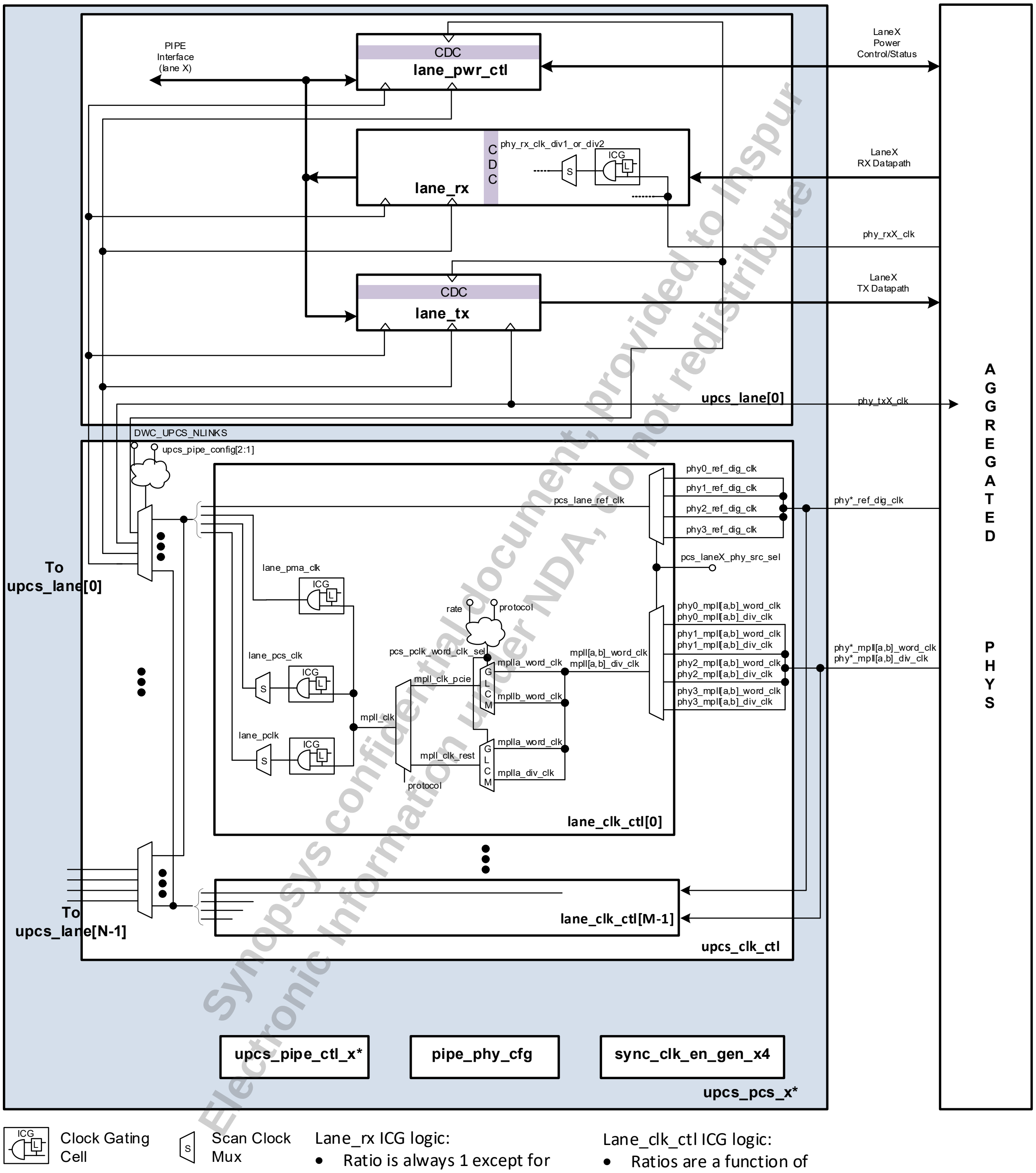
All internal registers are reset asynchronously with the assertion of reset inputs. In order to reset the internal registers, clock inputs to the PCIe 4.0 PCS do not have to be active.

PCIe: In addition to the primary reset inputs, the PCIe 4.0 PCS generates internal resets when Power-Gating mode is enabled (pg_mode_en = 1) and when the lanes are in a P1.2 power state. In this mode, when lane X enters a P1.2 power state, internal registers in lane X are reset in preparation for power to be disabled. In this mode, all registers in lane X, except the always ON logic (upcs_pipe_ctl_*), are reset. If all PCIe 4.0 PCS lanes connected to a particular PHY (PHY*N*, where N represents PHY number) are in a P1.2 power-gating state, the phyN_pg_reset output to PHY*N* is asserted, reseting the internal registers in the PHY. This action is performed in preparation for power to be disabled in the PHY.

2.2 Clocks

Figure 2-1 on page 23 shows the PCIe 4.0 PCS clock architecture. If the DWC_UPCS_PERLINK_CLK_ARCH parameter is not defined, the clock selection logic is implemented per lane. Each lane selects clock sources from the appropriate PHY (in the case of an aggregated PHY), then the lane_pclk, lane_pcs_clk, and lane_pma_clk signals are derived from MPLL clock sources of the selected PHY. If DWC_UPCS_PERLINK_CLK_ARCH is defined, the clock selection logic is implemented per link. Each link selects clocks sources from the appropriate PHY, then the link clocks are broadcast to all lanes that belong to that link—as a function of the bifurcation use mode selected by the application via upcs_pipe_config[2:1]. For information about the use modes, see "Clock Architecture and Link Selection" on page 246.

Figure 2-1 Clock Architecture



Lane_rx ICG logic:

- Ratio is always 1 except for PCIE Gen4 in 32bit mode (DWC_PCIE_GEN4_32B_MODE is set)
- This logic is always present although it is unnecessary for PCIE Gen4 in 16b mode (DWC_PCIE_GEN4_16B_MODE is set)

Lane_clk_ctl ICG logic:

- Ratios are a function of protocol, rate, PIPE interface width

The PCIe 4.0 PCS provides the following clock inputs:

- MPLL clock inputs from PHYs: phyN_mpll[a,b]_div_clk, phyN_mpll[a,b]_word_clk
- Reference clock inputs from PHYs: phyN_ref_dig_clk
- Receiver recovered clock input from PHY lanes: phy_rxX_clk

The MPLL clock inputs are used to derive the PCS datapath clocks and the pipe clock outputs (pipe_laneX_pclk, pipe_laneX_max_pclk) to the MAC. The reference clock input is used for power state and rate controls logic in the PCIe 4.0 PCS.

The PCIe 4.0 PCS provides the following clocks outputs:

- Per-lane PIPE PCLK: pipe_laneX_pclk
- Per-lane PIPE max PCLK: pipe_laneX_max_pclk

The pipe_laneX_pclk signal corresponds to the PCLK signal in the PIPE specification. The pipe_laneX_pclk outputs from all lanes configured to be part of the same link are phase-synchronous to each other, implying that pipe_laneX_pclk from one lane can be used to sample/launch synchronous PIPE interface signals for other lanes that are part of the same link. (For information about configuring for bifurcation, see "PHY Bifurcation Support" on page 247.)

The following table lists the pipe_laneX_pclk frequencies.

Table 2-1 pipe_laneX_pclk Frequencies

Protocol/Rate	pipe_laneX_pclk Frequency (MHz)		
	PIPE Interface Width: 8-bit	**PIPE Interface Width: 16-bit**	**PIPE Interface Width: 32-bit**
PCIe Gen1	250	125	62.5
PCIe Gen2	500	250	125
PCIe Gen3	N/A	500	250
PCIe Gen4	N/A	1,000	500

Lane X can be disabled as a result of down-configuration by asserting pipe_txX_elecidle and pipe_txX_compliance. In a lane-disabled state, the pipe_laneX_pclk output can remain enabled by setting the upcs_pipe_config[0] input to 1. Doing so enables pipe_laneX_pclk from the disabled lane X to be used for sampling/launching PIPE signals for other lanes in the same link. If DWC_UPCS_PERLINK_CLK_ARCH is defined, regardless of the upcs_pipe_config[0] setting, the pipe_laneX_pclk outputs for disabled lanes belonging to the same link remain enabled as long as one lane in the link is enabled.

For scenarios where the PIPE rate and PIPE interface width requires PCLK frequencies lower than the source MPLL clocks, the PCLK is generated by pulse-gating the MPLL source clocks. As a result in these cases, the duty cycle for the divided-down PCLK is not 50%. However, for implementation constraints, because the design is constrained to the highest frequency (that is, the highest rate and smallest interface width), the clock definitions in the synthesis constraints use a 50% duty cycle with the highest supported clock frequency.

The pipe_laneX_max_pclk signal corresponds to the MaxPCLK signal in the PIPE specification. The pipe_laneX_max_pclk output of all lanes that are part of the same link are phase-synchronous to each other. The following table provides the pipe_laneX_max_pclk frequencies for each protocol.

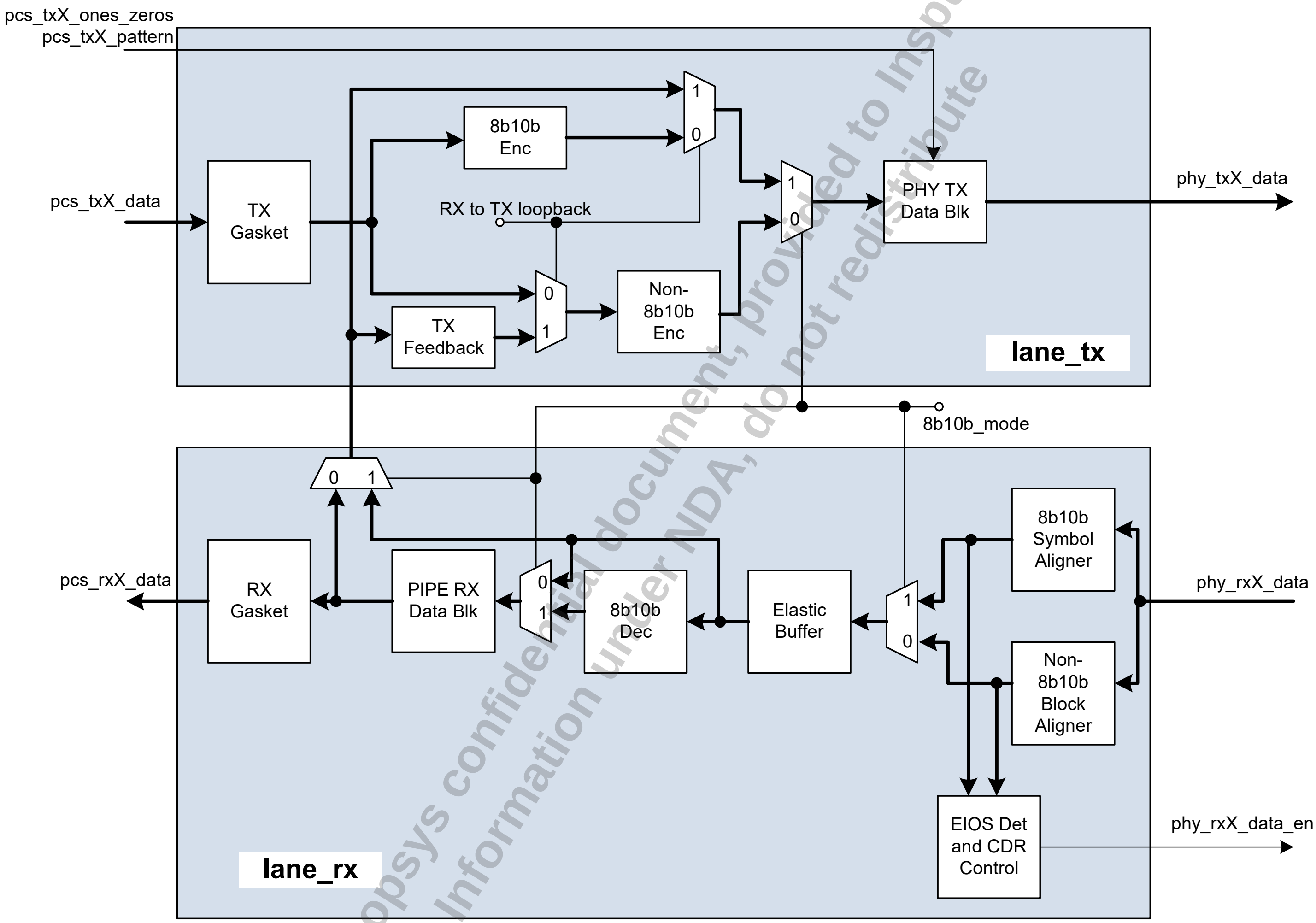
Table 2-2 pipe_laneX_max_pclk Frequencies

Protocol	Rate (MHz)
PCIe Gen1	500
PCIe Gen2	500
PCIe Gen3	1,000
PCIe Gen4	1,000

2.3 Transmit and Receive Datapaths

The following figure shows the transmit and receive datapaths within each lane.

Figure 2-2 Transmit and Receive Datapaths



The pipe_txX_data transmit data and controls are converted from the PIPE interface clock to the internal pcs_clk in the TX gasket. The fixed data-path-width data at the output of the TX gasket is encoded as follows:

- 8b10b-encoded for 8b10b protocols
- 128b-130b-encoded for PCIe Gen3/4

To support RX-to-TX loopback mode as specified in the PIPE specification, the RX recovered data is multiplexed into the TX datapath. Note that for 8b10b mode, the non-decoded 8b10b received data is sent to the transmitter. For non-8b10b data, the received data is decoded first; that is, the header is extracted from the data, then looped back into the TX datapath, where it is encoded before being transmitted to the PHY. The encoded TX data goes through override logic where fixed patterns are inserted into the TX data-stream controller by the pipe_txX_ones_zeros and pipe_txX_pattern[1:0] PIPE signals.

Data received from the PHY is processed to find symbol alignment for 8b10b data and block alignment for non-8b10b data. The aligned data is monitored for EIOS ordered sets (PCIe) to detect end-of-packets and to infer entry into electrical idle condition. After detecting EIOSs, the PCIe 4.0 PCS de-asserts phy_rxX_data_en to the PHY lane, causing the lane CDR to stop monitoring input data transitions and to switch to reference clock tracking mode. The aligned data is passed to the elastic buffer, where compensation for frequency drift is performed by adding or removing:

- SKP symbols (PCIe)

The frequency-compensated data is 8b10b-decoded for 8b10b data, then passed through the PIPE RX processing block where data and status signals are cycle-aligned. The RX data and status signals pass through the RX gasket that maps the internal fixed-width data to variable interface widths supported by the PIPE interface.

3
Signal Overview

3.1 PCS Signal Descriptions Overview

3.1.1 Signal Naming Conventions

Abstract signal names are used in this databook for convenience. Special notation enables one abstract signal name to represent a set of literal signals.

As an example, consider the signal names, pcs_laneX_clkreq_n and phyN_mplla_div_clk:

- X denotes lane number (0,1,2,3...15).

  This range is represented as (for X=0; X <=Nlanes-1).

  Per-lane pins are replicated for each lane in the PCIe 4.0 PCS.
- N denotes PHY number (0,1,2,3).

  This range is represented as (for N=0; N <=Nphy-1).
- P denotes consistency and reduces redundancy in the description of signals with the protocol prefix.

  This range is represented as (for P=0; P <=2).

3.1.2 Signal Descriptions per Configuration

Most signal descriptions are identical across all Signal Description chapters. However, depending on the configuration, specific signals will have different bus widths.

Use the following links for the correct bus width for each configuration:

- "Signal Descriptions for x4 Configuration" on page 31

4
Signal Descriptions for x4 Configuration

This chapter details all I/O signals in the IP. Inputs are on the left of the signal diagrams; outputs are on the right. In addition to describing the function of each signal, the signal descriptions in this chapter might include the following information:

Active State: Indicates whether the signal is active high or active low. When a signal is not intended to be used in a particular application, then this signal needs to be tied or driven to the inactive state (opposite of the active state).

Registered: Indicates whether or not the signal is registered directly inside the IP boundary without intervening logic (excluding simple buffers). A value of No does not imply that the signal is not synchronous, only that there is some combinatorial logic between the signal's origin or destination register and the boundary of the IP. A value of N/A indicates that this information is not provided for this IP title.

Synchronous to: Indicates which clocks in the IP sample this input (drive for an output). This clock might not be the same as the clock that your application logic should use to clock (sample/drive) this pin. For more details, consult the clock section in the databook.

The I/O signals are grouped as follows:

- PCS Configuration Signals on page 33
- PIPE Interface Signals on page 37
- MPLL Control Signals on page 51
- Reference Clock Control Signals on page 57
- Power Related Signals on page 60
- Analog I/O Signals on page 63
- Receiver Signals on page 65
- Transmitter Signals on page 67
- PCS Scan Interface Signals on page 68
- PHY Scan Interface Signals on page 72
- JTAG Port Signals on page 101
- Boundary Scan Port Signals on page 103
- External Protocol Override Signals on page 109

- Parallel Control Register Port Signals on page 219
- SRAM Control Signals on page 221
- PHY Test Signals on page 223
- Resistor Tune Signals (Common Block Group) on page 225

4.1 PCS Configuration Signals

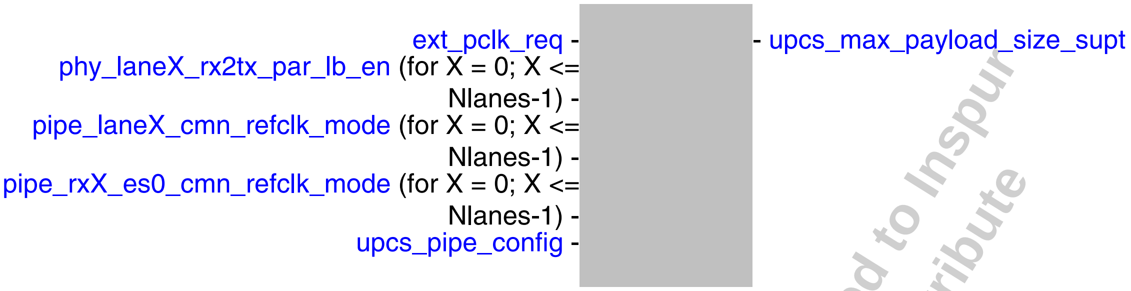

ext_pclk_req -
phy_laneX_rx2tx_par_lb_en (for X = 0; X <= Nlanes-1) -
pipe_laneX_cmn_refclk_mode (for X = 0; X <= Nlanes-1) -
pipe_rxX_es0_cmn_refclk_mode (for X = 0; X <= Nlanes-1) -
upcs_pipe_config -
- upcs_max_payload_size_supt

Table 4-1 PCS Configuration Signals

Port Name	I/O	Description
ext_pclk_req	I	External PCLK request. When asserted, the MPLL clock sources in the PHY are powered up and pipe_laneX_pclk outputs stay active, regardless of the pipe_laneX_powerdown[3:0] inputs. **Synchronous To:** Asynchronous **Active State:** High
phy_laneX_rx2tx_par_lb_en (for X = 0; X <= Nlanes-1)	I	Parallel (RX to TX) loopback enable. When this signal is asserted, recovered parallel data from the receiver is looped back to the transmit serializer. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
pipe_laneX_cmn_refclk_mode (for X = 0; X <= Nlanes-1)	I	When this signal is asserted, common reference clock settings are used for MPLL bandwidth inputs to the PHY. Otherwise SRNS/SRIS settings are used. **Note:** Any change to this input must be followed by phy_reset assertion. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-1 PCS Configuration Signals (Continued)

Port Name	I/O	Description
pipe_rxX_es0_cmn_refclk_mode (for X = 0; X <= Nlanes-1)	I	Elastic buffer mode for Nominal Half Full Buffer mode (Elasticity Buffer Mode is 0). This signal is "Don't care" during Nominal Empty Buffer mode (Elasticity Buffer Mode is 1). All lanes in a link must be in the same mode. ■ 1'b0: Nominal Half Full Buffer mode ■ 1'b1: Common Reference Clock mode **Note:** Any change to this input must be followed by phy_reset assertion. **Synchronous To:** Asynchronous **Active State:** N/A
upcs_max_payload_size_supt[2:0]	O	Maximum payload size supported. ■ This signal encodes what the maximum payload size is supported. ■ Currently, the 16G PCS supports a maximum 4096 bytes. For information about configuration parameters that impact the maximum payload size, see the "Device Configuration" section. ■ 3'b000: 128 bytes ■ 3'b001: 256 bytes ■ 3'b010: 512 bytes ■ 3'b011: 1,024 bytes ■ 3'b100: 2,046 bytes ■ 3'b101: 4,096 bytes **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-1 PCS Configuration Signals (Continued)

Port Name	I/O	Description
upcs_pipe_config[15:0]	I	PCS PIPE configuration. ■ When upcs_pipe_config[0] is set to 1, the PCS ignores lane-off via PIPE specification method (TxElecIdle = 1 and TxCompliance = 1) and responds to power-down/rate/width changes. ■ Otherwise, until the MAC de-asserts the "turned off" signaling, the PCS ignores any commands to change powerdown/rate/width after being turned off. ■ If the DWC_UPCS_PERLINK_CLK_ARCH parameter is defined, upcs_pipe_config[2:1] determines the bifurcation use mode for the PCS. The mapping between bits [2:1] and bifurcation use modes is as follows (N indicates total number of PCS lanes): ■ 2'b00: xN (example: x8 for N = 8) ■ 2'b01: xN/2xN/2 (example: x4x4 for N = 8) ■ 2'b10: xN/4xN/4xN/4xN/4 (example: x2x2x2x2 for N = 8) ■ 2'b11: Reserved ■ upcs_pipe_config[15:3] are reserved, tie these bits to 0. When DWC_UPCS_NLINKS = 1, 2'b00 is the only valid value. When DWC_UPCS_NLINKS = 2, valid values are 2'b00 and 2'b01. When DWC_UPCS_NLINKS = 4, valid values are 2'b00, 2'b01, and 2'b10. **Note:** Any change to this input must be followed by phy_reset assertion. **Synchronous To:** Asynchronous **Active State:** N/A

4.2 PIPE Interface Signals (for X = 0; X <= Nlanes-1)

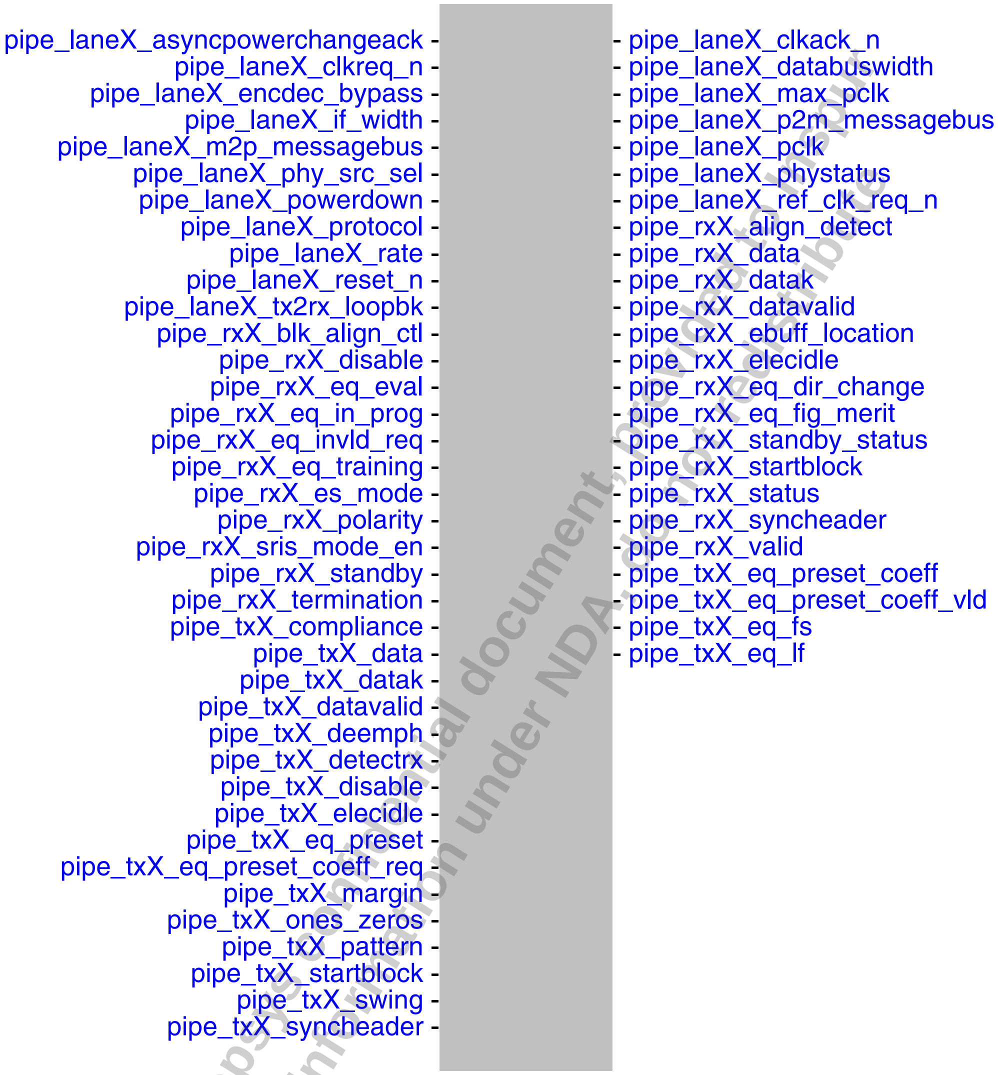


Table 4-2 PIPE Interface Signals (for X = 0; X <= Nlanes-1)

Port Name	I/O	Description
pipe_laneX_asyncpowerchangeack	I	Power change acknowledgment for lane X; PIPE signal "AsyncPowerChangeAck" (Sec 6.1, PIPE 4.4.1/4.3). ■ Used when transitioning between two power states without PCLK. ■ After the PHY asserts PhyStatus to acknowledge the power state change, the MAC responds by asserting pipe_laneX_asyncpowerchangeack until it samples PhyStatus de-asserted. ■ If controller does not support this signal, tie this input to the pipe_laneX_phystatus output of the PCS. **Note:** For PIPE 4.3, this signal is not used-tie it to 1'b0. **Synchronous To:** Asynchronous **Active State:** High
pipe_laneX_clkack_n	O	Clock acknowledge for lane X. PCIe: ■ This a side-band signal needed for the PIPE 4.2 controller to enter and exit P1.CPM, P1.1, and P1.2 power states. ■ For information about using this signal, see the "PCIe L1 Substate Transitions" section. ■ When integrating with a PIPE 4.4.1/4.3 controller, this output can be ignored. SATA: ■ This a side-band signal needed for the PIPE 4.2 controller to enter and exit P1.CPM, P1.1, and P1.2 power states. ■ For information about using this signal, see the "PCIe L1 Substate Transitions" section. ■ When integrating with a PIPE 4.3 controller, this output can be ignored. **Synchronous To:** Asynchronous **Active State:** Low

Table 4-2 PIPE Interface Signals (for X = 0; X <= Nlanes-1) (Continued)

Port Name	I/O	Description
pipe_laneX_clkreq_n	I	Clock request for lane X. PCIe: ■ This a side-band signal that a PIPE 4.2 controller needs to enter and exit P1.CPM, P1.1, and P1.2 power states. ■ For information about using this signal, see the "PCIe L1 Substate Transitions" section. ■ When integrating with a PIPE 4.4.1/4.3-compliant controller, set this input to an asserted value of 0. SATA: ■ This a side-band signal that a PIPE 4.2 controller needs to enter and exit P1.CPM, P1.1, and P1.2 power states. ■ For information about using this signal, see the "PCIe L1 Substate Transitions" section. **Synchronous To:** Asynchronous **Active State:** Low
pipe_laneX_databuswidth[1:0]	O	Bus width configuration for lane X; PIPE signal "DataBusWidth[1:0]" (Sec 6.1, PIPE 4.4.1/4.3). This field reports the width of the data bus configured for the PHY. **Synchronous To:** Asynchronous **Active State:** N/A
pipe_laneX_encdec_bypass	I	Encode and decode bypass for lane X; PIPE signal "EncodeDecodeBypass" (Sec 6.1, PIPE 4.4.1/4.3). Bypasses the 8b10 and block encoding/decoding in the 16G PCS. **Note:** ■ In PCIe operation, the current PCS implementation does not support pipe_laneX_encdec_bypass; therefore, tie this signal to 0. ■ Any change to this input must be followed by phy_reset assertion. **Synchronous To:** Asynchronous **Active State:** High
pipe_laneX_if_width[1:0]	I	PIPE interface width for lane X; PIPE signal "Width" (Sec 6.1, PIPE 4.4.1/4.3). Controls the data width of pipe_txX_data and pipe_rxX_data buses. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A

Table 4-2 PIPE Interface Signals (for X = 0; X <= Nlanes-1) (Continued)

Port Name	I/O	Description
pipe_laneX_m2p_messagebus[7:0]	I	Mac-to-PHY Message Bus for lane X PIPE signal "M2P_MessageBus" (Sec 6.1, PIPE 4.4.1/4.3). ■ MAC-driven register bus interface (RBI). ■ PIPE 4.4.1 supports all message bus commands. For PIPE 4.3, the supported command mapping is as follows: ■ NOP ■ Write-committed, non-posted ■ write_ack For more information, see the "Lane Margining at Receiver" section. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A
pipe_laneX_max_pclk	O	Maximum PCLK; PIPE signal "Max PCLK" (Sec 6.1, PIPE 4.4.1/4.3). ■ The frequency of this output clock is the maximum frequency supported for that protocol. ■ Contrary to what is indicated in the PIPE 4.4.1/4.3 specification, the frequency does not change with rate. ■ For information about the supported frequencies, see table "pipe_laneX_pclk Frequencies". **Synchronous To:** N/A **Active State:** N/A
pipe_laneX_p2m_messagebus[7:0]	O	PHY-to-Mac Message Bus for lane X PIPE signal "P2M_MessageBus" (Sec 6.1, PIPE 4.4.1/4.3). ■ PHY-driven register bus interface (RBI). ■ 4.4.1 supports all message bus commands. For PIPE 4.3, the supported command mapping is as follows: ■ NOP ■ Write-committed, non-posted ■ write_ack For more information, see the "Lane Margining at Receiver" section. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A
pipe_laneX_pclk	O	Parallel clock for lane X; PIPE signal "PCLK" (Sec 6.1, PIPE 4.4.1/4.3). ■ This signal is the parallel interface differential data clock. ■ The pipe_laneX_clk outputs from all lanes that are part of the same link are phase-synchronous; that is, pipe_laneX_clk from one lane can be used to sample and drive signals on the other lanes. ■ For a list of pipe_laneX_pclk frequencies, see table "pipe_laneX_pclk Frequencies". **Synchronous To:** N/A **Active State:** N/A

Table 4-2 PIPE Interface Signals (for X = 0; X <= Nlanes-1) (Continued)

Port Name	I/O	Description
pipe_laneX_phy_src_sel[1:0]	I	PHY source select for lane X. ■ When the PCS is connected to aggregated PHYs, the clock source for lane X can come from different PHYs based on bifurcation configuration. ■ For information about aggregating PHYs, see the "PHY Aggregation Support" section. **Note:** Any change to this input must be followed by phy_reset assertion. **Synchronous To:** Asynchronous **Active State:** N/A
pipe_laneX_phystatus	O	PHY status for lane X; PIPE signal "PhyStatus" (Sec 6.1, PIPE 4.4.1/4.3). Acknowledgment signal for the following control inputs: ■ pipe_laneX_if_width[1:0] ■ pipe_laneX_powerdown[3:0] ■ pipe_laneX_rate[1:0] ■ pipe_rxX_eq_eval ■ pipe_txX_detectrx (when pipe_laneX_powerdown[3:0] = P1) **Active State:** High **Synchronous To:** pipe_laneX_pclk with the following exceptions: ■ when lane reset is asserted it is asynchronously asserted ■ when transitioning from a state with pclk to a state without pclk, this signal is asserted synchronously, then de-asserted asynchronously ■ when transitioning from a state without pclk to a state with pclk, this signal is asserted asynchronously, then de-asserted synchronously

Table 4-2 PIPE Interface Signals (for X = 0; X <= Nlanes-1) (Continued)

Port Name	I/O	Description
pipe_laneX_powerdown[3:0]	I	Power state for lane X; PIPE signal "PowerDown[3:0]" (Sec 6.1, PIPE 4.4.1/4.3). Controls the PHY power states. The power-state mapping is as follows: **PCIe: PIPE 4.4.1 - PIPE 4.3** ■ P0: 4'b0000 - P0: 4'b0000 ■ P0s: 4'b0001 - P0s: 4'b0001 ■ P1: 4'b0010 - P1: 4'b0010 ■ P2: 4'b0011 - P2: 4'b0011 ■ P1.CPM: 4'b0100a - P1.CPM: 4'b0100b ■ P1.1: 4'b0100a - P1.1: 4'b0101 ■ P1.2: 4'b0100a - P1.2: 4'b1100 ■ P2.CPM: 4'b1101 - P2.CPM: 4'b1101 ■ P2.NOBEACON: 4'b1111 - N/A - RxEIDetectDisable and TxCommonModeDisable signals were added in PIPE 4.4.1. - The P1.CPM power-down encoding is used along with RxEIDetectDisable and TxCommonModeDisable to encode P1.1 and P1.2. SATA: ■ Active: 4'b0000 ■ Partial: 4'b0001 ■ Slumber: 4'b0010 ■ DevSleep: 4'b0011 ■ Slumber.PG: 4'b1111 For information about power state transitions, see the "Power State Transitions" section. **Synchronous To:** pipe_laneX_pclk unless exiting a power-down state where pclk has been removed; then treat this signal as asynchronous. **Active State:** N/A
pipe_laneX_protocol[1:0]	I	Protocol/mode for lane X; PIPE signal "PHY Mode[1:0]" (Sec 6.1, PIPE 4.4.1/4.3). ■ Selects the PHY operating mode. ■ For PCIe operation, set to 2'b00. ■ For SATA operation, set to 2'b10. **Note:** Any change to this input must be followed by phy_reset assertion. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-2 PIPE Interface Signals (for X = 0; X <= Nlanes-1) (Continued)

Port Name	I/O	Description
pipe_laneX_rate[1:0]	I	Protocol/mode for lane X; PIPE signal "Rate[1:0]" (Sec 6.1, PIPE 4.4.1/4.3). Controls the link signaling rate. PCIe: ■ 2.5 Gbps: 2'b00 ■ 5 Gbps: 2'b01 ■ 8 Gbps: 2'b10 ■ 16 Gbps: 2'b11 SATA: ■ 1.5 Gbps: 2'b00 ■ 3 Gbps: 2'b01 ■ 6 Gbps: 2'b10 **Synchronous To:** pipe_laneX_pclk **Active State:** N/A
pipe_laneX_ref_clk_req_n	O	Reference clock request for lane X. This a side-band signal that the following PIPE controllers need to disable the reference clock input to the PHY: ■ PCIe: 4.4.1/4.3/4.2 ■ SATA: 4.3/4.2 ■ The reference clock input to the PHY can be disabled when pipe_laneX_clkreq_n = 1 and pipe_laneX_ref_clk_req_n = 1 (acknowledge signal) for all lanes connected to the PHY. ■ For information about using this signal, see "Reference Clock Input" and "PCIe L1 Substate Transitions" sections. **Synchronous To:** Asynchronous **Active State:** Low
pipe_laneX_reset_n	I	Reset for lane X; PIPE signal "Reset#" (Sec 6.1, PIPE 4.4.1/4.3). Resets the transmitter and receiver. **Synchronous To:** Asynchronous **Active State:** Low
pipe_laneX_tx2rx_loopbk	I	TX-to-RX loopback enable for lane X. ■ When asserted, this input turns on the TX-to-RX serial loopback within the PHY. ■ Assertion of this signal in SATA mode changes the COM detection and locking mechanism. ■ This behavior is described in the "SATA BIST Loopback Mode" section. ■ This signal is for debug purposes only. **Synchronous To:** pipe_laneX_pclk **Active State:** High

Table 4-2 PIPE Interface Signals (for X = 0; X <= Nlanes-1) (Continued)

Port Name	I/O	Description
pipe_rxX_align_detect	O	RX ALIGN symbol detected for lane X; PIPE 4.3 signal "AlignDetect" (Sec 6.1, PIPE 4.3). Indicates receiver detection of an Align. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_rxX_blk_align_ctl	I	RX block align control for lane X; PIPE signal "BlockAlignControl" (Sec 6.1, PIPE 4.4.1/4.3). Controls whether the PHY performs block alignment. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_rxX_data[39:0]	O	RX data for lane X; PIPE signal "RxData[31:0]" (Sec 6.1, PIPE 4.4.1/4.3) RX parallel data. The valid data bits for pipe_rx_data[39:0] are as follows: **pipe_laneX_encdec_bypass - pipe_laneX_if_width[1:0] - Valid bits of pipe_rxX_data[39:0]** ■ 0 - 00 - [7:0] ■ 0 - 01 - [15:0] ■ 0 - 10 - [31:0] ■ 1 - 00 - [9:0] ■ 1 - 01 - [19:0] ■ 1 - 10 - [39:0] **Synchronous To:** pipe_laneX_pclk **Active State:** N/A
pipe_rxX_datak[3:0]	O	RX data control for lane X; PIPE signal "RxDataK[3:0]" (Sec 6.1, PIPE 4.4.1/4.3). Data/control for the symbols of received data. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A
pipe_rxX_datavalid	O	RX data valid; PIPE signal "RxDataValid" (Sec 6.1, PIPE 4.4.1/4.3). RX data valid to throttle RX data for non-8b10b encoding modes. **Synchronous To:** pipe_laneX_pclk **Active State:** High

Table 4-2 PIPE Interface Signals (for X = 0; X <= Nlanes-1) (Continued)

Port Name	I/O	Description
pipe_rxX_disable	I	RX disable control for lane X. ■ This a side-band signal that a PIPE 4.4.1/4.24.2 controller needs to enter and exit P1.1 and P1.2 power states. ■ When asserted, all RX lane circuitry (including RX Electrical Idle Exit Detection) for lane X is disabled. ■ When integrating with a PIPE 4.4.1/4.3 controller, pipe_rxX_disable must be set to 0. For P1.1 and P1.2 entry and exit sequences, see "PCIe L1 SubstateTransitions" section. **Synchronous To:** Asynchronous **Active State:** High
pipe_rxX_ebuff_location[8:0]	O	Entries in elastic buffer for lane X; PIPE signal "ElasticBufferLocation[8:0]" (Sec 6.1, PIPE 4.4.1/4.3). Encodes the number of entries currently in the elastic buffer. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A
pipe_rxX_elecidle	O	RX electrical idle detection for lane X; PIPE signal "RxElecIdle" (Sec 6.1, PIPE 4.4.1/4.3). Indicates receiver detection of an electrical idle. **Synchronous To:** Asynchronous **Active State:** High
pipe_rxX_eq_dir_change[5:0]	O	Direction change commands for link partner's TX equalization coefficients for lane X; PIPE signal "LinkEvaluationFeedbackDirectionChange[5:0]" (Sec 6.1, PIPE 4.4.1/4.3). Provides link equalization evaluation feedback in the direction change format. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A
pipe_rxX_eq_eval	I	RX equalization enable for lane X; PIPE signal "RxEqEval" (Sec 6.1, PIPE 4.4.1/4.3). When the MAC asserts this signal, the PHY starts evaluation of the far-end transmitter TX EQ settings. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_rxX_eq_fig_merit[7:0]	O	Figure of Merit for equalization settings of lane X; PIPE signal "LinkEvaluationFeedbackFigureMerit[7:0]" (Sec 6.1, PIPE 4.4.1/4.3). Provides the PHY link equalization evaluation Figure of Merit value. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A

Table 4-2 **PIPE Interface Signals (for X = 0; X <= Nlanes-1) (Continued)**

Port Name	I/O	Description
pipe_rxX_eq_in_prog	I	RX equalization in progress for lane X; PIPE signal "RxEqInProgress" (Sec 6.1, PIPE 4.4.1/4.3). This input drives phy_rxX_adapt_in_prog. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_rxX_eq_invld_req	I	RX equalization invalid request; PIPE signal "InvalidRequest" (Sec 6.1, PIPE 4.4.1/4.3). This input is ignored by the PCS and PHY. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_rxX_eq_training	I	RX equalization training mode enable for lane X; PIPE signal "RxEqTraining" (Sec 6.1, PIPE 4.4.1/4.3). Reserved. Tie to 1'b0. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_rxX_es_mode	I	Elastic buffer mode for lane X; PIPE signal "Elasticity Buffer Mode" (Sec 6.1, PIPE 4.4.1/4.3). Selects Elasticity Buffer operating mode. ■ 2'b00 (PCIe only): Nominal Half Full Buffer mode ■ 2'b01: Nominal Empty Buffer mode ■ 2'b10 (PCIe only): Common Reference Clock mode **Note:** ■ Any change to this input must be followed by phy_reset assertion. ■ When operating in SATA mode, tie this input to 2'b01. **Synchronous To:** Asynchronous **Active State:** N/A
pipe_rxX_polarity	I	RX polarity inversion enable for lane X; PIPE signal "RxPolarity" (Sec 6.1, PIPE 4.4.1/4.3). Requests the PHY to perform a polarity inversion on the received data. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_rxX_sris_mode_en	I	RX SRIS mode enable for lane X. When asserted, this input configures the PHY CDR and the elastic buffer to recover Independent Spread Spectrum Data. **Note:** Any change to this input must be followed by phy_reset assertion. **Synchronous To:** Asynchronous **Active State:** High

Table 4-2 PIPE Interface Signals (for X = 0; X <= Nlanes-1) (Continued)

Port Name	I/O	Description
pipe_rxX_standby	I	RX standby enable for lane X; PIPE signal "RxStandby" (Sec 6.1, PIPE 4.4.1/4.3). When asserted, the RX CDR for lane X is disabled. **Note:** ■ During an OOB signaling period in SATA mode, set this signal to 1. ■ Doing so disables RX CDR, so incoming OOB signals are not treated as valid data. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_rxX_standby_status	O	RX standby status for lane X; PIPE signal "RxStandbyStatus" (Sec 6.1, PIPE 4.4.1/4.3). When asserted, indicates that the RX has entered Standby state in response to pipe_rxX_standby assertion. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_rxX_startblock	O	RX start block indicator for lane X; PIPE signal "RxStartBlock" (Sec 6.1, PIPE 4.4.1/4.3). Enables PHY to provide MAC with the starting byte for a 128b block. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_rxX_status[2:0]	O	Receive status and error codes for lane X; PIPE signal "RxStatus[2:0]" (Sec 6.1, PIPE 4.4.1/4.3). Encodes receiver status and error codes for the received datastream when receiving data. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A
pipe_rxX_syncheader[3:0]	O	RX SYNC header for lane X; PIPE signal "RxSyncHeader[3:0]" (Sec 6.1, PIPE 4.4.1/4.3). PCIe: ■ Provides the SYNC header for the PHY to use in the next block. ■ The pipe_rxX_syncheader[1:0] bits are valid for PCIe. SATA: Reserved; leave floating. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A
pipe_rxX_termination	I	RX termination enable for lane X; PIPE signal "RX Termination" (Sec 6.1, PIPE 4.4.1/4.3). When asserted, the RX terminations are enabled. **Synchronous To:** Asynchronous **Active State:** High

Table 4-2 PIPE Interface Signals (for X = 0; X <= Nlanes-1) (Continued)

Port Name	I/O	Description
pipe_rxX_valid	O	RX valid for lane X; PIPE signal "RxValid" (Sec 6.1, PIPE 4.4.1/4.3). Indicates symbol lock and valid data on pipe_rxX_data and pipe_rxX_datak. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_txX_compliance	I	TX compliance for lane X; PIPE signal "TxCompliance" (Sec 6.1, PIPE 4.4.1/4.3). Sets the running disparity to negative. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_txX_data[39:0]	I	TX data for lane X; PIPE signal "TxData[31:0]" (Sec 6.1, PIPE 4.4.1/4.3). TX parallel data. The valid data bits for pipe_tx_data[39:0] are as follows: **pipe_laneX_encdec_bypass - pipe_laneX_if_width[1:0] - Valid bits of pipe_txX_data[39:0]** ■ 0 - 00 - [7:0] ■ 0 - 01 - [15:0] ■ 0 - 10 - [31:0] ■ 1 - 10 - [9:0] ■ 1 - 01 - [19:0] ■ 1 - 10 - [39:0] **Synchronous To:** pipe_laneX_pclk **Active State:** N/A
pipe_txX_datak[3:0]	I	TX data control for lane X; PIPE signal "TxDataK[3:0]" (Sec 6.1, PIPE 4.4.1/4.3). Data/control for the symbols of transmit data. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A
pipe_txX_datavalid	I	TX data valid; PIPE signal "TxDataValid" (Sec 6.1, PIPE 4.4.1/4.3). TX data valid to throttle TX data for non-8b10b encoding modes. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_txX_deemph[17:0]	I	TX equalization settings; PIPE signal "TxDeemph[17:0]" (Sec 6.1, PIPE 4.4.1/4.3). ■ Selects TX de-emphasis. ■ For information about TX de-emphasis, see the "Transmitter Equalization Settings" section. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A

Table 4-2 PIPE Interface Signals (for X = 0; X <= Nlanes-1) (Continued)

Port Name	I/O	Description
pipe_txX_detectrx	I	Receiver detect enable; PIPE signal "TxDetectRx/Loopback" (Sec 6.1, PIPE 4.4.1/4.3). Instructs the PHY to either begin a receiver detection operation, to begin loopback, or to signal LFPS during P0 for USB Polling state. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_txX_disable	I	This signal is transmitter disable. ■ This a side-band signal that a PIPE 4.4.1/4.24.2 controller needs to enter and exit P1.2 power state. ■ When asserted, the TX common mode for lane X is disabled. ■ When integrating with a PIPE 4.4.1/4.3 controller, pipe_txX_disable must be set to 0. ■ For P1.2 entry and exit sequences, see the "PCIe L1 Substate Transitions" section. **Synchronous To:** Asynchronous **Active State:** High
pipe_txX_elecidle	I	TX electrical idle enable for lane X; PIPE signal "TxElecIdle" (Sec 6.1, PIPE 4.4.1/4.3). When asserted (except during loopback), forces TX output to electrical idle. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_txX_eq_preset_coeff[17:0]	O	TX equalization preset coefficients for lane X; PIPE signal "LocalTxPresetCoefficients[17:0]" (Sec 6.1, PIPE 4.4.1/4.3). These signals are TX coefficients that correspond to the pipe_txX_eq_preset[3:0] input and are valid when pipe_txX_eq_preset_coeff_vld is asserted. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A
pipe_txX_eq_preset_coeff_vld	O	TX equalization preset coefficient valid for lane X; PIPE signal "LocalTxCoefficientsValid" (Sec 6.1, PIPE 4.4.1/4.3). When this signal is asserted, pipe_txX_eq_preset_coeff[17:0] is valid. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_txX_eq_fs[5:0]	O	TX equalization FS value for lane X; PIPE signal "LocalFS[5:0]" (Sec 6.1, PIPE 4.4.1/4.3). Provides the FS value for the PHY. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A

Table 4-2 PIPE Interface Signals (for X = 0; X <= Nlanes-1) (Continued)

Port Name	I/O	Description
pipe_txX_eq_lf[5:0]	O	TX equalization LF value for lane X; PIPE signal "LocalLF[5:0]" (Sec 6.1, PIPE 4.4.1/4.3). Provides the LF value for the PHY. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A
pipe_txX_eq_preset[4:0]	I	TX equalization preset for lane X; PIPE signal "LocalPresetIndex[3:0]" (Sec 6.1, PIPE 4.4.1/4.3). ■ Index for local PHY preset coefficients requested by the MAC. ■ For PIPE 4.3, bit 4 is unused; tie it to 1'b0. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A
pipe_txX_eq_preset_coeff_req	I	TX equalization preset mapping request for lane X; PIPE signal "GetLocalPresetCoeffcients" (Sec 6.1, PIPE 4.4.1/4.3). A MAC holds this signal high for one PCLK cycle when requesting a preset to co-efficient mapping for the preset on pipe_txX_eq_preset[3:0] to coefficients on pipe_txX_eq_preset_coeff[17:0]. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_txX_margin[2:0]	I	TX margin setting for lane X; PIPE signal "TxMargin[2:0]" (Sec 6.1, PIPE 4.4.1/4.3). Selects the transmitter voltage levels as defined in the PIPE 4.4.1/4.3 specification. **Note:** ■ If a PCIe application needs to change transmit margin and update FS/LF/coefficient for new value, the following flow must be executed while running at a speed greater than PCIe Gen2. ■ 1. Change rate from PCIe Gen3/4 to PCIe Gen1 speed. ■ 2. Change "Transmit Margin" register in PCIe controller (Link Control 2 Register in PCI Express base specification). ■ 3. Set "Perform Equalization" bit in PCIe controller (Link Control 3 register in PCI Express base specification). ■ 4. Initiate speed change request in PCIe controller. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A
pipe_txX_ones_zeros	I	USB TX compliance pattern enable for lane X; PIPE 4.3 signal "TxOnesZeros" (Sec 6.1, PIPE 4.3). Reserved **Synchronous To:** pipe_laneX_pclk **Active State:** High

Table 4-2 PIPE Interface Signals (for X = 0; X <= Nlanes-1) (Continued)

Port Name	I/O	Description
pipe_txX_pattern[1:0]	I	TX pattern for SATA OOB for lane X; PIPE 4.3 signal "TX Pattern[1:0]" (Sec 6.1, PIPE 4.3). ■ PCIe: Reserved ■ SATA: Controls which pattern the PHY sends at the 1.5G rate when sending OOB or initialization signaling. **Synchronous To:** pipe_laneX_pclk **Active State:** N/A
pipe_txX_startblock	I	TX start block for lane X; PIPE signal "TxStartBlock" (Sec 6.1, PIPE 4.4.1/4.3). Enables MAC to provide the PHY with the starting byte for a block using non-8b10b data rates. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_txX_swing	I	TX swing control; PIPE signal "TxSwing" (Sec 6.1, PIPE 4.4.1/4.3). Not supported. Reserved. **Synchronous To:** pipe_laneX_pclk **Active State:** High
pipe_txX_syncheader[3:0]	I	TX sync header for lane X; PIPE signal "TxSyncHeader[3:0]" (Sec 6.1, PIPE 4.4.1/4.3). PCIe: ■ Provides the SYNC header for the PHY to use in the next block. ■ The pipe_txX_syncheader[1:0] bits are valid for PCIe. SATA: Reserved **Synchronous To:** pipe_laneX_pclk **Active State:** N/A

4.3 MPLL Control Signals (for N = 0; N <= Nphy-1)

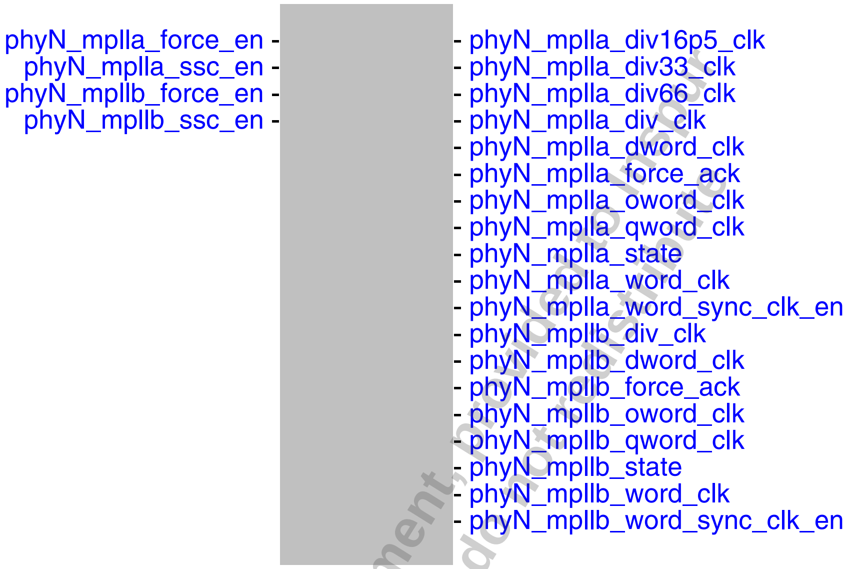


Table 4-3 MPLL Control Signals (for N = 0; N <= Nphy-1)

Port Name	I/O	Description
phyN_mplla_div16p5_clk	O	MPLLA divide by 16.5 clock. (phy0_mpllA only) Divide by 16.5 output clock derived from MPLL(A,B). **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_mplla_div33_clk	O	MPLLA divide by 33 clock. (phy0_mpllA only) Divide by 33 output clock derived from MPLL(A,B). **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_mplla_div66_clk	O	MPLLA divide by 66 clock. (phy0_mpllA only) Divide by 66 output clock derived from MPLL(A,B). **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-3 MPLL Control Signals (for N = 0; N <= Nphy-1) (Continued)

Port Name	I/O	Description
phyN_mplla_div_clk	O	MPLLA divide output clock. ■ mpll(a,b)_div_clk frequency = MPLL VCO frequency/ [mpll(a,b)_div_multiplier[6:0] * (1+ ref_clk_div2_en) * (1+ ref_clk_mpll(a,b)_div2_en)]. ■ When enabled, the frequency of this clock is equal to the MPLL(A,B) frequency divided by mpll(a,b)_div_multiplier. If mpll(a,b)_ssc_en is asserted, then SSC is applied on this clock. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_mplla_dword_clk	O	Double-Rate word clock (mplla_dword_clk). Double-symbol rate (x2) clock derived from MPLLA. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_mplla_force_ack	O	MPLLA Force Acknowledge. ■ Acknowledge output for mpll(a,b)_force_en. ■ This output asserts when the MPLL is powered up in response to mpll(a,b)_force_en assertion. ■ This output de-asserts as a response to mpll(a,b)_force_en de-assertion, and at that point the MPLL will be powereddown if txX_mpll_en of other lanes are de-asserted. **Note:** A four-way level handshake must be followed between mpll(a,b)_force_en and mpll(a,b)_force_ack. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High

Table 4-3 MPLL Control Signals (for N = 0; N <= Nphy-1) (Continued)

Port Name	I/O	Description
phyN_mplla_force_en	I	MPLLA force enable. ■ When asserted, the corresponding MPLL is forced to be powered up, irrespective of the txX_mpll_en input. ■ This input is used for applications where a free-running MPLL clock output is required. ■ If MPLL is not powered up, Synopsys recommends that you follow the txX_mpll_en controls as described in the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Note:** ■ There should be no outstanding pipe interface transition in process before changing phyN_mplla/b_force_en (Powerdown, Rate, Width, and TxDetectRx). ■ There should be no pipe interface transition initiated while phyN_mplla/b_force_en/ack handshake is in progress. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_mplla_oword_clk	O	Octuple-Rate word clock (mplla_oword_clk). Octuple-symbol rate (x8) clock derived from MPLLA. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_mplla_qword_clk	O	Quadruple-Rate word clock (mplla_qword_clk). Quadruple-symbol rate (x4) clock derived from MPLLA. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_mplla_ssc_en	I	Spread spectrum enable (mplla_ssc_en). ■ Enables spread-spectrum clock (SSC) generation on the mpll(a,b)_div_clk output. ■ If the reference clock already has spread spectrum applied, mpll(a,b)_ssc_en must be de-asserted. ■ These inputs can only be changed when the txX_mpll_en inputs for all lanes are de-asserted. **Note:** ■ Use of built-in SSC generation results in increased deterministic jitter (DJ). ■ The SSC profile is not synchronized across aggregated PHYs. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High

Table 4-3 MPLL Control Signals (for N = 0; N <= Nphy-1) (Continued)

Port Name	I/O	Description
phyN_mplla_state	O	MPLLA state indicator. ■ Indicates the state of MPLLA. ■ This signal is asserted when MPLLA is powered up and phase-locked. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_mplla_word_clk	O	Word clock (mplla_word_clk). Symbol rate clock derived from MPLLA. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_mplla_word_sync_clk_en	O	MPLLA word clock sync enable. ■ Controller following "PCLK as PHY output" method should ignore this signal, and leave it unconnected. ■ It is implied that Controller's source for PCLK generation will be the phyN_mplla_word_clk generated by the PLL. **Synchronous To:** phyN_mplla_word_clk **Active State:** N/A
phyN_mpllb_div_clk	O	MPLLB divide output clock. ■ mpll(a,b)_div_clk frequency = MPLL VCO frequency/ [mpll(a,b)_div_multiplier[6:0] * (1+ ref_clk_div2_en) * (1+ ref_clk_mpll(a,b)_div2_en)] ■ When enabled, the frequency of this clock is equal to the MPLL(A,B) frequency divided by mpll(a,b)_div_multiplier. If mpll(a,b)_ssc_en is asserted, then SSC is applied on this clock. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_mpllb_dword_clk	O	Double-Rate word clock (mpllb_dword_clk). Double-symbol rate (x2) clock derived from MPLLB. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-3 MPLL Control Signals (for N = 0; N <= Nphy-1) (Continued)

Port Name	I/O	Description
phyN_mpllb_force_ack	O	MPLLB Force Acknowledge. ■ Acknowledge output for mpll(a,b)_force_en. ■ This output asserts when the MPLL is powered up in response to mpll(a,b)_force_en assertion. ■ This output de-asserts as a response to mpll(a,b)_force_en de-assertion, and at that point the MPLL will be powereddown if txX_mpll_en of other lanes are de-asserted. **Note:** A four-way level handshake must be followed between mpll(a,b)_force_en and mpll(a,b)_force_ack. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_mpllb_force_en	I	MPLLB force enable. ■ When asserted, the corresponding MPLL is forced to be powered up, irrespective of the txX_mpll_en input. ■ This input is used for applications where a free-running MPLL clock output is required. ■ If MPLL is not powered up, Synopsys recommends that you follow the txX_mpll_en controls as described in the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_mpllb_oword_clk	O	Octuple-Rate word clock (mpllb_oword_clk). Octuple-symbol rate (x8) clock derived from MPLLBs. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_mpllb_qword_clk	O	Quadruple-Rate word clock (mpllb_qword_clk). Quadruple-symbol rate (x4) clock derived from MPLLB. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-3 MPLL Control Signals (for N = 0; N <= Nphy-1) (Continued)

Port Name	I/O	Description
phyN_mpllb_ssc_en	I	Spread spectrum enable (mpllb_ssc_en). ■ Enables spread-spectrum clock (SSC) generation on the mpll(a,b)_div_clk output. ■ If the reference clock already has spread spectrum applied, mpll(a,b)_ssc_en must be de-asserted. ■ These inputs can only be changed when the txX_mpll_en inputs for all lanes are de-asserted. **Note:** ■ Use of built-in SSC generation results in increased deterministic jitter (DJ). ■ The SSC profile is not synchronized across aggregated PHYs. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_mpllb_state	O	MPLLB state indicator. ■ Indicates the state of MPLLB. ■ This signal is asserted when MPLLB is powered up and phase-locked. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_mpllb_word_clk	O	Word clock (mpllb_word_clk). Symbol rate clock derived from MPLLB. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_mpllb_word_sync_clk_en	O	MPLLB word clock sync enable. ■ Controller following "PCLK as PHY output" method should ignore this signal, and leave it unconnected. ■ It is implied that Controller's source for PCLK generation will be the phyN_mpllb_word_clk generated by the PLL. **Synchronous To:** phyN_mpllb_word_clk **Active State:** N/A

4.4 Reference Clock Control Signals (for N = 0; N <= Nphy-1)

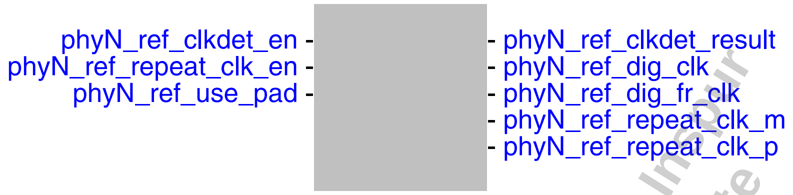


Table 4-4 Reference Clock Control Signals (for N = 0; N <= Nphy-1)

Port Name	I/O	Description
phyN_ref_clkdet_en	I	Enable Reference Clock Detection. Enables detection of the reference clock on either the pads input or the alt input (depending on ref_use_pad). **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_ref_clkdet_result	O	Reference Clock Detection result. ■ Indicates the presence of the reference clock on either the pads input or the alt input (depending on ref_use_pad). ■ The result is only valid when ref_clkdet_en is asserted. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_ref_dig_clk	O	Connects to the phy_ref_dig_clk signal on PHYN. For the full signal description, refer to the DesignWare Cores PHY databook, "Signal Descriptions" chapter. **Synchronous To:** N/A **Active State:** N/A

Table 4-4 Reference Clock Control Signals (for N = 0; N <= Nphy-1) (Continued)

Port Name	I/O	Description
phyN_ref_dig_fr_clk	O	Free-running buffered version of input reference clock. ■ This is a buffered, single-ended version of either ref_pad_clk_p/ref_pad_clk_m or ref_alt_clk_p/ref_alt_clk_m, depending on the settings of ref_use_pad. ■ The frequency of this clock matches that of the input reference clock and is not affected by the setting of ref_clk_div2_en. **Note:** ■ Do not use this signal as a precision clock source for other PHY instantiations. It is not appropriate for such use. ■ This signal may be undriven when power gating is enabled. For more information on power gating, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. ■ ref_dig_fr_clk will be running during phy_reset assertion, if ref_clk_en=1. To ensure clean start, customer should first deassert ref_clk_en and then assert it. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_ref_repeat_clk_en	I	Repeat reference clock enable. ■ Enables the CML output clocks ref_repeat_clk_[p,m]. ■ This pair of clocks can be used as reference clocks for other on-chip PHYs. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_ref_repeat_clk_m	O	Current-mode logic (CML) buffered version of input reference clock (ref_repeat_clk_m). ■ This is a CML-buffered version of either ref_pad_clk_p/ref_pad_clk_m or ref_alt_clk_p/ref_alt_clk_m, depending on the settings of ref_use_pad. ■ This pair of clocks can be used as reference clocks for other PHYs on chip. **Note:** ■ This signal may be undriven when power gating is enabled. ■ For more information on power gating, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-4 Reference Clock Control Signals (for N = 0; N <= Nphy-1) (Continued)

Port Name	I/O	Description
phyN_ref_repeat_clk_p	O	Current-mode logic (CML) buffered version of input reference clock (ref_repeat_clk_p). ■ This is a CML-buffered version of either ref_pad_clk_p/ref_pad_clk_m or ref_alt_clk_p/ref_alt_clk_m, depending on the settings of ref_use_pad. ■ This pair of clocks can be used as reference clocks for other PHYs on chip. **Note:** ■ This signal may be undriven when power gating is enabled. ■ For more information on power gating, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_ref_use_pad	I	Select reference clock connected to ref_pad_clk_p/ref_pad_clk_m. ■ Selects the external ref_pad_clk_p and ref_pad_clk_m inputs as the reference clock source when asserted. ■ When de-asserted, ref_alt_clk_p and ref_alt_clk_m are the sources of the reference clock. ■ Any change in this input must be followed by phy_reset assertion. **Note:** A transition on this input must be followed by the assertion of phy_reset. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** High

4.5 Power Related Signals

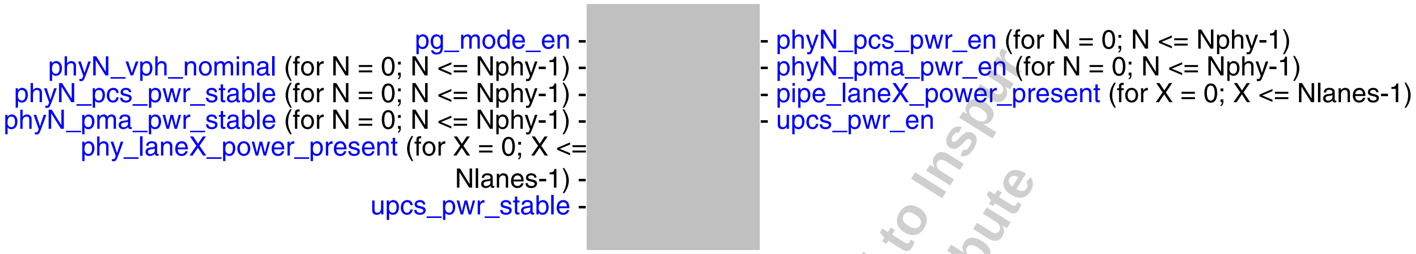

pg_mode_en -
phyN_vph_nominal (for N = 0; N <= Nphy-1) -
phyN_pcs_pwr_stable (for N = 0; N <= Nphy-1) -
phyN_pma_pwr_stable (for N = 0; N <= Nphy-1) -
phy_laneX_power_present (for X = 0; X <= Nlanes-1) -
upcs_pwr_stable -
- phyN_pcs_pwr_en (for N = 0; N <= Nphy-1)
- phyN_pma_pwr_en (for N = 0; N <= Nphy-1)
- pipe_laneX_power_present (for X = 0; X <= Nlanes-1)
- upcs_pwr_en

Table 4-5 Power Related Signals

Port Name	I/O	Description
pg_mode_en	I	Power gating support enable. ■ Control input to enable the power gating support. ■ When de-asserted, the control inputs related to power gating are ignored. ■ For power gating sequencing, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. ■ This input is meant to be a tie-off and can be transitioned only during the PHY reset sequence. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_vph_nominal[1:0] (for N = 0; N <= Nphy-1)	I	For information about this signal, refer to chapter "Signal Descriptions" in the DesignWare Cores PHY databook. **Synchronous To:** N/A **Active State:** N/A
phyN_pcs_pwr_en (for N = 0; N <= Nphy-1)	O	Power enable for Raw PCS power switches. ■ Enable signal for external switches to supply power to the power gated logic in Raw PCS. ■ For power gating sequencing, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High

Table 4-5 Power Related Signals (Continued)

Port Name	I/O	Description
phyN_pcs_pwr_stable (for N = 0; N <= Nphy-1)	I	Power stable for Raw PCS. ■ Status signal indicating that the power for the Raw PCS is stable. ■ The pcs_pwr_stable signal should only be asserted if the supply is 90% of nominal or higher. ■ For power gating sequencing, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_pma_pwr_en (for N = 0; N <= Nphy-1)	O	Power enable for PMA power switch. ■ Enable signal for PMA power switch (external) to supply power to the PMA. ■ For power gating sequencing, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_pma_pwr_stable (for N = 0; N <= Nphy-1)	I	Power stable for PMA. ■ Status signal indicating that the power for the PMA is stable. ■ The pma_pwr_stable signal should only be asserted if the supply is 90% of nominal or higher. ■ For power gating sequencing, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phy_laneX_power_present (for X = 0; X <= Nlanes-1)	I	Power present for lane X; PIPE signal "PowerPresent" (Sec 6.1, PIPE 4.4.1/4.3). ■ The VBUS detection circuitry is meant to be implemented externally to the PHY and PCS. ■ This output is the same as the phy_laneX_power_present input, which must be driven by external VBUS detection logic. **Synchronous To:** Asynchronous **Active State:** High
pipe_laneX_power_present (for X = 0; X <= Nlanes-1)	O	Power present for lane X; PIPE signal "PowerPresent" (Sec 6.1, PIPE 4.4.1/4.3). ■ The VBUS detection circuitry is meant to be implemented externally to the PHY and PCS. ■ This output is the same as the phy_laneX_power_present input, which must be driven by external VBUS detection logic. **Synchronous To:** Asynchronous **Active State:** High

Table 4-5 Power Related Signals (Continued)

Port Name	I/O	Description
upcs_pwr_en	O	Power enable for PCS power switch(es). ■ Enable signal for external switch(es) to supply power to the power-gated logic in the PCS. ■ For information about power gating, see the "Power-Gating Support" section. **Synchronous To:** Asynchronous **Active State:** High
upcs_pwr_stable	I	Power stable to PCS. ■ Status signal indicating that power for the PCS is stable. ■ For information about power gating, see the "Power-Gating Support" section. **Synchronous To:** Asynchronous **Active State:** High

4.6 Analog I/O Signals

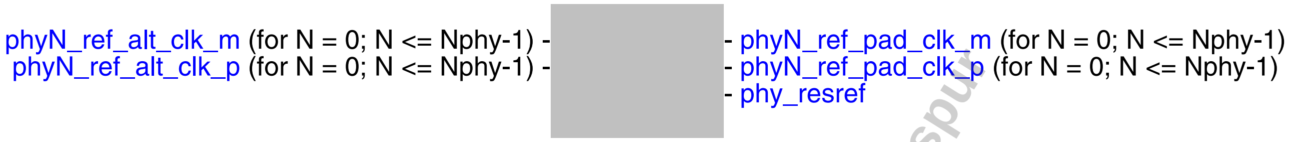

phyN_ref_alt_clk_m (for N = 0; N <= Nphy-1) -
phyN_ref_alt_clk_p (for N = 0; N <= Nphy-1) -
- phyN_ref_pad_clk_m (for N = 0; N <= Nphy-1)
- phyN_ref_pad_clk_p (for N = 0; N <= Nphy-1)
- phy_resref

Table 4-6 Analog I/O Signals

Port Name	I/O	Description
phyN_ref_alt_clk_m (for N = 0; N <= Nphy-1)	I	Low-Swing differential input clock pair internal pin (ref_alt_clk_m). ■ Either ref_pad_clk_[p,m] or ref_alt_clk_[p,m] is available on a core, but both must not be active at the same time. ■ The selection is based on whether the reference clock is provided to the core via an on-die distribution network or directly to the core from off- die. ■ The pair that is selected depends on the value of the ref_use_pad input. **Voltage Range:** 0-vp **Synchronous To:** N/A **Active State:** N/A
phyN_ref_alt_clk_p (for N = 0; N <= Nphy-1)	I	Low-Swing differential input clock pair internal pin (ref_alt_clk_p). ■ Either ref_pad_clk_[p,m] or ref_alt_clk_[p,m] is available on a core, but both must not be active at the same time. ■ The selection is based on whether the reference clock is provided to the core via an on-die distribution network or directly to the core from off- die. ■ The pair that is selected depends on the value of the ref_use_pad input. **Voltage Range:** 0-vp **Synchronous To:** N/A **Active State:** N/A
phyN_ref_pad_clk_m (for N = 0; N <= Nphy-1)	IO	Low-Swing differential input clock pair with pad (ref_pad_clk_m). ■ Either ref_pad_clk_[p,m] or ref_alt_clk_[p,m] is available on a core, but both must not be active at the same time. ■ The selection is based on whether the reference clock is provided to the core through an on-die distribution network or directly to the core from off-die. ■ The pair that is selected depends on the value of the ref_use_pad input. **Voltage Range:** 0-vp **Synchronous To:** N/A **Active State:** N/A

Table 4-6 Analog I/O Signals (Continued)

Port Name	I/O	Description
phyN_ref_pad_clk_p (for N = 0; N <= Nphy-1)	IO	Low-Swing differential input clock pair with pad (ref_pad_clk_p). ■ Either ref_pad_clk_[p,m] or ref_alt_clk_[p,m] is available on a core, but both must not be active at the same time. ■ The selection is based on whether the reference clock is provided to the core through an on-die distribution network or directly to the core from off-die. ■ The pair that is selected depends on the value of the ref_use_pad input. **Voltage Range:** 0-vp **Synchronous To:** N/A **Active State:** N/A
phy_resref	IO	Reference Resistor Connection Attach a precision resistor-to-ground on the board, with the following specifications: ■ Resistance: 200 Ohm ■ Temperature: Coefficient +/-100 ppm/degree C ■ Tolerance: +/-1% For information about resref, see the "Physical-Level Implementation" and "Board- and Package-Level Implementation" chapters. **Note:** The .LIB file defines "related_power_pin" to be vph as the internal circuitry and ESD protection are related to vph. **Voltage Range:** 0-vp **Synchronous To:** N/A **Active State:** N/A

4.7 Receiver Signals (for X = 0; X <= Nlanes-1)

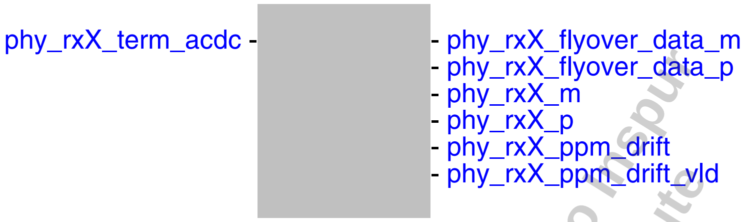


Table 4-7 Receiver Signals (for X = 0; X <= Nlanes-1)

Port Name	I/O	Description
phy_rxX_flyover_data_m	O	Receive Flyover Data Outputs (rxX_flyover_data_m). Outputs are directly connected to RX bump pads when test_flyover_en is enabled. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phy_rxX_flyover_data_p	O	Receive Flyover Data Outputs (rxX_flyover_data_p). Outputs are directly connected to RX bump pads when test_flyover_en is enabled. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phy_rxX_m	IO	High-Speed differential receive pair (rxX_m). Receive differential pair for lane X. **Voltage Range:** 0-vp **Synchronous To:** N/A **Active State:** N/A
phy_rxX_p	IO	High-Speed differential receive pair (rxX_p). Receive differential pair for lane X. **Voltage Range:** 0-vp **Synchronous To:** N/A **Active State:** N/A

Table 4-7 Receiver Signals (for X = 0; X <= Nlanes-1) (Continued)

Port Name	I/O	Description
phy_rxX_ppm_drift[5:0]	O	RX CDR PPM Drift. ■ This value represents the amount of ppm on the rxX_clk with respect to the ideal desired frequency. ■ For more information, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. ■ This output is valid when rxX_ppm_drift_vld is asserted. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** N/A
phy_rxX_ppm_drift_vld	O	RX CDR PPM Drift Valid. ■ Indicates when the rxX_ppm_drift[5:0] output is valid and can be sampled. ■ For more information, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phy_rxX_term_acdc	I	Receiver termination control. Controls the terminations of the RX. Selector for floating versus grounded RX termination: ■ 0 : Floating RX termination ■ 1 : Grounded RX termination **Note:** ■ Both AC and DC coupled links should only use the grounded termination option. ■ Use of a DC coupled link with grounded termination can affect operation of some functions, that is TX RX detect, ACJTAG, and so on. ■ If you intend to use a DC coupled link, consult Synopsys. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High

4.8 Transmitter Signals (for X = 0; X <= Nlanes-1)

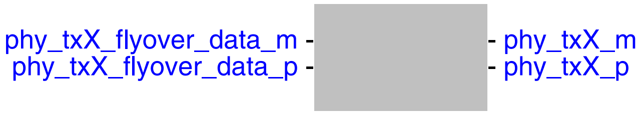


Table 4-8 Transmitter Signals (for X = 0; X <= Nlanes-1)

Port Name	I/O	Description
phy_txX_flyover_data_m	I	Transmit Flyover Data inputs (txX_flyover_data_m). Inputs are directly connected to TX bump pads when test_flyover_en is enabled. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phy_txX_flyover_data_p	I	Transmit Flyover Data inputs (txX_flyover_data_p). Inputs are directly connected to TX bump pads when test_flyover_en is enabled. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phy_txX_m	IO	High-Speed differential transmit pair (txX_m). Transmit differential pair for lane X. **Voltage Range:** 0-vptxX **Synchronous To:** N/A **Active State:** N/A
phy_txX_p	IO	High-Speed differential transmit pair (txX_p). Transmit differential pair for lane X. **Voltage Range:** 0-vptxX **Synchronous To:** N/A **Active State:** N/A

4.9 PCS Scan Interface Signals

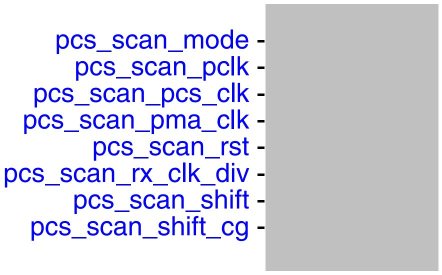


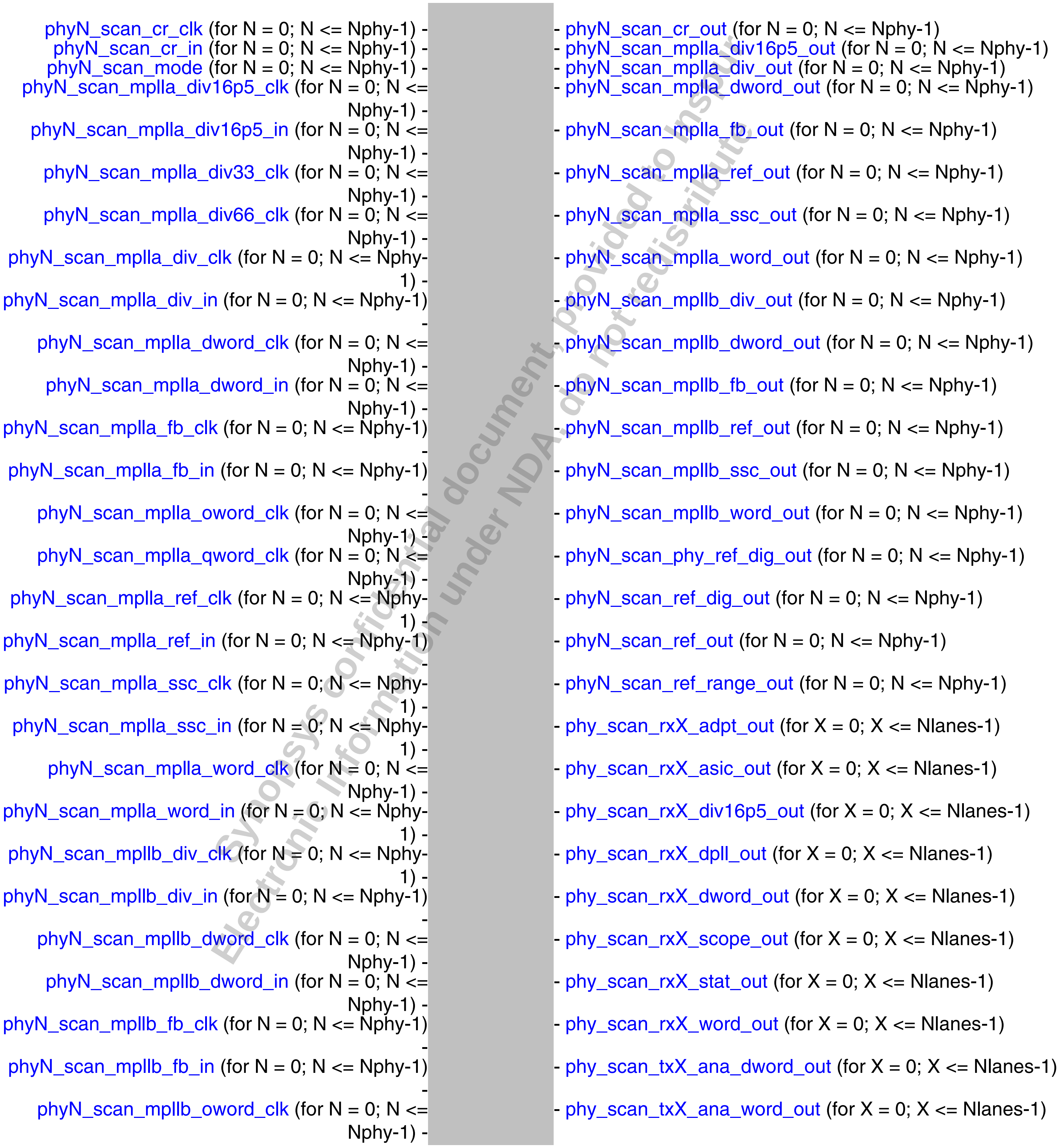
Table 4-9 PCS Scan Interface Signals

Port Name	I/O	Description
pcs_scan_mode	I	This signal is PCS scan mode enable. Scan mode enable input for the PCS. **Synchronous To:** N/A **Active State:** High
pcs_scan_pclk	I	PCS scan clock for PCLK. Scan clock input for laneX_pclk functional clock domain. **Synchronous To:** N/A **Active State:** N/A
pcs_scan_pcs_clk	I	PCS scan clock. Scan clock input for laneX_pcs_clk internal, functional clock domain. **Synchronous To:** N/A **Active State:** N/A
pcs_scan_pma_clk	I	This signal is PMA scan clock. Scan clock input for laneX_pma_clk internal, functional clock domain. **Synchronous To:** N/A **Active State:** N/A
pcs_scan_rst	I	This signal is PCS scan reset. Scan reset input for the PCS; should be de-asserted when pcs_scan_mode is de-asserted. **Synchronous To:** N/A **Active State:** High
pcs_scan_rx_clk_div	I	This signal is scan for RX clock div clock. Scan clock input for laneX.rx.phy_rx_clk_div1_or_div2 internal functional clock domain. **Synchronous To:** N/A **Active State:** N/A

Table 4-9 PCS Scan Interface Signals (Continued)

Port Name	I/O	Description
pcs_scan_shift	I	This signal is scan enable port for PCS. Dedicated scan enable for PCS. **Synchronous To:** N/A **Active State:** High
pcs_scan_shift_cg	I	This signal is scan enable port for PCS clock gates. Dedicated scan enable for clock gating logic in PCS. **Synchronous To:** N/A **Active State:** High

4.10 PHY Scan Interface Signals


phyN_scan_cr_clk (for N = 0; N <= Nphy-1) -
phyN_scan_cr_in (for N = 0; N <= Nphy-1) -
phyN_scan_mode (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_div16p5_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_div16p5_in (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_div33_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_div66_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_div_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_div_in (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_dword_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_dword_in (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_fb_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_fb_in (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_oword_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_qword_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_ref_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_ref_in (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_ssc_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_ssc_in (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_word_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mplla_word_in (for N = 0; N <= Nphy-1) -
phyN_scan_mpllb_div_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mpllb_div_in (for N = 0; N <= Nphy-1) -
phyN_scan_mpllb_dword_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mpllb_dword_in (for N = 0; N <= Nphy-1) -
phyN_scan_mpllb_fb_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mpllb_fb_in (for N = 0; N <= Nphy-1) -
phyN_scan_mpllb_oword_clk (for N = 0; N <= Nphy-1) -
- phyN_scan_cr_out (for N = 0; N <= Nphy-1)
- phyN_scan_mplla_div16p5_out (for N = 0; N <= Nphy-1)
- phyN_scan_mplla_div_out (for N = 0; N <= Nphy-1)
- phyN_scan_mplla_dword_out (for N = 0; N <= Nphy-1)
- phyN_scan_mplla_fb_out (for N = 0; N <= Nphy-1)
- phyN_scan_mplla_ref_out (for N = 0; N <= Nphy-1)
- phyN_scan_mplla_ssc_out (for N = 0; N <= Nphy-1)
- phyN_scan_mplla_word_out (for N = 0; N <= Nphy-1)
- phyN_scan_mpllb_div_out (for N = 0; N <= Nphy-1)
- phyN_scan_mpllb_dword_out (for N = 0; N <= Nphy-1)
- phyN_scan_mpllb_fb_out (for N = 0; N <= Nphy-1)
- phyN_scan_mpllb_ref_out (for N = 0; N <= Nphy-1)
- phyN_scan_mpllb_ssc_out (for N = 0; N <= Nphy-1)
- phyN_scan_mpllb_word_out (for N = 0; N <= Nphy-1)
- phyN_scan_phy_ref_dig_out (for N = 0; N <= Nphy-1)
- phyN_scan_ref_dig_out (for N = 0; N <= Nphy-1)
- phyN_scan_ref_out (for N = 0; N <= Nphy-1)
- phyN_scan_ref_range_out (for N = 0; N <= Nphy-1)
- phy_scan_rxX_adpt_out (for X = 0; X <= Nlanes-1)
- phy_scan_rxX_asic_out (for X = 0; X <= Nlanes-1)
- phy_scan_rxX_div16p5_out (for X = 0; X <= Nlanes-1)
- phy_scan_rxX_dpll_out (for X = 0; X <= Nlanes-1)
- phy_scan_rxX_dword_out (for X = 0; X <= Nlanes-1)
- phy_scan_rxX_scope_out (for X = 0; X <= Nlanes-1)
- phy_scan_rxX_stat_out (for X = 0; X <= Nlanes-1)
- phy_scan_rxX_word_out (for X = 0; X <= Nlanes-1)
- phy_scan_txX_ana_dword_out (for X = 0; X <= Nlanes-1)
- phy_scan_txX_ana_word_out (for X = 0; X <= Nlanes-1)

phyN_scan_mpllb_qword_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mpllb_ref_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mpllb_ref_in (for N = 0; N <= Nphy-1) -
phyN_scan_mpllb_ssc_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mpllb_ssc_in (for N = 0; N <= Nphy-1) -
phyN_scan_mpllb_word_clk (for N = 0; N <= Nphy-1) -
phyN_scan_mpllb_word_in (for N = 0; N <= Nphy-1) -
phyN_scan_phy_ref_dig_clk (for N = 0; N <= Nphy-1) -
phyN_scan_phy_ref_dig_in (for N = 0; N <= Nphy-1) -
phyN_scan_ref_clk (for N = 0; N <= Nphy-1) -
phyN_scan_ref_dig_clk (for N = 0; N <= Nphy-1) -
phyN_scan_ref_dig_in (for N = 0; N <= Nphy-1) -
phyN_scan_ref_in (for N = 0; N <= Nphy-1) -
phyN_scan_ref_range_clk (for N = 0; N <= Nphy-1) -
phyN_scan_ref_range_in (for N = 0; N <= Nphy-1) -
phyN_scan_set_rst (for N = 0; N <= Nphy-1) -
phyN_scan_shift (for N = 0; N <= Nphy-1) -
phyN_scan_shift_cg (for N = 0; N <= Nphy-1) -
phy_scan_rxX_adpt_clk (for X = 0; X <= Nlanes-1) -
phy_scan_rxX_adpt_in (for X = 0; X <= Nlanes-1) -
phy_scan_rxX_asic_clk (for X = 0; X <= Nlanes-1) -
phy_scan_rxX_asic_in (for X = 0; X <= Nlanes-1) -
phy_scan_rxX_div16p5_clk (for X = 0; X <= Nlanes-1) -
phy_scan_rxX_div16p5_in (for X = 0; X <= Nlanes-1) -
phy_scan_rxX_dpll_clk (for X = 0; X <= Nlanes-1) -
phy_scan_rxX_dpll_in (for X = 0; X <= Nlanes-1) -
phy_scan_rxX_dword_clk (for X = 0; X <= Nlanes-1) -
phy_scan_rxX_dword_in (for X = 0; X <= Nlanes-1) -
phy_scan_rxX_scope_clk (for X = 0; X <= Nlanes-1) -
phy_scan_rxX_scope_in (for X = 0; X <= Nlanes-1) -
phy_scan_rxX_stat_clk (for X = 0; X <= Nlanes-1) -
phy_scan_rxX_stat_in (for X = 0; X <= Nlanes-1) -
phy_scan_rxX_word_clk (for X = 0; X <= Nlanes-1) -

- phy_scan_txX_out (for X = 0; X <= Nlanes-1) phy_scan_rxX_word_in (for X = 0; X <= Nlanes-1) -

phy_scan_txX_ana_dword_clk (for X = 0; X <= Nlanes-1) -

phy_scan_txX_ana_dword_in (for X = 0; X <= Nlanes-1) -

phy_scan_txX_ana_word_clk (for X = 0; X <= Nlanes-1) -

phy_scan_txX_ana_word_in (for X = 0; X <= Nlanes-1) -

phy_scan_txX_in (for X = 0; X <= Nlanes-1) -

Table 4-10 PHY Scan Interface Signals

Port Name	I/O	Description
phyN_scan_cr_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_cr_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_scan_cr_in[150:0] (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_cr_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_cr_out[150:0] (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_cr_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_mode (for N = 0; N <= Nphy-1)	I	Scan mode enable. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_scan_mplla_div16p5_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mplla_div16p5_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_scan_mplla_div16p5_in (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_mplla_div16p5_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phyN_scan_mplla_div16p5_out (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_mplla_div16p5_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_mplla_div33_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mplla_div33_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_scan_mplla_div66_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mplla_div66_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_scan_mplla_div_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mplla_div_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_mplla_div_in (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_mplla_div_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phyN_scan_mplla_div_out (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_mplla_div_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_mplla_dword_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mplla_dword_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_mplla_dword_in (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_mplla_dword_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phyN_scan_mplla_dword_out (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_mplla_dword_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_mplla_fb_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mplla_fb_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_mplla_fb_in[0:0] (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_mplla_fb_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phyN_scan_mplla_fb_out[0:0] (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_mplla_fb_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_mplla_oword_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mplla_oword_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_scan_mplla_qword_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mplla_qword_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_mplla_ref_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mplla_ref_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_scan_mplla_ref_in[0:0] (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_mplla_ref_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phyN_scan_mplla_ref_out[0:0] (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_mplla_ref_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_mplla_ssc_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mplla_ssc_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_mplla_ssc_in[2:0] (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_mplla_ssc_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phyN_scan_mplla_ssc_out[2:0] (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_mplla_ssc_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_mplla_word_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mplla_word_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_mplla_word_in[0:0] (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_mplla_word_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phyN_scan_mplla_word_out[0:0] (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_mplla_word_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_mpllb_div_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mpllb_div_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_mpllb_div_in (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_mpllb_div_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phyN_scan_mpllb_div_out (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_mpllb_div_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_mpllb_dword_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mpllb_dword_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_mpllb_dword_in (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_mpllb_dword_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phyN_scan_mpllb_dword_out (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_mpllb_dword_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_mpllb_fb_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mpllb_fb_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_mpllb_fb_in[0:0] (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_mpllb_fb_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phyN_scan_mpllb_fb_out[0:0] (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_mpllb_fb_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_mpllb_oword_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mpllb_oword_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_scan_mpllb_qword_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mpllb_qword_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_mpllb_ref_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mpllb_ref_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_scan_mpllb_ref_in[0:0] (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_mpllb_ref_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phyN_scan_mpllb_ref_out[0:0] (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_mpllb_ref_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_mpllb_ssc_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mpllb_ssc_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_mpllb_ssc_in[2:0] (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_mpllb_ssc_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phyN_scan_mpllb_ssc_out[2:0] (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_mpllb_ssc_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_mpllb_word_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_mpllb_word_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_mpllb_word_in[0:0] (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_mpllb_word_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phyN_scan_mpllb_word_out[0:0] (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_mpllb_word_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_phy_ref_dig_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_phy_ref_dig_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_phy_ref_dig_in[0:0] (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_phy_ref_dig_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phyN_scan_phy_ref_dig_out[0:0] (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_phy_ref_dig_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_ref_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_ref_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_scan_ref_dig_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_ref_dig_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_ref_dig_in[10:0] (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_ref_dig_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phyN_scan_ref_dig_out[10:0] (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_ref_dig_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_ref_in[0:0] (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_ref_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_ref_out[0:0] (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_ref_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_ref_range_clk (for N = 0; N <= Nphy-1)	I	Scan clock per clock domain (scan_ref_range_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_scan_ref_range_in[22:0] (for N = 0; N <= Nphy-1)	I	Scan chain inputs per clock domain (scan_ref_range_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phyN_scan_ref_range_out[22:0] (for N = 0; N <= Nphy-1)	O	Scan chain outputs per clock domain (scan_ref_range_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phyN_scan_set_rst (for N = 0; N <= Nphy-1)	I	Scan set reset. **Voltage Range:** 0-vpdig **Synchronous To:** scan_CLKDOMAIN_clk **Active State:** High

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phyN_scan_shift (for N = 0; N <= Nphy-1)	I	Scan shift. **Voltage Range:** 0-vpdig **Synchronous To:** scan_CLKDOMAIN_clk **Active State:** High
phyN_scan_shift_cg (for N = 0; N <= Nphy-1)	I	Scan shift for clock gates. **Voltage Range:** 0-vpdig **Synchronous To:** scan_CLKDOMAIN_clk **Active State:** High
phy_scan_rxX_adpt_clk (for X = 0; X <= Nlanes-1)	I	Scan clock per clock domain (scan_rxX_adpt_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phy_scan_rxX_adpt_in[5:0] (for X = 0; X <= Nlanes-1)	I	Scan chain inputs per clock domain (scan_rxX_adpt_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phy_scan_rxX_adpt_out[5:0] (for X = 0; X <= Nlanes-1)	O	Scan chain outputs per clock domain (scan_rxX_adpt_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phy_scan_rxX_asic_clk (for X = 0; X <= Nlanes-1)	I	Scan clock per clock domain (scan_rxX_asic_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phy_scan_rxX_asic_in[0:0] (for X = 0; X <= Nlanes-1)	I	Scan chain inputs per clock domain (scan_rxX_asic_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phy_scan_rxX_asic_out[0:0] (for X = 0; X <= Nlanes-1)	O	Scan chain outputs per clock domain (scan_rxX_asic_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phy_scan_rxX_div16p5_clk (for X = 0; X <= Nlanes-1)	I	Scan clock per clock domain (scan_rxX_div16p5_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phy_scan_rxX_div16p5_in[0:0] (for X = 0; X <= Nlanes-1)	I	Scan chain inputs per clock domain (scan_rxX_div16p5_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phy_scan_rxX_div16p5_out[0:0] (for X = 0; X <= Nlanes-1)	O	Scan chain outputs per clock domain (scan_rxX_div16p5_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phy_scan_rxX_dpll_clk (for X = 0; X <= Nlanes-1)	I	Scan clock per clock domain (scan_rxX_dpll_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phy_scan_rxX_dpll_in[1:0] (for X = 0; X <= Nlanes-1)	I	Scan chain inputs per clock domain (scan_rxX_dpll_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phy_scan_rxX_dpll_out[1:0] (for X = 0; X <= Nlanes-1)	O	Scan chain outputs per clock domain (scan_rxX_dpll_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phy_scan_rxX_dword_clk (for X = 0; X <= Nlanes-1)	I	Scan clock per clock domain (scan_rxX_dword_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phy_scan_rxX_dword_in[9:0] (for X = 0; X <= Nlanes-1)	I	Scan chain inputs per clock domain (scan_rxX_dword_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phy_scan_rxX_dword_out[9:0] (for X = 0; X <= Nlanes-1)	O	Scan chain outputs per clock domain (scan_rxX_dword_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phy_scan_rxX_scope_clk (for X = 0; X <= Nlanes-1)	I	Scan clock per clock domain (scan_rxX_scope_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phy_scan_rxX_scope_in[0:0] (for X = 0; X <= Nlanes-1)	I	Scan chain inputs per clock domain (scan_rxX_scope_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phy_scan_rxX_scope_out[0:0] (for X = 0; X <= Nlanes-1)	O	Scan chain outputs per clock domain (scan_rxX_scope_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phy_scan_rxX_stat_clk (for X = 0; X <= Nlanes-1)	I	Scan clock per clock domain (scan_rxX_stat_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phy_scan_rxX_stat_in[8:0] (for X = 0; X <= Nlanes-1)	I	Scan chain inputs per clock domain (scan_rxX_stat_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phy_scan_rxX_stat_out[8:0] (for X = 0; X <= Nlanes-1)	O	Scan chain outputs per clock domain (scan_rxX_stat_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phy_scan_rxX_word_clk (for X = 0; X <= Nlanes-1)	I	Scan clock per clock domain (scan_rxX_word_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phy_scan_rxX_word_in[0:0] (for X = 0; X <= Nlanes-1)	I	Scan chain inputs per clock domain (scan_rxX_word_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phy_scan_rxX_word_out[0:0] (for X = 0; X <= Nlanes-1)	O	Scan chain outputs per clock domain (scan_rxX_word_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phy_scan_txX_ana_dword_clk (for X = 0; X <= Nlanes-1)	I	Scan clock per clock domain (scan_txX_ana_dword_clk). ■ Test Clock muxed into the clock pin of registers during Scan operation. ■ All scan clock inputs are assumed to be asynchronous to each other. ■ All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phy_scan_txX_ana_dword_in[1:0] (for X = 0; X <= Nlanes-1)	I	Scan chain inputs per clock domain (scan_txX_ana_dword_in). **Note:** - The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). - For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: - scan_mplla_div33_clk - scan_mplla_div66_clk - scan_mplla_oword_clk - scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phy_scan_txX_ana_dword_out[1:0] (for X = 0; X <= Nlanes-1)	O	Scan chain outputs per clock domain (scan_txX_ana_dword_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phy_scan_txX_ana_word_clk (for X = 0; X <= Nlanes-1)	I	Scan clock per clock domain (scan_txX_ana_word_clk). - Test Clock muxed into the clock pin of registers during Scan operation. - All scan clock inputs are assumed to be asynchronous to each other. - All scan clock inputs can be active at the same time during scan shift operation; however, only one scan clock can be active at any time during the scan capture cycle. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phy_scan_txX_ana_word_in[0:0] (for X = 0; X <= Nlanes-1)	I	Scan chain inputs per clock domain (scan_txX_ana_word_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.
phy_scan_txX_ana_word_out[0:0] (for X = 0; X <= Nlanes-1)	O	Scan chain outputs per clock domain (scan_txX_ana_word_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.
phy_scan_txX_in[2:0] (for X = 0; X <= Nlanes-1)	I	Scan chain inputs per clock domain (scan_txX_in). **Note:** ■ The scan shift inputs and outputs reflect the scan chains for only the hard macro (PMA). ■ For the PCS portion of the PHY, it is assumed that you insert scan and add the appropriate scan chains. The following scan clocks do not have associated scan chains in the PHY, but get muxed into functional clocks outputs: ■ scan_mplla_div33_clk ■ scan_mplla_div66_clk ■ scan_mplla_oword_clk ■ scan_mplla_oword_clk **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_in[*] is synchronous to txX_clk.

Table 4-10 PHY Scan Interface Signals (Continued)

Port Name	I/O	Description
phy_scan_txX_out[2:0] (for X = 0; X <= Nlanes-1)	O	Scan chain outputs per clock domain (scan_txX_out). **Voltage Range:** 0-vpdig **Active State:** High **Synchronous To:** scan_CLKDOMAIN_clk **Note:** scan_txX_out[*] is synchronous to txX_clk.

4.11 JTAG Port Signals (for N = 0; N <= Nphy-1)

phyN_jtag_tck
phyN_jtag_tdi
phyN_jtag_tms
phyN_jtag_trst_n
phyN_jtag_tdo
phyN_jtag_tdo_en

Table 4-11 JTAG Port Signals (for N = 0; N <= Nphy-1)

Port Name	I/O	Description
phyN_jtag_tck	I	JTAG clock. JTAG state machine clock signal. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_jtag_tdi	I	JTAG input port. JTAG test data input signal. **Voltage Range:** 0-vpdig **Synchronous To:** jtag_tck **Active State:** N/A
phyN_jtag_tdo	O	JTAG output port. JTAG test data output signal. **Voltage Range:** 0-vpdig **Synchronous To:** jtag_tck **Active State:** N/A
phyN_jtag_tdo_en	O	JTAG output enable. JTAG test data output enable signal. **Voltage Range:** 0-vpdig **Synchronous To:** jtag_tck **Active State:** High
phyN_jtag_tms	I	JTAG state machine control. JTAG test mode select signal. **Voltage Range:** 0-vpdig **Synchronous To:** jtag_tck **Active State:** High

Table 4-11 JTAG Port Signals (for N = 0; N <= Nphy-1) (Continued)

Port Name	I/O	Description
phyN_jtag_trst_n	I	JTAG reset. ■ JTAG state machine reset signal. ■ While this signal is held low, the state machine is held in test logic reset state. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** Low

4.12 Boundary Scan Port Signals (for N = 0; N <= Nphy-1)

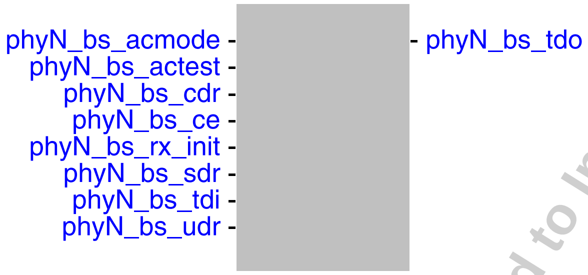


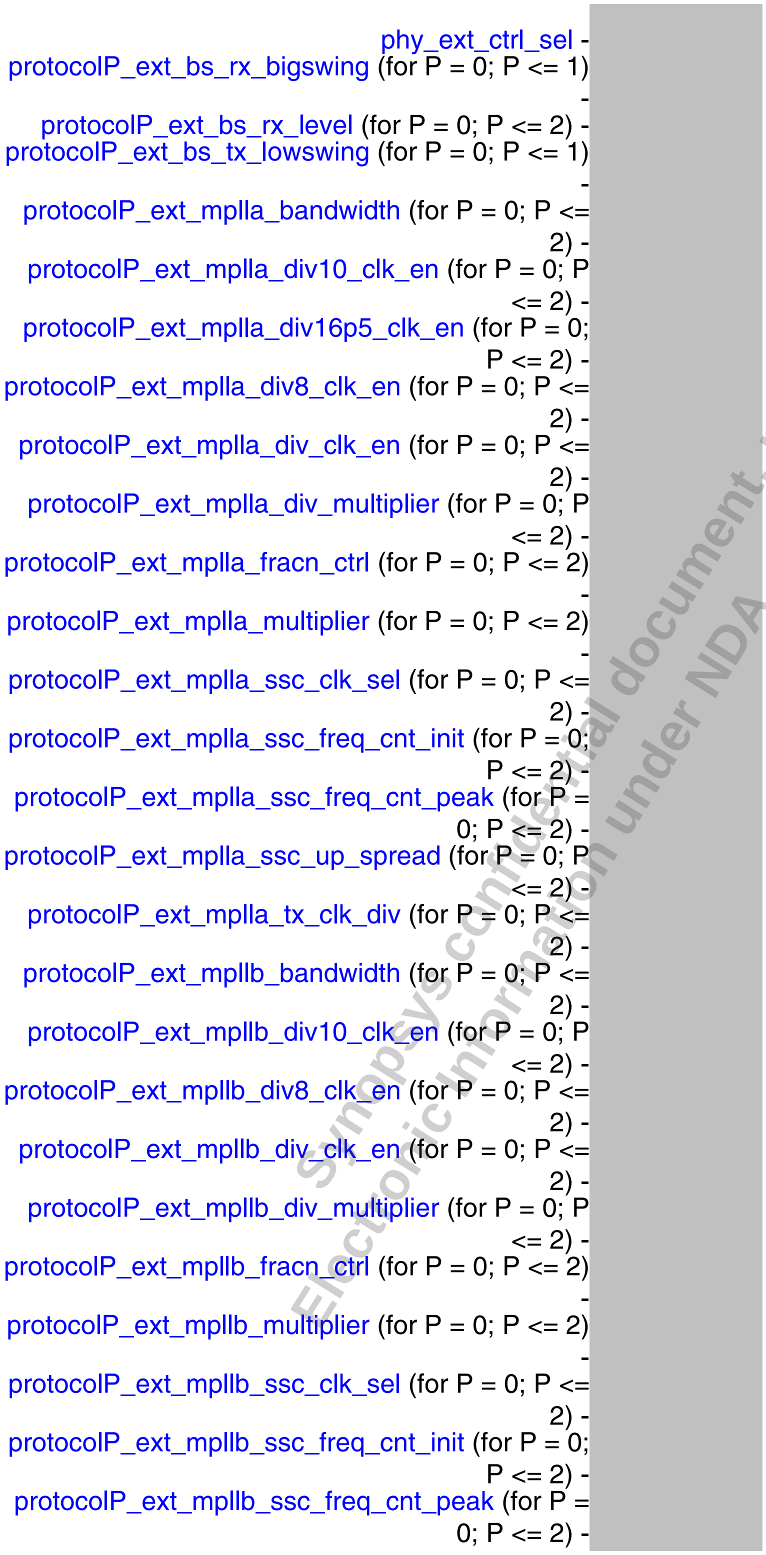
Table 4-12 Boundary Scan Port Signals (for N = 0; N <= Nphy-1)

Port Name	I/O	Description
phyN_bs_acmode	I	Boundary Scan AC Mode (?Figure 51?, IEEE 1149.6). Enables Boundary Scan AC mode. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_bs_actest	I	Boundary Scan AC Test Signal (?Figure 51?, IEEE 1149.6). The AC Test Signal is generated by the user logic when AC EXTEST instruction is decoded. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_bs_cdr	I	Boundary Scan Clock-DR (IEEE 1149.1). Clocks the Boundary Scan capture registers, in either Scan or Capture modes. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_bs_ce	I	Boundary scan compliance enable signal. ■ Enables the Boundary Scan Transmitter and Receiver register cells on all lanes. ■ This input should be driven from a sequential element and not from any combinatorial logic. ■ Any glitch on this input causes the PHY to go into reset state. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High

Table 4-12 Boundary Scan Port Signals (for N = 0; N <= Nphy-1) (Continued)

Port Name	I/O	Description
phyN_bs_rx_init	I	RX Boundary Scan Initialize Signal. ■ This signal is similar to the 'Init Clk' referenced in test receiver diagrams in the ACJTAG specification (Figure 48, IEEE 1149.6). ■ While this signal is high, the hysteresis in the receiver comparator is reset. ■ The falling edge of this signal must occur at appropriate times, as specified in the ACJTAG specification. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_bs_sdr	I	Boundary Scan Shift-DR (IEEE 1149.1). Enables shifting of the Boundary Scan capture registers. **Voltage Range:** 0-vpdig **Synchronous To:** bs_cdr **Active State:** High
phyN_bs_tdi	I	Boundary Scan Test Data In. Boundary scan shift register input signal. **Voltage Range:** 0-vpdig **Synchronous To:** bs_cdr **Active State:** High
phyN_bs_tdo	O	Boundary Scan Test Data Out. Boundary scan shift register output signal. **Voltage Range:** 0-vpdig **Synchronous To:** bs_cdr **Active State:** High
phyN_bs_udr	I	Boundary Scan Update-DR (IEEE 1149.1). Clocks the Boundary Scan update registers. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

4.13 External Protocol Override Signals


phy_ext_ctrl_sel -
protocolP_ext_bs_rx_bigswing (for P = 0; P <= 1) -
protocolP_ext_bs_rx_level (for P = 0; P <= 2) -
protocolP_ext_bs_tx_lowswing (for P = 0; P <= 1) -
protocolP_ext_mplla_bandwidth (for P = 0; P <= 2) -
protocolP_ext_mplla_div10_clk_en (for P = 0; P <= 2) -
protocolP_ext_mplla_div16p5_clk_en (for P = 0; P <= 2) -
protocolP_ext_mplla_div8_clk_en (for P = 0; P <= 2) -
protocolP_ext_mplla_div_clk_en (for P = 0; P <= 2) -
protocolP_ext_mplla_div_multiplier (for P = 0; P <= 2) -
protocolP_ext_mplla_fracn_ctrl (for P = 0; P <= 2) -
protocolP_ext_mplla_multiplier (for P = 0; P <= 2) -
protocolP_ext_mplla_ssc_clk_sel (for P = 0; P <= 2) -
protocolP_ext_mplla_ssc_freq_cnt_init (for P = 0; P <= 2) -
protocolP_ext_mplla_ssc_freq_cnt_peak (for P = 0; P <= 2) -
protocolP_ext_mplla_ssc_up_spread (for P = 0; P <= 2) -
protocolP_ext_mplla_tx_clk_div (for P = 0; P <= 2) -
protocolP_ext_mpllb_bandwidth (for P = 0; P <= 2) -
protocolP_ext_mpllb_div10_clk_en (for P = 0; P <= 2) -
protocolP_ext_mpllb_div8_clk_en (for P = 0; P <= 2) -
protocolP_ext_mpllb_div_clk_en (for P = 0; P <= 2) -
protocolP_ext_mpllb_div_multiplier (for P = 0; P <= 2) -
protocolP_ext_mpllb_fracn_ctrl (for P = 0; P <= 2) -
protocolP_ext_mpllb_multiplier (for P = 0; P <= 2) -
protocolP_ext_mpllb_ssc_clk_sel (for P = 0; P <= 2) -
protocolP_ext_mpllb_ssc_freq_cnt_init (for P = 0; P <= 2) -
protocolP_ext_mpllb_ssc_freq_cnt_peak (for P = 0; P <= 2) -

protocolP_ext_mpllb_ssc_up_spread (for P = 0; P <= 2) -

protocolP_ext_mpllb_tx_clk_div (for P = 0; P <= 2) -

protocolP_ext_ref_clk_div2_en (for P = 0; P <= 2) -

protocolP_ext_ref_clk_mplla_div2_en (for P = 0; P <= 2) -

protocolP_ext_ref_clk_mpllb_div2_en (for P = 0; P <= 2) -

protocolP_ext_ref_range (for P = 0; P <= 2) -

protocolP_ext_rx_adapt_mode_g1 (for P = 0; P <= 2) -

protocolP_ext_rx_adapt_mode_g2 (for P = 0; P <= 2) -

protocolP_ext_rx_adapt_mode_g3 (for P = 0; P <= 2) -

protocolP_ext_rx_adapt_mode_g4 (for P = 0; P <= 2) -

protocolP_ext_rx_adapt_sel_g1 (for P = 0; P <= 2) -

protocolP_ext_rx_adapt_sel_g2 (for P = 0; P <= 2) -

protocolP_ext_rx_adapt_sel_g3 (for P = 0; P <= 2) -

protocolP_ext_rx_adapt_sel_g4 (for P = 0; P <= 2) -

protocolP_ext_rx_cdr_ppm_max_g1 (for P = 0; P <= 2) -

protocolP_ext_rx_cdr_ppm_max_g2 (for P = 0; P <= 2) -

protocolP_ext_rx_cdr_ppm_max_g3 (for P = 0; P <= 2) -

protocolP_ext_rx_cdr_ppm_max_g4 (for P = 0; P <= 2) -

protocolP_ext_rx_cdr_vco_freqband_g1 (for P = 0; P <= 2) -

protocolP_ext_rx_cdr_vco_freqband_g2 (for P = 0; P <= 2) -

protocolP_ext_rx_cdr_vco_freqband_g3 (for P = 0; P <= 2) -

protocolP_ext_rx_cdr_vco_freqband_g4 (for P = 0; P <= 2) -

protocolP_ext_rx_cdr_vco_step_ctrl_g1 (for P = 0; P <= 2) -

protocolP_ext_rx_cdr_vco_step_ctrl_g2 (for P = 0; P <= 2) -

protocolP_ext_rx_cdr_vco_step_ctrl_g3 (for P = 0; P <= 2) -

protocolP_ext_rx_cdr_vco_step_ctrl_g4 (for P = 0; P <= 2) -

protocolP_ext_rx_cdr_vco_temp_comp_en_g1 (for P = 0; P <= 2) -

protocolP_ext_rx_cdr_vco_temp_comp_en_g2 (for P = 0; P <= 2) -

protocolP_ext_rx_cdr_vco_temp_comp_en_g3 (for P = 0; P <= 2) -

protocolP_ext_rx_cdr_vco_temp_comp_en_g4 (for P = 0; P <= 2) -	
protocolP_ext_rx_delta_iq_g1 (for P = 0; P <= 2) -	
protocolP_ext_rx_delta_iq_g2 (for P = 0; P <= 2) -	
protocolP_ext_rx_delta_iq_g3 (for P = 0; P <= 2) -	
protocolP_ext_rx_delta_iq_g4 (for P = 0; P <= 2) -	
protocolP_ext_rx_dfe_bypass_g1 (for P = 0; P <= 2) -	
protocolP_ext_rx_dfe_bypass_g2 (for P = 0; P <= 2) -	
protocolP_ext_rx_dfe_bypass_g3 (for P = 0; P <= 2) -	
protocolP_ext_rx_dfe_bypass_g4 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_att_lvl_g1 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_att_lvl_g2 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_att_lvl_g3 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_att_lvl_g4 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_ctle_boost_g1 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_ctle_boost_g2 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_ctle_boost_g3 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_ctle_boost_g4 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_ctle_pole_g1 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_ctle_pole_g2 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_ctle_pole_g3 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_ctle_pole_g4 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_dfe_tap1_g1 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_dfe_tap1_g2 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_dfe_tap1_g3 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_dfe_tap1_g4 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_vga1_gain_g1 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_vga1_gain_g2 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_vga1_gain_g3 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_vga1_gain_g4 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_vga2_gain_g1 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_vga2_gain_g2 (for P = 0; P <= 2) -	

protocolP_ext_rx_eq_vga2_gain_g3 (for P = 0; P <= 2) -	
protocolP_ext_rx_eq_vga2_gain_g4 (for P = 0; P <= 2) -	
protocolP_ext_rx_los_lfps_en (for P = 0; P <= 2) -	
protocolP_ext_rx_los_pwr_up_cnt (for P = 0; P <= 2) -	
protocolP_ext_rx_los_threshold (for P = 0; P <= 2) -	
protocolP_ext_rx_misc_g1 (for P = 0; P <= 2) -	
protocolP_ext_rx_misc_g2 (for P = 0; P <= 2) -	
protocolP_ext_rx_misc_g3 (for P = 0; P <= 2) -	
protocolP_ext_rx_misc_g4 (for P = 0; P <= 2) -	
protocolP_ext_rx_ref_ld_val_g1 (for P = 0; P <= 2) -	
protocolP_ext_rx_ref_ld_val_g2 (for P = 0; P <= 2) -	
protocolP_ext_rx_ref_ld_val_g3 (for P = 0; P <= 2) -	
protocolP_ext_rx_ref_ld_val_g4 (for P = 0; P <= 2) -	
protocolP_ext_rx_vco_ld_val_g1 (for P = 0; P <= 2) -	
protocolP_ext_rx_vco_ld_val_g2 (for P = 0; P <= 2) -	
protocolP_ext_rx_vco_ld_val_g3 (for P = 0; P <= 2) -	
protocolP_ext_rx_vco_ld_val_g4 (for P = 0; P <= 2) -	
protocolP_ext_rx_vref_ctrl (for P = 0; P <= 2) -	
protocolP_ext_sup_misc_g1 (for P = 0; P <= 2) -	
protocolP_ext_sup_misc_g2 (for P = 0; P <= 2) -	
protocolP_ext_sup_misc_g3 (for P = 0; P <= 2) -	
protocolP_ext_sup_misc_g4 (for P = 0; P <= 2) -	
protocolP_ext_tx_eq_main_g1 (for P = 0; P <= 2) -	
protocolP_ext_tx_eq_main_g2 (for P = 0; P <= 2) -	
protocolP_ext_tx_eq_main_g3 (for P = 0; P <= 2) -	
protocolP_ext_tx_eq_main_g4 (for P = 0; P <= 2) -	
protocolP_ext_tx_eq_ovrd_g1 (for P = 0; P <= 2) -	
protocolP_ext_tx_eq_ovrd_g2 (for P = 0; P <= 2) -	
protocolP_ext_tx_eq_ovrd_g3 (for P = 0; P <= 2) -	
protocolP_ext_tx_eq_ovrd_g4 (for P = 0; P <= 2) -	
protocolP_ext_tx_eq_post_g1 (for P = 0; P <= 2) -	
protocolP_ext_tx_eq_post_g2 (for P = 0; P <= 2) -	
protocolP_ext_tx_eq_post_g3 (for P = 0; P <= 2) -	
protocolP_ext_tx_eq_post_g4 (for P = 0; P <= 2) -	
protocolP_ext_tx_eq_pre_g1 (for P = 0; P <= 2) -	
protocolP_ext_tx_eq_pre_g2 (for P = 0; P <= 2) -	
protocolP_ext_tx_eq_pre_g3 (for P = 0; P <= 2) -	
protocolP_ext_tx_eq_pre_g4 (for P = 0; P <= 2) -	
protocolP_ext_tx_iboost_lvl (for P = 0; P <= 2) -	
protocolP_ext_tx_misc_g1 (for P = 0; P <= 2) -	
protocolP_ext_tx_misc_g2 (for P = 0; P <= 2) -	
protocolP_ext_tx_misc_g3 (for P = 0; P <= 2) -	
protocolP_ext_tx_misc_g4 (for P = 0; P <= 2) -	
protocolP_ext_tx_vboost_lvl (for P = 0; P <= 2) -	

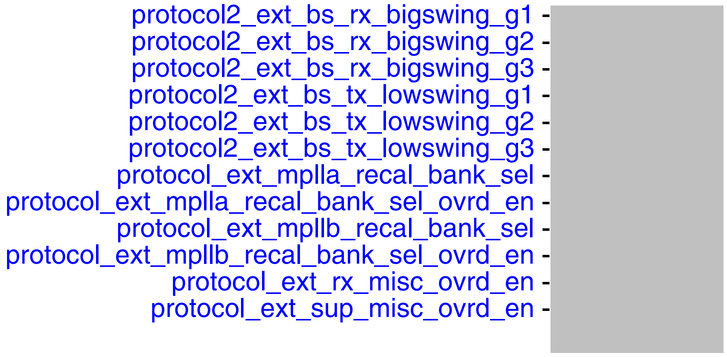

protocol2_ext_bs_rx_bigswing_g1
protocol2_ext_bs_rx_bigswing_g2
protocol2_ext_bs_rx_bigswing_g3
protocol2_ext_bs_tx_lowswing_g1
protocol2_ext_bs_tx_lowswing_g2
protocol2_ext_bs_tx_lowswing_g3
protocol_ext_mplla_recal_bank_sel
protocol_ext_mplla_recal_bank_sel_ovrd_en
protocol_ext_mpllb_recal_bank_sel
protocol_ext_mpllb_recal_bank_sel_ovrd_en
protocol_ext_rx_misc_ovrd_en
protocol_ext_sup_misc_ovrd_en

Table 4-13 External Protocol Override Signals

Port Name	I/O	Description
phy_ext_ctrl_sel	I	phy_ext_ctrl_sel is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ External overrides protocol[0,1,2]_ext_sup_misc[g1,g2,g3,g4] are enabled by protocol_ext_sup_misc_ovrd_en. ■ External override protocol_ext_mplla_recal_bank_sel[1:0] is enabled by protocol_ext_mplla_recal_bank_sel_ovrd_en. ■ External override protocol_ext_mpllb_recal_bank_sel[1:0] is enabled by protocol_ext_mpllb_recal_bank_sel_ovrd_en. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_bs_rx_bigswing (for P = 0; P <= 1)	I	protocolP_ext_bs_rx_bigswing is TX/RX boundary scan bigswing settings. ■ Sets the boundary scan swing and level settings for the PHY. ■ For recommended values, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_bs_rx_level[4:0] (for P = 0; P <= 2)	I	protocolP_ext_bs_rx_level TX/RX boundary scan level settings. ■ Sets the boundary scan swing and level settings for the PHY. ■ For recommended values, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_bs_tx_lowswing (for P = 0; P <= 1)	I	protocolP_ext_bs_tx_lowswing is TX/RX boundary scan swing settings. ■ Sets the boundary scan swing and level settings for the PHY. ■ For recommended values, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_mplla_bandwidth[15:0] (for P = 0; P <= 2)	I	protocolP_ext_mplla_bandwidth is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_mplla_div10_clk_en (for P = 0; P <= 2)	I	protocolP_ext_mplla_div10_clk_en is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_mplla_div16p5_clk_en (for P = 0; P <= 2)	I	protocolP_ext_mplla_div16p5_clk_en is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_mplla_div8_clk_en (for P = 0; P <= 2)	I	protocolP_ext_mplla_div8_clk_en is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_mplla_div_clk_en (for P = 0; P <= 2)	I	protocolP_ext_mplla_div_clk_en is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_mplla_div_multiplier[6:0] (for P = 0; P <= 2)	I	protocolP_ext_mplla_div_multiplier is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_mplla_fracn_ctrl[10:0] (for P = 0; P <= 2)	I	protocolP_ext_mplla_fracn_ctrl is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_mplla_multiplier[7:0] (for P = 0; P <= 2)	I	protocolP_ext_mplla_multiplier is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_mplla_ssc_clk_sel (for P = 0; P <= 2)	I	protocolP_ext_mplla_ssc_clk_sel is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_mplla_ssc_freq_cnt_init[11:0] (for P = 0; P <= 2)	I	protocolP_ext_mplla_ssc_freq_cnt_init is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_mplla_ssc_freq_cnt_peak[7:0] (for P = 0; P <= 2)	I	protocolP_ext_mplla_ssc_freq_cnt_peak is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_mplla_ssc_up_spread (for P = 0; P <= 2)	I	protocolP_ext_mplla_ssc_up_spread is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_mplla_tx_clk_div[1:0] (for P = 0; P <= 2)	I	protocolP_ext_mplla_tx_clk_div is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_mpllb_bandwidth[15:0] (for P = 0; P <= 2)	I	protocolP_ext_mpllb_bandwidth is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_mpllb_div10_clk_en (for P = 0; P <= 2)	I	protocolP_ext_mpllb_div10_clk_en is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_mpllb_div8_clk_en (for P = 0; P <= 2)	I	protocolP_ext_mpllb_div8_clk_en is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_mpllb_div_clk_en (for P = 0; P <= 2)	I	protocolP_ext_mpllb_div_clk_en is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_mpllb_div_multiplier[6:0] (for P = 0; P <= 2)	I	protocolP_ext_mpllb_div_multiplier is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_mpllb_fracn_ctrl[10:0] (for P = 0; P <= 2)	I	protocolP_ext_mpllb_fracn_ctrl is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_mpllb_multiplier[7:0] (for P = 0; P <= 2)	I	protocolP_ext_mpllb_multiplier is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_mpllb_ssc_clk_sel (for P = 0; P <= 2)	I	protocolP_ext_mpllb_ssc_clk_sel is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_mpllb_ssc_freq_cnt_init[11:0] (for P = 0; P <= 2)	I	protocolP_ext_mpllb_ssc_freq_cnt_init is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_mpllb_ssc_freq_cnt_peak[7:0] (for P = 0; P <= 2)	I	protocolP_ext_mpllb_ssc_freq_cnt_peak is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_mpllb_ssc_up_spread (for P = 0; P <= 2)	I	protocolP_ext_mpllb_ssc_up_spread is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_mpllb_tx_clk_div[1:0] (for P = 0; P <= 2)	I	protocolP_ext_mpllb_tx_clk_div is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_ref_clk_div2_en (for P = 0; P <= 2)	I	protocolP_ext_ref_clk_div2_en is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_ref_clk_mplla_div2_en (for P = 0; P <= 2)	I	protocolP_ext_ref_clk_mplla_div2_en is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_ref_clk_mpllb_div2_en (for P = 0; P <= 2)	I	protocolP_ext_ref_clk_mpllb_div2_en is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_ref_range[2:0] (for P = 0; P <= 2)	I	protocolP_ext_ref_range is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_adapt_mode_g1[7:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_adapt_mode_g1 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_adapt_mode_g2[7:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_adapt_mode_g2 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_adapt_mode_g3[7:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_adapt_mode_g3 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_adapt_mode_g4[7:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_adapt_mode_g4 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_adapt_sel_g1[3:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_adapt_sel_g1 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_adapt_sel_g2[3:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_adapt_sel_g2 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_adapt_sel_g3[3:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_adapt_sel_g3 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_adapt_sel_g4[3:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_adapt_sel_g4 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_cdr_ppm_max_g1[19:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_cdr_ppm_max_g1 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_cdr_ppm_max_g2[19:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_cdr_ppm_max_g2 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_cdr_ppm_max_g3[19:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_cdr_ppm_max_g3 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_cdr_ppm_max_g4[19:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_cdr_ppm_max_g4 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_cdr_vco_freqband_g1[7:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_cdr_vco_freqband_g1 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_cdr_vco_freqband_g2[7:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_cdr_vco_freqband_g2 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_cdr_vco_freqband_g3[7:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_cdr_vco_freqband_g3 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_cdr_vco_freqband_g4[7:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_cdr_vco_freqband_g4 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_cdr_vco_step_ctrl_g1[3:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_cdr_vco_step_ctrl_g1 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_cdr_vco_step_ctrl_g2[3:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_cdr_vco_step_ctrl_g2 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_cdr_vco_step_ctrl_g3[3:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_cdr_vco_step_ctrl_g3 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_cdr_vco_step_ctrl_g4[3:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_cdr_vco_step_ctrl_g4 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_cdr_vco_temp_comp_en_g1[3:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_cdr_vco_temp_comp_en_g1 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_cdr_vco_temp_comp_en_g2[3:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_cdr_vco_temp_comp_en_g2 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_cdr_vco_temp_comp_en_g3[3:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_cdr_vco_temp_comp_en_g3 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_cdr_vco_temp_comp_en_g4[3:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_cdr_vco_temp_comp_en_g4 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_delta_iq_g1[15:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_delta_iq_g1 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_delta_iq_g2[15:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_delta_iq_g2 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_delta_iq_g3[15:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_delta_iq_g3 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_delta_iq_g4[15:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_delta_iq_g4 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_dfe_bypass_g1[3:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_dfe_bypass_g1 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_dfe_bypass_g2[3:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_dfe_bypass_g2 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_dfe_bypass_g3[3:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_dfe_bypass_g3 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_dfe_bypass_g4[3:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_dfe_bypass_g4 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_att_lvl_g1[11:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_att_lvl_g1 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_att_lvl_g2[11:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_att_lvl_g2 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_att_lvl_g3[11:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_att_lvl_g3 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_att_lvl_g4[11:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_att_lvl_g4 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_ctle_boost_g1[19:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_ctle_boost_g1 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_ctle_boost_g2[19:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_ctle_boost_g2 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_ctle_boost_g3[19:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_ctle_boost_g3 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_ctle_boost_g4[19:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_ctle_boost_g4 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_ctle_pole_g1[7:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_ctle_pole_g1 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_ctle_pole_g2[7:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_ctle_pole_g2 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_ctle_pole_g3[7:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_ctle_pole_g3 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_ctle_pole_g4[7:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_ctle_pole_g4 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_dfe_tap1_g1[31:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_dfe_tap1_g1 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_dfe_tap1_g2[31:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_dfe_tap1_g2 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_dfe_tap1_g3[31:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_dfe_tap1_g3 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_dfe_tap1_g4[31:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_dfe_tap1_g4 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_vga1_gain_g1[11:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_vga1_gain_g1 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_vga1_gain_g2[11:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_vga1_gain_g2 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_vga1_gain_g3[11:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_vga1_gain_g3 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_vga1_gain_g4[11:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_vga1_gain_g4 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_vga2_gain_g1[11:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_vga2_gain_g1 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_vga2_gain_g2[11:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_vga2_gain_g2 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_vga2_gain_g3[11:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_vga2_gain_g3 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_eq_vga2_gain_g4[11:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_eq_vga2_gain_g4 is receiver equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's RX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal RX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_rx_eq_dfe_tap1_({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_rx_los_lfps_en (for P = 0; P <= 2)	I	protocolP_ext_rx_los_lfps_en is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_los_pwr_up_cnt[10:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_los_pwr_up_cnt is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_rx_los_threshold[11:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_los_threshold is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_misc_g1[31:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_misc_g1 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_misc_g2[31:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_misc_g2 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_misc_g3[31:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_misc_g3 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_misc_g4[31:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_misc_g4 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}), **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_rx_ref_ld_val_g1[6:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_ref_ld_val_g1 is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_ref_ld_val_g2[6:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_ref_ld_val_g2 is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_rx_ref_ld_val_g3[6:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_ref_ld_val_g3 is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_ref_ld_val_g4[6:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_ref_ld_val_g4 is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_rx_vco_ld_val_g1[12:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_vco_ld_val_g1 is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_vco_ld_val_g2[12:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_vco_ld_val_g2 is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_rx_vco_ld_val_g3[12:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_vco_ld_val_g3 is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_rx_vco_ld_val_g4[12:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_vco_ld_val_g4 is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_rx_vref_ctrl[4:0] (for P = 0; P <= 2)	I	protocolP_ext_rx_vref_ctrl is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_sup_misc_g1[7:0] (for P = 0; P <= 2)	I	protocolP_ext_sup_misc_g1 is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ External overrides protocol[0,1,2]_ext_sup_misc[g1,g2,g3,g4] are enabled by protocol_ext_sup_misc_ovrd_en. For more information, contact the "Support Center". Bits [2:1] control the voltage mode: ■ 'b00: 0.8-V mode ■ 'b01: 0.85-V mode ■ 'b10: 0.9-V mode ■ 'b11: 1.0-V mode ■ External override protocol_ext_mplla_recal_bank_sel[1:0] is enabled by protocol_ext_mplla_recal_bank_sel_ovrd_en. ■ External override protocol_ext_mpllb_recal_bank_sel[1:0] is enabled by protocol_ext_mpllb_recal_bank_sel_ovrd_en. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_sup_misc_g2[7:0] (for P = 0; P <= 2)	I	protocolP_ext_sup_misc_g2 is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ External overrides protocol[0,1,2]_ext_sup_misc[g1,g2,g3,g4] are enabled by protocol_ext_sup_misc_ovrd_en. For more information, contact the "Support Center". Bits [2:1] control the voltage mode: ■ 'b00: 0.8-V mode ■ 'b01: 0.85-V mode ■ 'b10: 0.9-V mode ■ 'b11: 1.0-V mode ■ External override protocol_ext_mplla_recal_bank_sel[1:0] is enabled by protocol_ext_mplla_recal_bank_sel_ovrd_en. ■ External override protocol_ext_mpllb_recal_bank_sel[1:0] is enabled by protocol_ext_mpllb_recal_bank_sel_ovrd_en. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_sup_misc_g3[7:0] (for P = 0; P <= 2)	I	protocolP_ext_sup_misc_g3 is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ External overrides protocol[0,1,2]_ext_sup_misc[g1,g2,g3,g4] are enabled by protocol_ext_sup_misc_ovrd_en. For more information, contact the "Support Center". Bits [2:1] control the voltage mode: ■ 'b00: 0.8-V mode ■ 'b01: 0.85-V mode ■ 'b10: 0.9-V mode ■ 'b11: 1.0-V mode ■ External override protocol_ext_mplla_recal_bank_sel[1:0] is enabled by protocol_ext_mplla_recal_bank_sel_ovrd_en. ■ External override protocol_ext_mpllb_recal_bank_sel[1:0] is enabled by protocol_ext_mpllb_recal_bank_sel_ovrd_en. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_sup_misc_g4[7:0] (for P = 0; P <= 2)	I	protocolP_ext_sup_misc_g4 is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ External overrides protocol[0,1,2]_ext_sup_misc[g1,g2,g3,g4] are enabled by protocol_ext_sup_misc_ovrd_en. For more information, contact the "Support Center". Bits [2:1] control the voltage mode: ■ 'b00: 0.8-V mode ■ 'b01: 0.85-V mode ■ 'b10: 0.9-V mode ■ 'b11: 1.0-V mode ■ External override protocol_ext_mplla_recal_bank_sel[1:0] is enabled by protocol_ext_mplla_recal_bank_sel_ovrd_en. ■ External override protocol_ext_mpllb_recal_bank_sel[1:0] is enabled by protocol_ext_mpllb_recal_bank_sel_ovrd_en. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_eq_main_g1[23:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_eq_main_g1 is transmitter equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's TX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal TX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], a per-protocol basis from these top-level pins when the corresponding protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_eq_main_g1({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_eq_main_g1({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_eq_main_g2[23:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_eq_main_g2 is transmitter equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's TX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal TX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], a per-protocol basis from these top-level pins when the corresponding protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_eq_main_g1({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_eq_main_g1({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_eq_main_g3[23:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_eq_main_g3 is transmitter equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's TX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal TX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], a per-protocol basis from these top-level pins when the corresponding protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_eq_main_g1({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_eq_main_g1({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_eq_main_g4[23:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_eq_main_g4 is transmitter equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's TX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal TX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], a per-protocol basis from these top-level pins when the corresponding protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_eq_main_g1({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_eq_main_g1({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_eq_ovrd_g1[3:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_eq_ovrd_g1 is transmitter equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's TX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal TX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], a per-protocol basis from these top-level pins when the corresponding protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_eq_main_g1({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_eq_main_g1({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_eq_ovrd_g2[3:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_eq_ovrd_g2 is transmitter equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's TX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal TX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], a per-protocol basis from these top-level pins when the corresponding protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_eq_main_g1({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_eq_main_g1({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_eq_ovrd_g3[3:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_eq_ovrd_g3 is transmitter equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's TX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal TX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], a per-protocol basis from these top-level pins when the corresponding protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_eq_main_g1({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_eq_main_g1({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_eq_ovrd_g4[3:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_eq_ovrd_g4 is transmitter equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's TX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal TX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], a per-protocol basis from these top-level pins when the corresponding protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_eq_main_g1({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_eq_main_g1({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_eq_post_g1[19:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_eq_post_g1 is transmitter equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's TX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal TX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], a per-protocol basis from these top-level pins when the corresponding protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_eq_main_g1({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_eq_main_g1({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_eq_post_g2[19:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_eq_post_g2 is transmitter equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's TX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal TX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], a per-protocol basis from these top-level pins when the corresponding protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_eq_main_g1({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_eq_main_g1({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_eq_post_g3[19:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_eq_post_g3 is transmitter equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's TX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal TX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], a per-protocol basis from these top-level pins when the corresponding protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_eq_main_g1({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_eq_main_g1({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_eq_post_g4[19:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_eq_post_g4 is transmitter equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's TX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal TX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], a per-protocol basis from these top-level pins when the corresponding protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_eq_main_g1({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_eq_main_g1({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_eq_pre_g1[19:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_eq_pre_g1 is transmitter equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's TX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal TX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], a per-protocol basis from these top-level pins when the corresponding protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_eq_main_g1({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_eq_main_g1({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_eq_pre_g2[19:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_eq_pre_g2 is transmitter equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's TX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal TX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], a per-protocol basis from these top-level pins when the corresponding protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_eq_main_g1({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_eq_main_g1({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_eq_pre_g3[19:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_eq_pre_g3 is transmitter equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's TX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal TX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], a per-protocol basis from these top-level pins when the corresponding protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_eq_main_g1({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_eq_main_g1({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_eq_pre_g4[19:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_eq_pre_g4 is transmitter equalization setting overrides. ■ External overrides for the per-protocol settings of the PHY's TX equalization inputs. ■ The 16G PCS internally determines the hard-coded optimal TX equalization settings for each protocol and rate. ■ However, these settings can be overwritten on protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], a per-protocol basis from these top-level pins when the corresponding protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Note:** ■ Because only protocol0 supports four rates, protocol[1,2]_ext_*_g4 inputs are not provided. ■ Because protocol1 supports only two rates, protocol1_ext_*_g3 inputs are not provided. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_eq_main_g1({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_eq_main_g1({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_iboost_lvl[15:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_iboost_lvl is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_misc_g1[31:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_misc_g1 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_misc_g2[31:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_misc_g2 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_misc_g3[31:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_misc_g3 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocolP_ext_tx_misc_g4[31:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_misc_g4 is lane-based PHY configuration setting per-protocol. ■ Lane-based external overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. However, these settings can be overwritten on a per-protocol basis from these top-level pins when the phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, and 10, respectively. ■ These signals are concatenated buses whose width is dependent on the number of lanes in the PCS (NLANES). ■ The buses are ordered from highest lane in the MSBs to lane0 in the LSBs. Example tie-off to constant value: .protocol0_ext_tx_vboost_lvl({NLANES{5'd0}}) Example with a single programmable register driving all lanes to a common value: .protocol0_ext_tx_vboost_lvl({NLANES{<REG_A>}}) **Synchronous To:** Asynchronous **Active State:** N/A
protocolP_ext_tx_vboost_lvl[2:0] (for P = 0; P <= 2)	I	protocolP_ext_tx_vboost_lvl is PHY configuration setting per-protocol. ■ External overrides for the per-protocol settings of the PHY configuration inputs. ■ (For per-protocol settings to configure the PHY, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter). ■ The PCS internally determines the hard-coded optimal settings for each protocol. ■ However, these settings can be overwritten on a per-protocol basis from these top-level pins when phy_ext_ctrl_sel input is asserted. ■ The protocol0, protocol1, and protocol2 signals correspond to the pipe_laneX_protocol[1:0] value of 00, 01, 10, respectively. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocol2_ext_bs_rx_bigswing_g1	I	protocol2_ext_bs_rx_bigswing_g1 TX/RX boundary scan bigswing settings. ■ Sets the boundary scan swing and level settings for the PHY. ■ For recommended values, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Synchronous To:** Asynchronous **Active State:** N/A
protocol2_ext_bs_rx_bigswing_g2	I	protocol2_ext_bs_rx_bigswing_g2 TX/RX boundary scan bigswing settings. ■ Sets the boundary scan swing and level settings for the PHY. ■ For recommended values, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Synchronous To:** Asynchronous **Active State:** N/A
protocol2_ext_bs_rx_bigswing_g3	I	protocol2_ext_bs_rx_bigswing_g3 TX/RX boundary scan bigswing settings. ■ Sets the boundary scan swing and level settings for the PHY. ■ For recommended values, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Synchronous To:** Asynchronous **Active State:** N/A
protocol2_ext_bs_tx_lowswing_g1	I	protocol2_ext_bs_tx_lowswing_g1 TX/RX boundary scan bigswing settings. ■ Sets the boundary scan swing and level settings for the PHY. ■ For recommended values, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Synchronous To:** Asynchronous **Active State:** N/A
protocol2_ext_bs_tx_lowswing_g2	I	protocol2_ext_bs_tx_lowswing_g2 TX/RX boundary scan bigswing settings. ■ Sets the boundary scan swing and level settings for the PHY. ■ For recommended values, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocol2_ext_bs_tx_lowswing_g3	I	protocol2_ext_bs_tx_lowswing_g3 TX/RX boundary scan bigswing settings. ■ Sets the boundary scan swing and level settings for the PHY. ■ For recommended values, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Synchronous To:** Asynchronous **Active State:** N/A
protocol_ext_mplla_recal_bank_sel[1:0]	I	protocol_ext_mplla_recal_bank_sel[1:0] is a device-based PHY configuration settings. ■ When protocol_ext_mplla_recal_bank_sel_ovrd_en = 1, protocol_ext_mplla_recal_bank_sel is used to drive the mplla_recal_bank_sel for each PHY in the device; otherwise, the PCS drives this signal. ■ For more information about mplla/b_recal_bank_sel, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter. **Synchronous To:** Asynchronous **Active State:** N/A
protocol_ext_mplla_recal_bank_sel_ovrd_en	I	protocol_ext_mplla_recal_bank_sel_ovrd_en is a device-based PHY configuration settings. ■ When protocol_ext_mplla_recal_bank_sel_ovrd_en = 1, protocol_ext_mplla_recal_bank_sel is used to drive the mplla_recal_bank_sel for each PHY in the device; otherwise, the PCS drives this signal. ■ For more information about mplla/b_recal_bank_sel, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter. **Synchronous To:** Asynchronous **Active State:** N/A
protocol_ext_mpllb_recal_bank_sel[1:0]	I	protocol_ext_mpllb_recal_bank_sel[1:0] is a device-based PHY configuration settings. ■ When protocol_ext_mpllb_recal_bank_sel_ovrd_en = 1, protocol_ext_mpllb_recal_bank_sel is used to drive the mpllb_recal_bank_sel for each PHY in the device; otherwise, the PCS drives this signal. ■ For more information about mplla/b_recal_bank_sel, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter. **Synchronous To:** Asynchronous **Active State:** N/A

Table 4-13 External Protocol Override Signals (Continued)

Port Name	I/O	Description
protocol_ext_mpllb_recal_bank_sel_ovrd_en	I	protocol_ext_mpllb_recal_bank_sel_ovrd_en is a device-based PHY configuration settings. ■ When protocol_ext_mpllb_recal_bank_sel_ovrd_en = 1, protocol_ext_mpllb_recal_bank_sel is used to drive the mpllb_recal_bank_sel for each PHY in the device; otherwise, the PCS drives this signal. ■ For more information about mplla/b_recal_bank_sel, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter. **Synchronous To:** Asynchronous **Active State:** N/A
protocol_ext_rx_misc_ovrd_en	I	This signal enables protocol*_ext_rx_misc_[g1,g2,g3,g4] and protocol0_esm_ext_rx_misc_rX. **Synchronous To:** N/A **Active State:** N/A
protocol_ext_sup_misc_ovrd_en	I	When protocol_ext_sup_misc_ovrd_en = 1, protocol[0,1,2]_ext_sup_misc[g1,g2,g3,g4] is used to drive the sup_misc input for each PHY in the device; otherwise, the PCS drives this signal. For more information about this signal, refer to the DesignWare Cores PHY databook, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Synchronous To:** Asynchronous **Active State:** N/A

4.14 Parallel Control Register Port Signals (for N = 0; N <= Nphy-1)

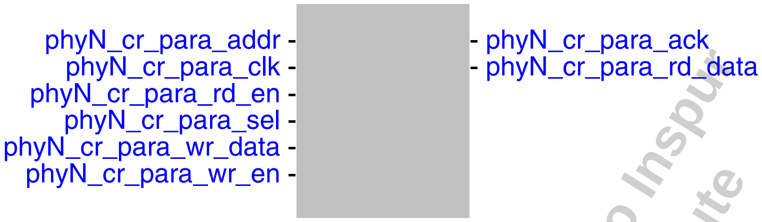


Table 4-14 Parallel Control Register Port Signals (for N = 0; N <= Nphy-1)

Port Name	I/O	Description
phyN_cr_para_ack	O	CR Read/Write acknowledgment. ■ Acknowledge signal indicating completion of the CR parallel read/write access. ■ For more information about operation during SRAM update, see the "PHY Usage and Configuration" chapter in the DesignWare Cores PHY databook. **Voltage Range:** 0-vpdig **Synchronous To:** cr_para_clk **Active State:** High
phyN_cr_para_addr[15:0]	I	CR address. **Voltage Range:** 0-vpdig **Synchronous To:** cr_para_clk **Active State:** N/A
phyN_cr_para_clk	I	CR clock. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_cr_para_rd_data[15:0]	O	CR read data. **Voltage Range:** 0-vpdig **Synchronous To:** cr_para_clk **Active State:** N/A
phyN_cr_para_rd_en	I	CR read enable. Reads from the referenced address on cr_para_addr and provides the data on cr_para_rd_data. **Voltage Range:** 0-vpdig **Synchronous To:** cr_para_clk **Active State:** High

Table 4-14 Parallel Control Register Port Signals (for N = 0; N <= Nphy-1) (Continued)

Port Name	I/O	Description
phyN_cr_para_sel	I	Control Register (CR) parallel interface select. Controls selection between JTAG and CR interfaces: ■ 0: JTAG ■ 1: Control Register (CR) This input can only be changed when the cr_para_clk and jtag_tck clock inputs are disabled. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_cr_para_wr_data[15:0]	I	CR write data. **Voltage Range:** 0-vpdig **Synchronous To:** cr_para_clk **Active State:** N/A
phyN_cr_para_wr_en	I	CR write enable. Writes the data on cr_para_wr_data to the referenced address on cr_para_addr. **Voltage Range:** 0-vpdig **Synchronous To:** cr_para_clk **Active State:** High

4.15 SRAM Control Signals (for N = 0; N <= Nphy-1)

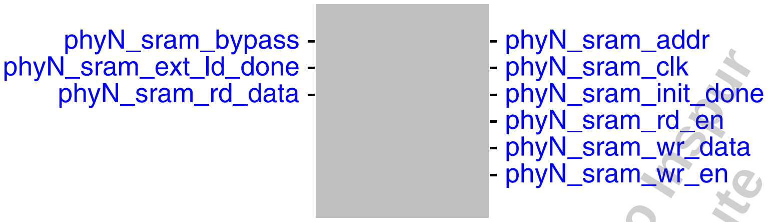


Table 4-15 SRAM Control Signals (for N = 0; N <= Nphy-1)

Port Name	I/O	Description
phyN_sram_addr[13:0]	O	SRAM address bus. ■ Address port for the SRAM. ■ Selected if macro *_SRAM_SUPPORT is defined. **Voltage Range:** 0-vpdig **Synchronous To:** sram_clk **Active State:** N/A
phyN_sram_bypass	I	SRAM bypass. ■ Control signal when asserted, bypasses the SRAM interface. In this case, the adaptation and calibration algorithms are executed from the hard wired values within the Raw PCS. ■ If SRAM is not bypassed, the internal algorithms are first loaded by Raw PCS into the SRAM at which point user can change the contents of the SRAM. ■ The updated SRAM contents are used for the adaptation and calibration routines. ■ This signal is meant to be used only for debugging purposes and must not change after phy_reset is negated. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_sram_clk	O	SRAM Clock. ■ Clock used to drive SRAM interface. ■ Selected if macro *_SRAM_SUPPORT is defined. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A

Table 4-15 SRAM Control Signals (for N = 0; N <= Nphy-1) (Continued)

Port Name	I/O	Description
phyN_sram_ext_ld_done	I	SRAM external load done. Signal asserted by user after any updates to the SRAM have been loaded. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_sram_init_done	O	SRAM Initialization done. Signal indicating that the SRAM has been initialized by the boot loader in the Raw PCS. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phyN_sram_rd_data[15:0]	I	SRAM read bus (sram_rd_data). ■ Read data port for the SRAM. ■ Selected if macro *_SRAM_SUPPORT is defined. **Voltage Range:** 0-vpdig **Synchronous To:** sram_clk **Active State:** N/A
phyN_sram_rd_en	O	SRAM read enable. ■ Read enable control to read data from SRAM. ■ Selected if macro *_SRAM_SUPPORT is defined. **Voltage Range:** 0-vpdig **Synchronous To:** sram_clk **Active State:** High
phyN_sram_wr_data[15:0]	O	SRAM read bus (sram_wr_data). ■ Write data port for the SRAM. ■ Selected if macro *_SRAM_SUPPORT is defined. **Voltage Range:** 0-vpdig **Synchronous To:** sram_clk **Active State:** N/A
phyN_sram_wr_en	O	SRAM write enable. ■ Write enable control to write data to SRAM. ■ Selected if macro *_SRAM_SUPPORT is defined. **Voltage Range:** 0-vpdig **Synchronous To:** sram_clk **Active State:** High

4.16 PHY Test Signals

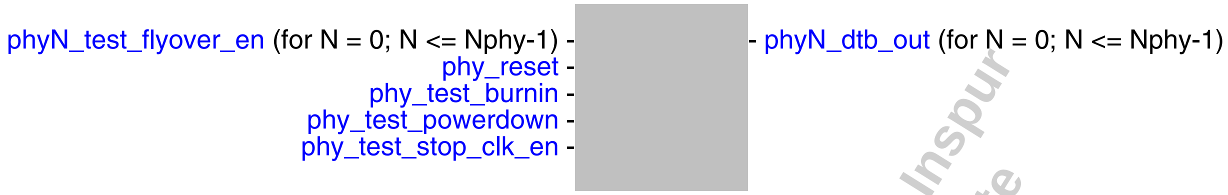

phyN_test_flyover_en (for N = 0; N <= Nphy-1) -
phy_reset -
phy_test_burnin -
phy_test_powerdown -
phy_test_stop_clk_en -
- phyN_dtb_out (for N = 0; N <= Nphy-1)

Table 4-16 PHY Test Signals

Port Name	I/O	Description
phyN_dtb_out[1:0] (for N = 0; N <= Nphy-1)	O	Digital test bus output. Used in debug mode for probing the values of various internal signals in the PHY. **Voltage Range:** 0-vpdig **Synchronous To:** N/A **Active State:** N/A
phyN_test_flyover_en (for N = 0; N <= Nphy-1)	I	Enable flyover test mode. ■ Flyover test mode enables direct access to the TX and RX bump pads, from the txX_flyover_data_p/m and rxX_flyover_data_p/m I/Os. ■ This mode can be enabled in either normal mode or scan mode. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phy_reset	I	PHY reset. ■ Asynchronously resets the core and all state machines. ■ Asserting phy_reset resets all sequential elements in the design, including control registers. ■ As a result, any register programming needs to be redone after phy_reset is de-asserted. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phy_test_burnin	I	All circuits activator. ■ Activates all circuitry in the PHY for burn-in testing. ■ This signal must not be asserted while test_powerdown signal is set to 1'b1. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High

Table 4-16 PHY Test Signals (Continued)

Port Name	I/O	Description
phy_test_powerdown	I	All circuits Power-Down control. Powers down all circuitry in the PHY for IDDQ testing. **Note:** The PHY is not functional in this mode and must be reset after this signal is de-asserted. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phy_test_stop_clk_en	I	Stop clock test mode enable. Reserved **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High

4.17 Resistor Tune Signals (Common Block Group)

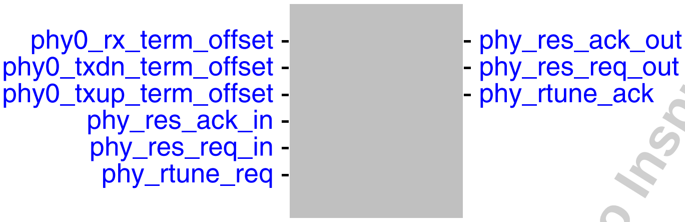


Table 4-17 Resistor Tune Signals (Common Block Group)

Port Name	I/O	Description
phy0_rx_term_offset[4:0]	I	Offset for RX Termination. ■ Specifies an additional fixed offset to calibrated RX termination value. ■ This is a signed input with 2's complement encoding. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** N/A
phy0_txdn_term_offset[8:0]	I	Offset for TX Down Termination. ■ Specifies an additional fixed offset to calibrated TX down termination value. ■ This is a signed input with 2's complement encoding. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** N/A
phy0_txup_term_offset[8:0]	I	Offset for TX Up Termination. ■ Specifies an additional fixed offset to calibrated TX up termination value. ■ This is a signed input with 2's complement encoding. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** N/A
phy_res_ack_in	I	Arbitration lines for sharing single reference resistor (res_ack_in). Arbitration lines used for sharing a single-reference resistor among multiple cores that are connected to a single set of pads. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High

Table 4-17 Resistor Tune Signals (Common Block Group) (Continued)

Port Name	I/O	Description
phy_res_ack_out	O	Arbitration lines for sharing single reference resistor (res_ack_out). Arbitration lines used for sharing a single reference resistor among multiple cores that are connected to a single set of pads. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phy_res_req_in	I	Arbitration lines for sharing single reference resistor (res_req_in). Arbitration lines used for sharing a single-reference resistor among multiple cores that are connected to a single set of pads. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phy_res_req_out	O	Arbitration lines for sharing single reference resistor (res_req_out). Arbitration lines used for sharing a single reference resistor among multiple cores that are connected to a single set of pads. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phy_rtune_ack	O	Resistor tune acknowledge. Indicates that a resistor tune has completed. **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High
phy_rtune_req	I	Resistor tune request. Assertion triggers a resistor tune request (if one is not already in progress). **Voltage Range:** 0-vpdig **Synchronous To:** Asynchronous **Active State:** High

5 Usage Model

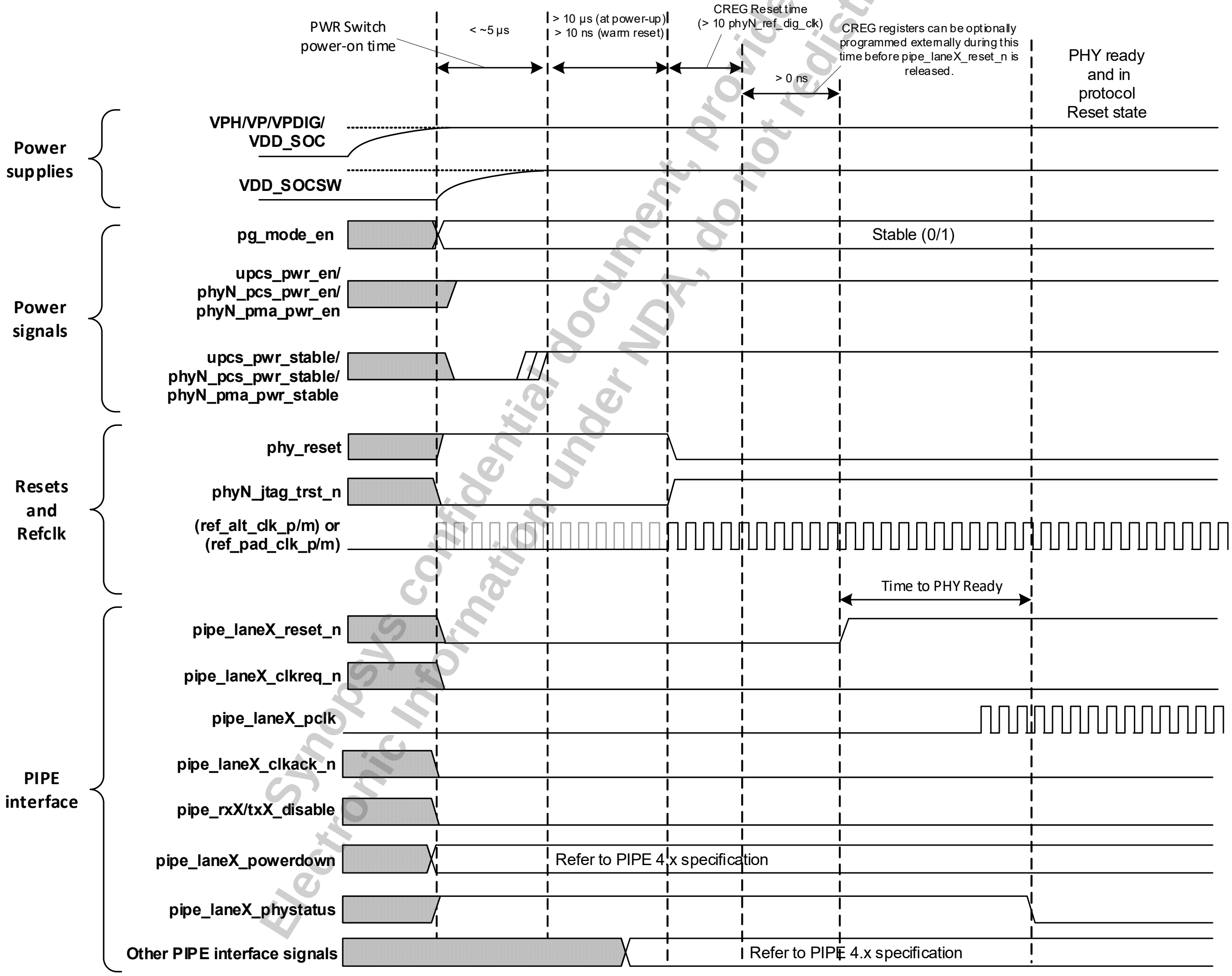
This chapter includes the following sections:

5.1 Power-Up Sequence

The following figure shows the PCIe 4.0 PCS power-up reset sequence along with the power supply ramp-up. The figure shows signals on the PIPE wrapper module that encapsulate the PCIe 4.0 PCS and the PHY. (For a description of the PHY signals, refer to DesignWare Cores PCIe 4.0 PHY databook, "Signal Descriptions" chapter.)

The PCIe 4.0 PCS and the PCIe 4.0 PHY support power gating. The figure also shows the state of the power switch enable signals during the reset sequence.

Figure 5-1 Power-up Reset Timing



VPH, VP, and VPDIG are power supply inputs of the PMA macro in the PHY. VDD_SOC is the always-ON core supply for the Raw PCS (in the PHY) and the PCIe 4.0 PCS.

Note VPH, VP, and VPDIG power supplies in the PCS map to vph, vp, and vpdig in the PHY, respectively.

VDD_SOCSW refers to the switched power supply for the Raw PCS and the PCIe 4.0 PCS, which is controlled by the power enable signals, upcs_pwr_en (UPCS) and phyN_pcs_pwr_en/phyN_pma_pwr_en (PHY). For information about implementing power gating in the PHY and PCIe 4.0 PCS, see "Power-Gating Support" on page 260.

To support debug of the design, you might need to update control registers in the PCIe 4.0 PHY before running any operation using the PCIe 4.0 PCS and the PCIe 4.0 PHY. You can program control registers in the PHY (using the PHY's CR parallel or JTAG interface) by de-asserting the phy_reset input while keeping the lane resets (pipe_laneX_reset_n) asserted.

Note Before accessing any registers in the PHY, a register reset time of 10 phyN_ref_dig_clk cycles is required.

5.2 Lane Reset Sequence

Each lane can be independently reset using the pipe_laneX_reset_n signal. For a multi-lane link, a common PIPE reset signal from the MAC can be connected to all lanes on the PCIe 4.0 PCS operating as part of the same link. Figure 5-1 on page 228 shows the PIPE interface signals involved in lane reset assertion and exit.

5.3 Clocks

For details on clock architecture and clock frequencies, see “Clocks” on page 22. The clock outputs from the PCIe 4.0 PCS and the PHYs are dependent on both PIPE signals and some side-band signals.

5.3.1 PCIe 4.0 PCS Output Clocks

The following table(s) provides the conditions under which the PCIe 4.0 PCS clock outputs are available.

Table 5-1 PCIe 4.0 PCS Clock Dependency

I/O	Signal	Conditions X = Don’t care, D = Disabled, E = Enabled								
Inputs	phy_reset	X	1	0	0	0	0	0	0	0
	pipe_laneX_reset_n	X	X	0	1	1	1	1	1	1
	Reference clock	D	E	E	E	E	E	E	E	E
	ext_pclk_req	X	X	X	ext_pclk_req=1	0	0	0	0	0
	upcs_pipe_config[0]	X	X	X	X	X	X	X	0	1
	LaneX (disabled/enabled)	X	X	X	X	X	X	E	D	D
	pipe_laneX_powerdown	X	(controller reset value) PCIe: ■ P1	(controller reset value) PCIe: ■ P1		PCIe: ■ P1.CPM ■ P2.CPM ■ P2..NOBEACON ■ Slumber.PG	PCIe: ■ P2	PCIe: ■ P0 ■ P0s ■ P1 ■	PCIe: ■ P0 ■ P0s ■ P1 ■	PCIe: ■ P0 ■ P0s ■ P1
Outputs	pipe_laneX_pclk pipe_laneX_max_pclk	D	D	D	E	D	D	E	D	E
	pcs_laneX_ref_clk_en (to PIPE_GLUE)	X	1	1	1	0	1	1	1	1

Notes:
- LaneX is disabled when pipe_txX_elecidle = 1 and pipe_txX_compliance = 1.
- Reference clock = phyN_ref_[alt, pad]_clk_[p,m]
- When the DWC_UPCS_PERLINK_CLK_ARCH parameter is defined, the MAC must drive each laneX input for all lanes that belong to the same link to the same identical value in each lane.

Clock availability is indicated on the PIPE interface using the pipe_laneX_phystatus signal. When the pipe_laneX_pclk/pipe_laneX_max_pclk clocks transition from Disabled (D) state to Enabled (E) state, these clocks are available before the de-assertion (0) of pipe_laneX_phystatus if pipe_laneX_phystatus is already asserted (1) either in the Clock Disabled state or before the assertion (1) of pipe_laneX_phystatus if pipe_laneX_phystatus is de-asserted (0) in the Clock Disabled state.

Note that the PCIe 4.0 PCS clock outputs, pipe_laneX_pclk/pipe_laneX_max_pclk, are available per lane and can be used by the MAC to sample/launch the synchronous signals on the PIPE interface for the corresponding lane X. This option requires independent clock architectures in the MAC on a per-lane basis. To ease this implementation in the MAC, the PCIe 4.0 PCS clock outputs for each lane are designed to be phase-synchronous to other lanes configured to be part of the same link (for information about lane configuration, see "PHY Configuration Overrides" on page 254). Therefore, the PCIe 4.0 PCS clocks from one lane can be used to sample/launch the synchronous signals on the PIPE interface for other lanes that are part of the same link. If the lane's Down-Configure feature is required, it is recommended that you set the upcs_pipe_config[0] signal to 1.

Doing so enables the PCIe 4.0 PCS clocks for lane X to be enabled for the following states even when lane X is disabled (for clock availability conditions, see Table 6-2 on page 435) as a result of down-configuration:

- PCIe: P0, P0s, and P1

If upcs_pipe_config[0] is reset to 0 and DWC_UPCS_PERLINK_CLK_ARCH is not defined, the PIPE interface signals are ignored by the PCIe 4.0 PCS, and the PCIe 4.0 PCS clock from lane X might get disabled for the down-configured lane. Consequently, the MAC cannot use the clock from lane X as a common clock source to sample/launch the synchronous PIPE interface signals for other active lanes on the same link.

When DWC_UPCS_PERLINK_CLK_ARCH is defined, regardless of upcs_pipe_config[0], PCIe 4.0 PCS clocks for laneX are enabled during the following states—even when laneX is disabled as a result of down-configuration—as long as a lane belonging to the same link is not disabled:

- PCIe: P0, P0s, and P1

5.3.2 PHY Output Clocks

This section describes the output clocks from the PHY.

5.3.2.1 MPLL Output Clocks

In addition to the PCIe 4.0 PCS output clocks (pipe_laneX_pclk/pipe_laneX_max_pclk), the PHY's MPLL output clocks can also be used by logic external to the PCIe 4.0 PCS and can be kept alive irrespective of the state of the per-lane PIPE control signals. These clock outputs are tapped from the PHY interface as shown in the PIPE wrapper included in the package release (./macro/rtl/dwc_*_upcs_*_pipe.v).

The following table provides the conditions under which the MPLL clocks are available. As indicated in the table, the phyN_mpll[a,b]_force_en input forces the MPLL clocks to be enabled—assuming phy_reset is de-asserted and a reference clock input to the PHY is available.

Table 5-2 PHY MPLL Output Clock Dependency

I/O	Signal	Conditions X = Don't care, D = Disabled, E = Enabled									
Inputs	phyN_mpll[a,b]_force_en	X	X	X	1	0	0	0	0	0	0
	phy_reset	X	1	0	0	0	0	0	0	0	0
	pipe_laneX_reset_n	X	X	0	1	1	1	1	1	1	1
	Reference clock	D	E	E	E	E	E	E	E	E	E
	ext_pclk_req	X	X	X	X	For the remaining conditions, see Table 5-1 on page 230. Because MPLL clocks are required to generate PCIe 4.0 PCS clock outputs, all conditions from Table 5-1 and the corresponding states of pipe_laneX_pclk/ pipe_laneX_max_pclk are also applicable to phyN_mpll[a,b,]_*_clk with the following additional requirements: ■ phyN_mpll[a,b]_*_clk signals are Enabled (E) when **any** lane Xs using MPLL clocks from PHY*N* meet the conditions required to enable pipe_laneX_pclk/pipe_laneX_max_pclk as specified in Table 5-1. ■ phyN_mpll[a,b]_*_clk signals are Disabled (D) when **all** the lane Xs using MPLL clocks from PHY*N* meet the conditions required to disable pipe_laneX_pclk/pipe_laneX_max_pclk as specified in Table 5-1.					
	upcs_pipe_config[0]	X	X	X	X						
	LaneX (Disabled/Enabled)	X	X	X	X						
	pipe_laneX_powerdown	X	(controller reset value) PCIe: P1	(controller reset value) PCIe: P1	X						
Output Clocks	phyN_mpll[a,b]_*_clk	D	D	D	E						

Notes:
- LaneX is disabled when pipe_txX_elecidle = 1 and pipe_txX_compliance = 1.
- Reference clock = phyN_ref_[alt, pad]_clk_[p,m]

MPLL clock availability is indicated by the phyN_mpll[a,b]_state outputs from the PHY. When the phyN_mpll[a,b]_state outputs are asserted (1), the phyN_mpll[a,b]_*_clk MPLL clocks are available.

5.3.2.2 Reference Clock Outputs

The reference clock outputs from the PHY, phyN_ref_dig_clk/phyN_ref_dig_fr_clk, can also be used by logic external to the PCIe 4.0 PCS.

When any pcs_laneX_ref_clk_en in Table 5-1 on page 230 is 1b for laneX using MPLLs clocks from PHYN, the phyN_ref_clk_en signals are 1b.

When all pcs_laneX_ref_clk_en in Table 5-1 on page 230 are 0b for laneX using MPLLs clocks from PHYN, the phyN_ref_clk_en signals are 0b.

Refer to the PHY databook for conditions for enabling phyN_ref_dig_clk and phyN_ref_dig_fr_clk.

5.3.3 Reference Clock Input

The PCIe 4.0 PCS generates the clock configuration settings for the PHY based on the frequency of the reference clock supplied to the PHYs. The reference clock frequency supplied to the PHY must be defined as a compile macro to the PCIe 4.0 PCS. The following table lists the supported macros corresponding to various reference clock frequencies.

Table 5-3 PCIe Reference Clock Macros

Reference clock Frequency Macro (`define)	Reference Clock Frequency (MHz)
REF_25M	25
Default	100

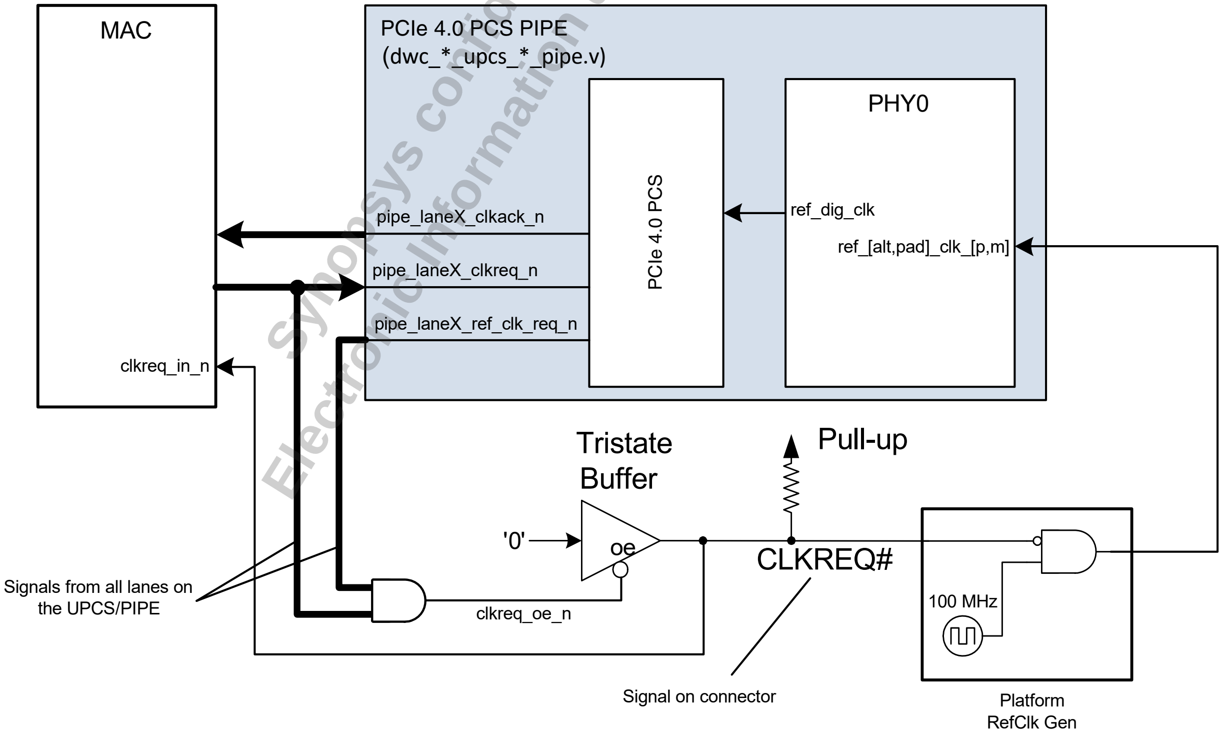
PCIe: The PCIe 4.0 PCS supports low power states where the reference clock input to the PHY (phyN_ref_[alt, pad]_clk_[p,m]) can be turned off; for example:

- P1.CPM, P1.1, P1.2, P2.CPM, and P2.NOBEACON.

To support this feature, handshake signals are provided on the PCS to send and acknowledge requests to gate the reference clock. Gate the reference clock to the PHY only if all PCIe 4.0 PCS lanes connected to the PHY are in a state where the reference clock can be turned off.

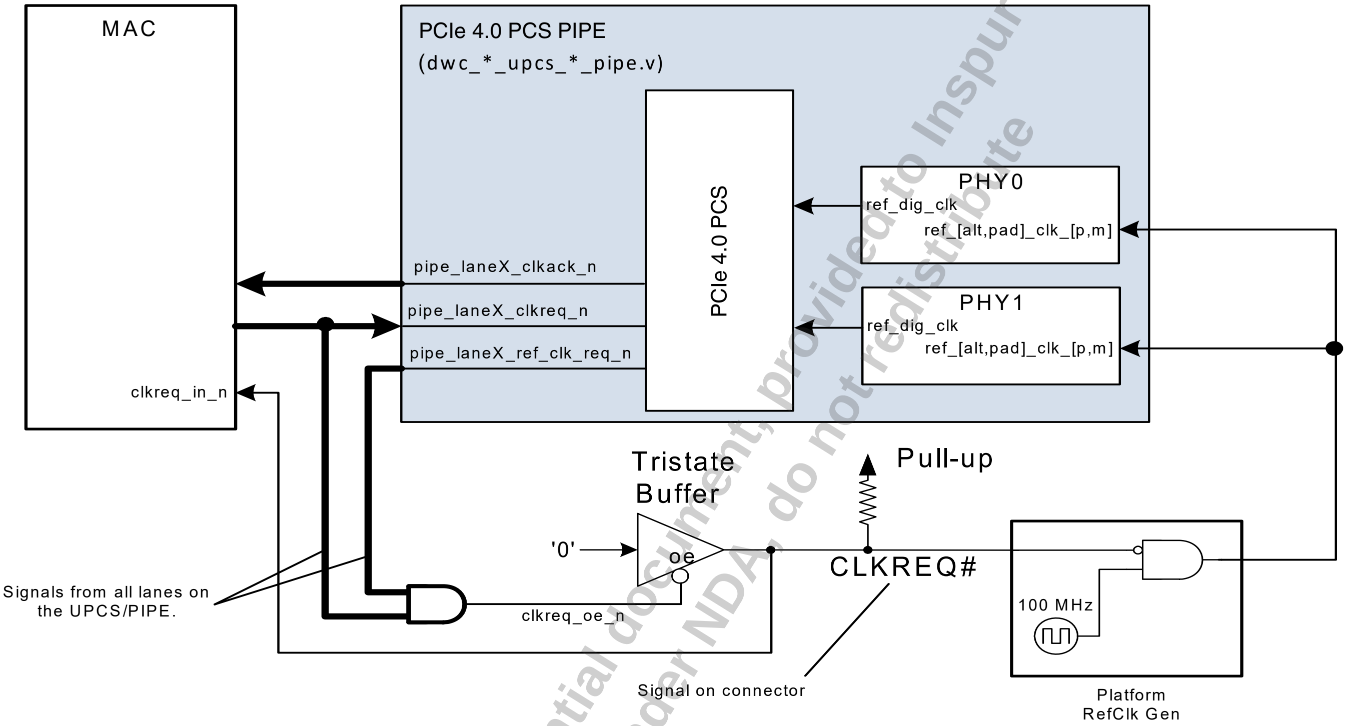
The following figure shows the reference clock topology for a single PHY and one or more (bifurcation) links on the PCIe 4.0 PCS.

Figure 5-2 Reference Clock Topology for One PHY and One or More Links



The following figure shows the reference clock topology for two or more PHYs and one link on the PCIe 4.0 PCS.

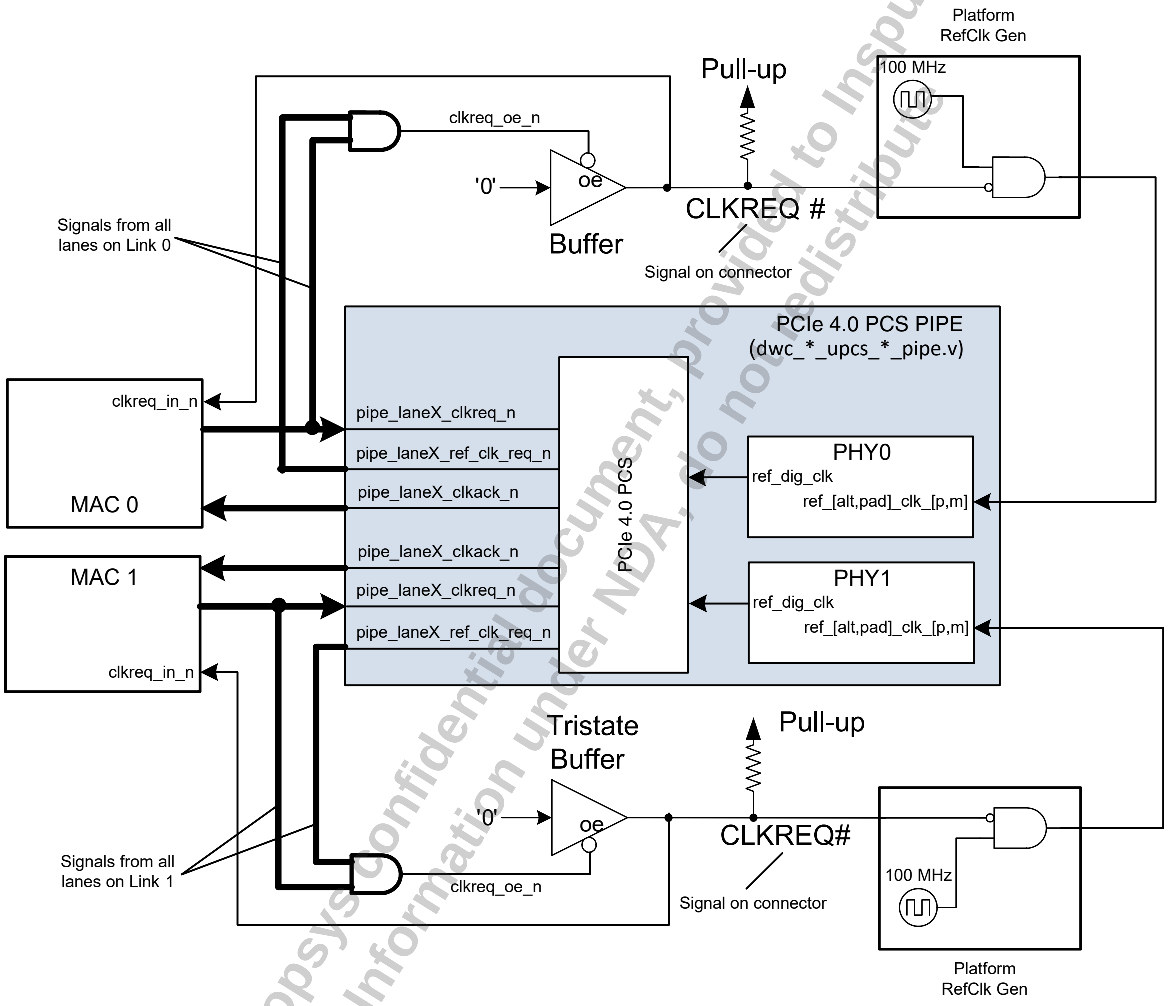
Figure 5-3 Reference Clock Topology for Two or More PHYs and One Link



Generation of the common CLKREQ# signal is the same as in Figure 5-2.

The following figure shows the reference clock topology for two or more PHYs and two or more links on the PCIe 4.0 PCS. In this figure, it is assumed that all lanes for link 0 are connected to PHY0 and all lanes for link 1 are connected to PHY1.

Figure 5-4 Reference Clock Topology for Two or More Links and Two or More PHYs



5.4 Datapath

The datapath is active when the lane is in a P0 state (pipe_laneX_powerdown[3:0]=4'h0), pipe_txX_elecidle is de-asserted, and pipe_rxX_valid is asserted. Table 5-4 provides TX datapath latencies for the PCIe PCS, and Table 5-5 provides RX datapath latencies for the PCIe PCS in the various supported modes. In all cases, the latency is shown for pipe_laneX_if_width[1:0] = 2’b01 (16-bit mode).

Table 5-4 TX Datapath Latency

16-bit PIPE Interface, 16-bit PCS Datapath, 4,096 Payload (UI)	
Rate	**Latency**
PCIe Gen1	48
PCIe Gen2	48
PCIe Gen3	80
PCIe Gen4	80

Table 5-5 RX Datapath Latency

16-bit PIPE Interface, 16-bit PCS Datapath, 4,096 Payload (UI)						
	pipe_rxX_es_mode = HalfFull		**pipe_rxX_es_mode = NomEmpty**		**pipe_rxX_es_mode = comRefClk**	
Rate	**Min Latency**	**Max Latency**	**Min Latency**	**Max Latency**	**Min Latency**	**Max Latency**
PCIe Gen1 SRNS	417.0	457.0	168.3	208.3	221.3	221.3
PCIe Gen2 SRNS	399.5	407.5	154.5	162.5	208.5	208.5
PCIe Gen3 SRNS	361.6	369.6	168.8	176.8	220.0	220.0
PCIe Gen4 SRNS	342.4	346.4	179.2	183.2	252.8	252.8
PCIe Gen1 SRIS	573.5	597.5	152.0	176.0	N/A	N/A
PCIe Gen2 SRIS	555.0	567.0	141.5	153.5	N/A	N/A
PCIe Gen3 SRIS	472.8	484.8	176.0	188.0	N/A	N/A
PCIe Gen4 SRIS	489.6	495.6	188.8	194.8	N/A	N/A

5.5 Power State Transitions

The PCIe 4.0 PCS and PCIe 4.0 PHY support the following power states:

- **PCIe**: P0, P0s, P1, P2, P1.CPM, P1.1, P1.2, P2.CPM, and P2.NOBEACON.

  The transition to and from the P0, P0s, P1, and P2 power states are indicated using the pipe_laneX_phystatus output from the PCIe 4.0 PCS. The transition to and from the other power states is indicated using either the pipe_laneX_phystatus output (PIPE 4.4.1/4.3) or the side-band signals (PIPE 4.2).

  PIPE 4.4.1 adds support for the following transitions:

  - P0 -> P1.CPM
  - P1.CPM -> P0
  - P0 -> P2.CPM
  - P2.CPM -> P1
  - P1 -> P2.CPM
  - P2.CPM -> P0

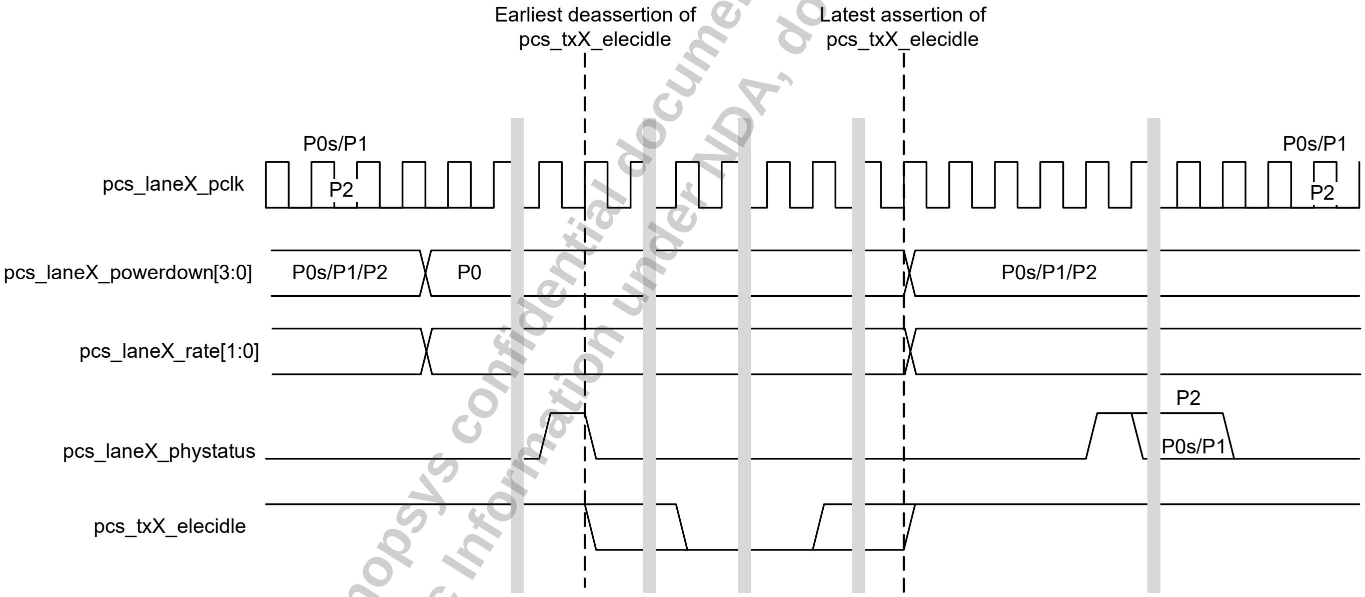
5.5.1 Power State Transition Timing Diagrams

The timing diagram for PIPE interface signals match the PIPE specification.

> **Note** The power state and/or rate change transitions take effect after the TX datapath pipeline is internally cleared, enabling the MAC to change the TX controls in the same clock cycle as changing the power state and rate inputs.
>
> The PCIe 4.0 PHY does not support this statement from *PHY Interface For the PCI Express, SATA, and USB 3.1 Architectures*, version 4.4.1/4.3: “Some PHY architectures may allow a speed change and a power state change to occur at the same time as a rate and/or width and/or PCLK rate change.”

The following figure(s) show(s) an example waveform illustrating entry to and exit from a P0 power state along with a rate transition.

Figure 5-5 PCIe: P0 Entry and Exit Timing



The pipe_laneX_pclk signal is disabled (0) in the following state(s), assuming other conditions do not enable the signal (for PCIe 4.0 PCS clock dependencies, see Table 6-2 on page 435):

- PCIe: P1.CPM, P2, P2.CPM, P2.NOBEACON

In this case, when transitioning into the following state(s), the PCIe 4.0 PCS is guaranteed to generate at least one rising clock edge on pipe_laneX_pclk after the assertion of pipe_laneX_phystatus:

- PCIe: P1.CPM, P2, P2.CPM (from P0), P2.NOBEACON (from P0)

5.5.1.1 PCIe L1 Substate Transitions

The PCIe 4.0 PCS has two interfaces to support the L1 substate for PCIe4 applications: PIPE4.2 (using side-band signals) and PIPE 4.4.1/4.3. The states on the PCIe 4.0 PCS are referred to as P1.1 and P1.2. P1.2 is the lowest power state in which a particular lane can be placed.

For details on the generation of reference clock control CLKREQ# for PCIe applications, see "Reference Clock Input" on page 235.

For a MAC that supports only the PIPE 4.2 specification, P1 substate entry and exit are supported using side-band signals. Figure 5-6 and Figure 5-7 show timings for P1.2 entry and exit using side-band signals. Timing for a P1.1 entry and exit are similar to a P1.2 entry and exit, with the exception that in a P1.1 case, pipe_txX_disable remains at 0.

For a MAC that supports the PIPE 4.4.1 specification, P1 substate entry and exit are supported using side-band signals. Figure 5-10 on page 243 shows timing for P1.1/P1.2 entry and exit using side-band signals. Timing for a P1.1 entry and exit is similar to a P1.2 entry and exit, with the exception that in the case of P1.1, pipe_txX_disable remains at 0.

Figure 5-6 PCIe P1.2 Entry and Locally Initiated Exit Using Side-Band Signals (PIPE 4.2)

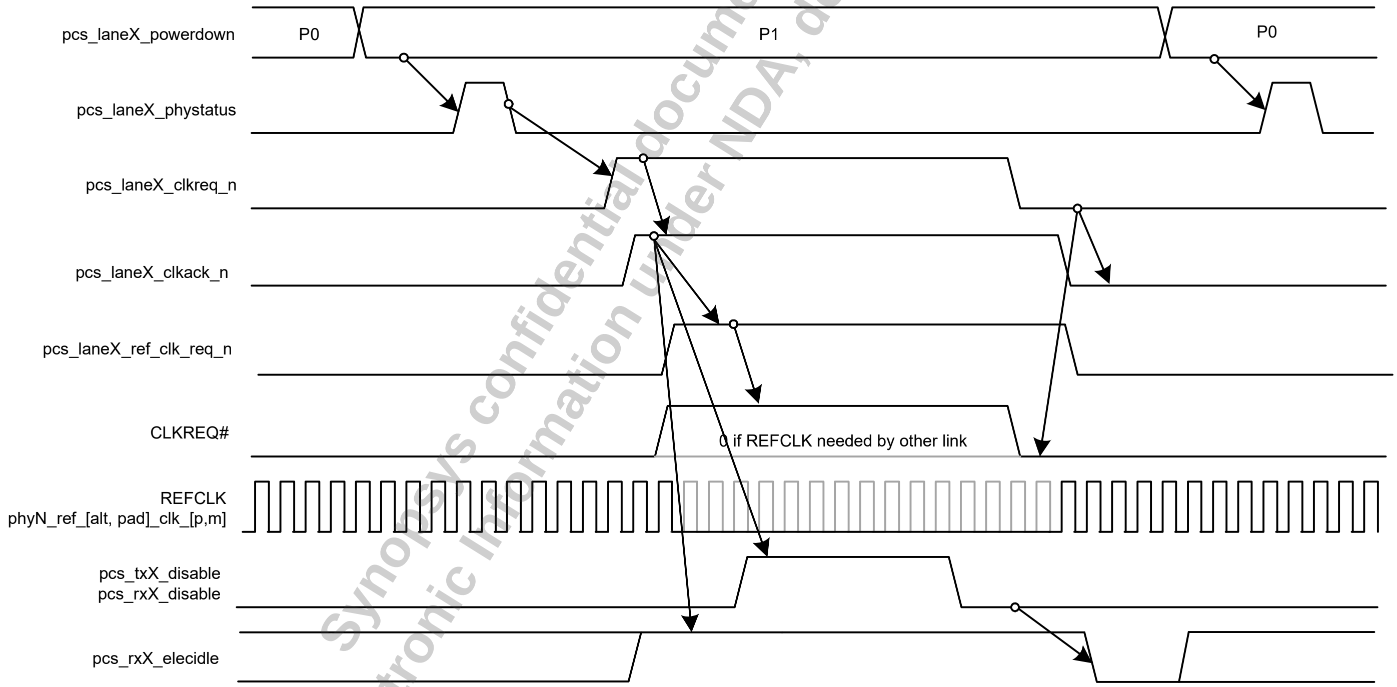


Figure 5-7 P1.2 Entry and Remotely Initiated Exit Using Side-Band Signals (PIPE 4.2)

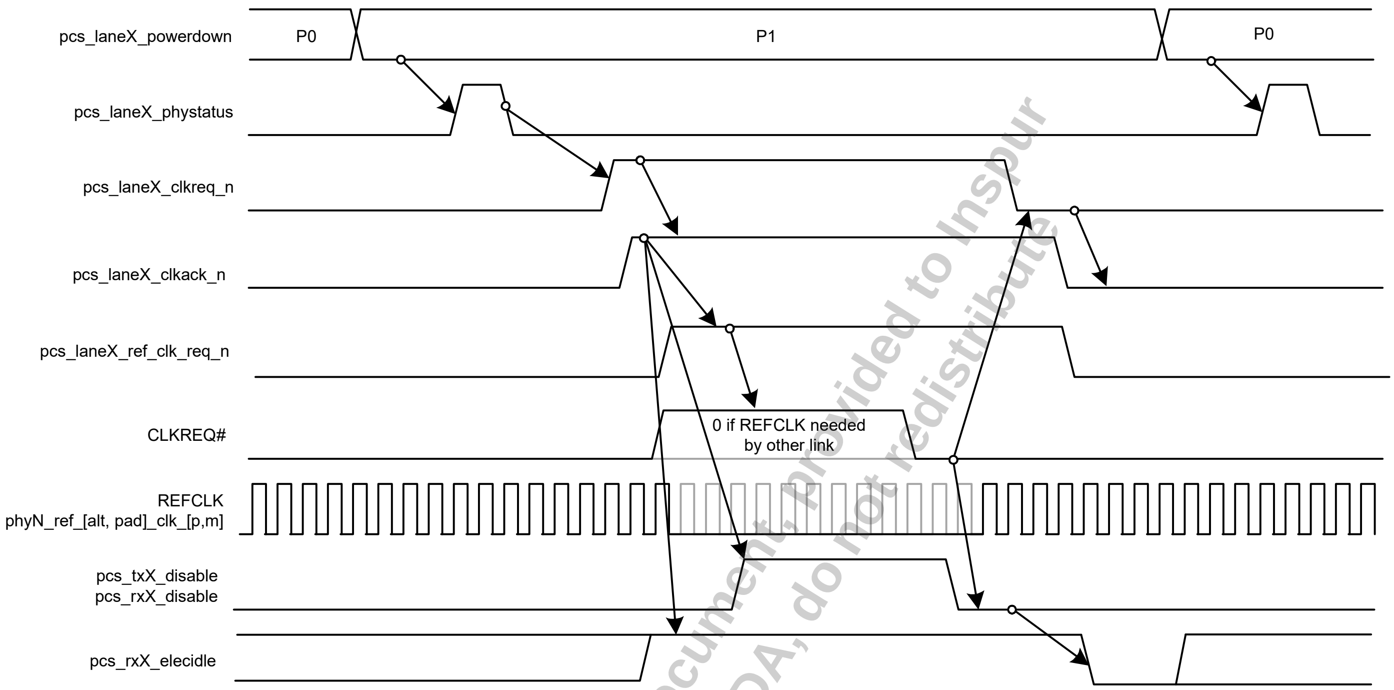


Figure 5-8 and Figure 5-9 show timings for P1.2 entry and exit using a PIPE 4.3 interface.

Figure 5-8 P1.2 Entry and Locally Initiated Exit Using PIPE 4.3 Interface

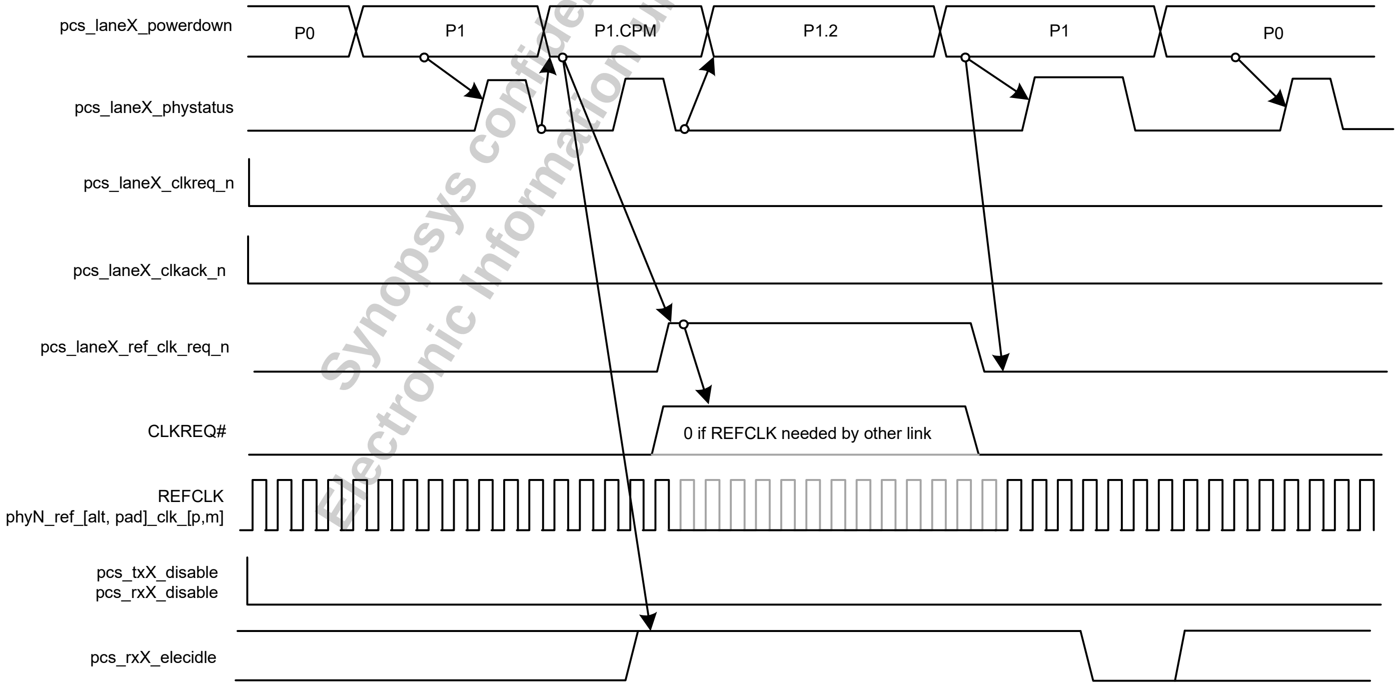
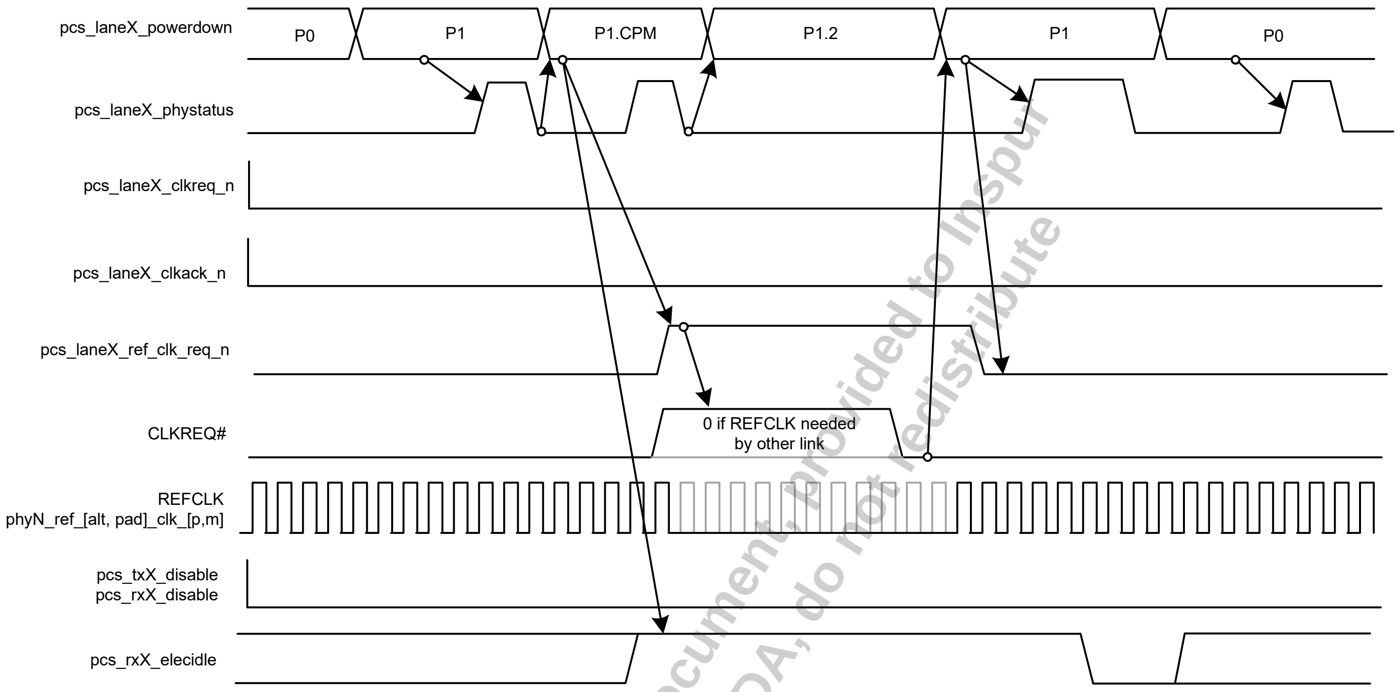


Figure 5-9 P1.2 Entry and Remotely Initiated Exit Using PIPE 4.3 Interface



Timings for a P1.1 entry and exit are similar to the P1.2 entry and exit, with the exception that in a P1.1 case, pcs_laneX_powerdown transitions from P1.CPM to P1.1.

Note Timings in Figure 5-6 on page 241 through Figure 5-10 are valid for both cases of power gating; that is, when power gating in the PCIe 4.0 PCS and PCIe 4.0 PHY is enabled (pg_mode_en = 1) and when power gating is disabled (pg_mode_en = 0).

Figure 5-10 P1.1/P1.2 Entry and Locally Initiated Exit Using PIPE 4.4.1 Interface

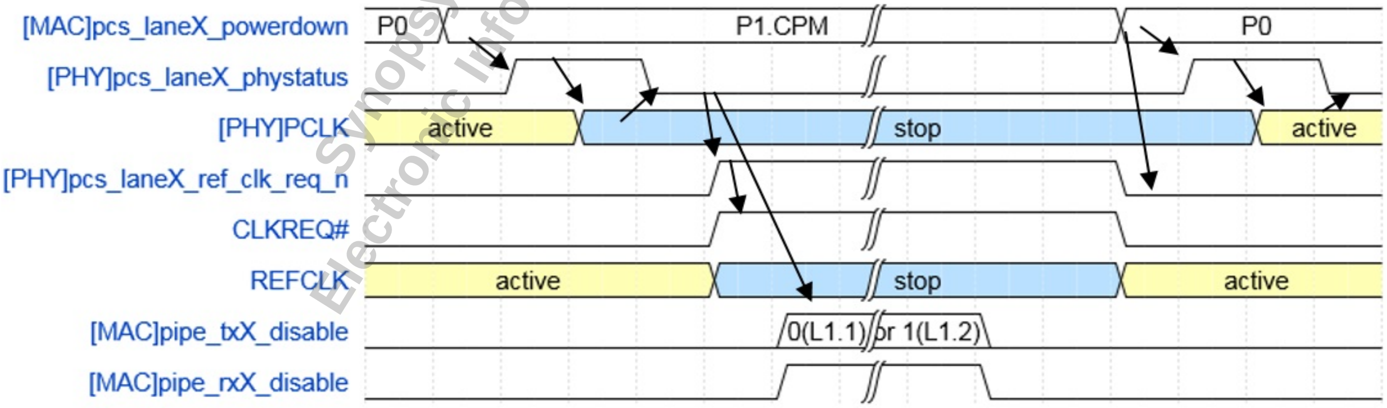
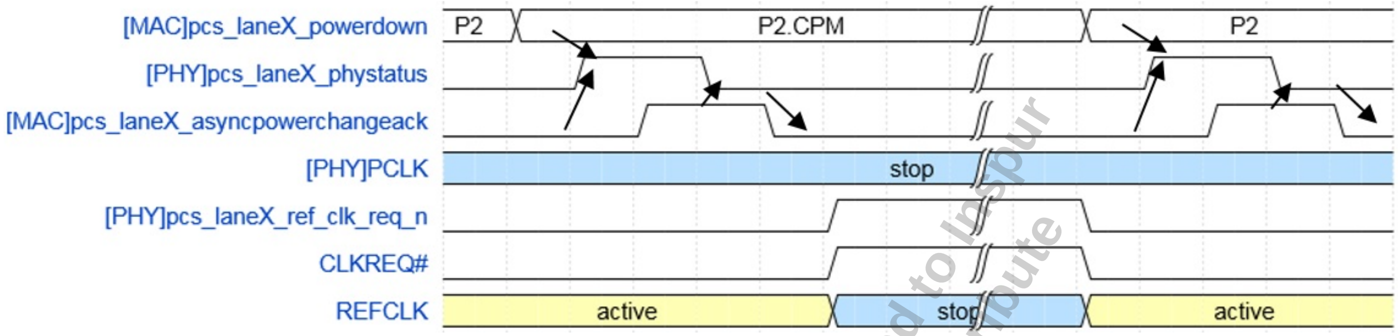


Figure 5-11 P2 to P2.CPM Transition Using PIPE 4.4.1 Interface


[MAC]pcs_laneX_powerdown
[PHY]pcs_laneX_physstatus
[MAC]pcs_laneX_asyncpowerchangeack
[PHY]PCLK
[PHY]pcs_laneX_ref_clk_req_n
CLKREQ#
REFCLK
P2
P2.CPM
P2
stop
active
stop
active

5.5.2 Power State Transition Times

The amount of time that the PCIe PCS takes to acknowledge completion (via pipe_laneX_phystatus) in response to a pipe_laneX_powerdown[3:0] transition is referred to as the power state transition time. Power state transition times are dependent on the reference clock frequency and the data rate.

The PCIe PCS transition times are the PHY's transition time plus 0.2us.

5.6 Clock Architecture and Link Selection

The PCIe 4.0 PCS supports two clock architectures: Per-Lane and Per-Link. If the DWC_UPCS_PERLINK_CLK_ARCH parameter is not defined, the clock-selection logic is implemented per lane. When DWC_UPCS_PERLINK_CLK_ARCH is defined the clock-selection logic is implemented per link. If a design needs only one, two, or four links, it is best to choose the Per-Link clock architecture. This clock architecture simplifies Clock Tree Synthesis (CTS) by reducing the number of clocks in the design.

To use the Per-Link clock architecture, you must define DWC_UPCS_PERLINK_CLK_ARCH at compile time and set DWC_UPCS_NLINKS to 1, 2, or 4. Also, when DWC_UPCS_NLINKS > 1 and DWC_UPCS_NLINKS > 2 respectively, you must set the DWC_UPCS_NLINKS_GTR1 and DWC_UPCS_GTR2 parameters.

When DWC_UPCS_NLINKS = 1, a single use mode is possible at runtime: all lanes belong to the same link. In this configuration, upcs_pipe_config[2:1] must be constant at runtime and equal to 0 at all times.

When DWC_UPCS_NLINKS = 2, two use modes are possible at runtime: (1) all lanes belong to the same link; (2) half of the lanes (lower lanes) belongs to one link, while the other half (upper lanes) belongs to another link. In the case of (1), upcs_pipe_config[2:1] must be constant at runtime and equal to 0. In the case of (2), upcs_pipe_config[2:1] must be constant at runtime and equal to 1. Uneven or asymmetric repartition of lanes to links is not supported (for example, x4x12 with 16 lanes is not supported).

DWC_UPCS_NLINKS = 3 is not supported.

When DWC_UPCS_NLINKS = 4, three use modes are possible at runtime: (1) all lanes belong to the same link; (2) half of the lanes (lower lanes) belongs to one link, while the other half (upper lanes) belongs to another link; (3) a quarter of the lanes (lower) belongs to one link, the next quarter to another link, and so on up to four links. For these cases, upcs_pipe_config[2:1] must be constant at runtime and equal to 0, 1, or 2, respectively. Uneven or asymmetric repartition of lanes to links is not supported (for example, x1x2x4x1 with eight lanes is not supported).

For a description of the parameters mentioned in this section, see "Device Configuration" on page 17.

5.7 PHY Bifurcation Support

Splitting different lanes in the same PHY across different links is referred to as PHY bifurcation. PHY bifurcation in totally independent links is not supported, because the control inputs to the PHY for reset, reference clock enable, and power enable are shared within the PHY by all the lanes belonging to that PHY. For details on restrictions regarding bifurcation support in the PHY, refer to the PCIe 4.0 PHY databook, "Aggregation and Bifurcation Support" section. When any lane is asserting pipe_laneX_ref_clk_req_n, REFCLK must not be removed. When any PHY or UPCS is asserting power enable (phyN_pcs_pwr_en, phyN_pma_pwr_en, upcs_pwr_en), power must remain applied.

If the DWC_UPCS_PERLINK_CLK_ARCH parameter is undefined, each PCIe 4.0 PCS lane can be independently configured to be part of a particular link using the pipe_laneX_phy_src_sel[3:0] input.

If DWC_UPCS_PERLINK_CLK_ARCH is defined, bifurcation is limited. Please see section 4.x "Per-lane or Per-link Clock architecture selection." For details on the number of links supported, see "Clock Architecture and Link Selection" on page 246.

5.8 PHY Aggregation Support

The PCIe 4.0 PCS can be connected to a maximum of four PHYs, so that the number of lanes on the PCS are equal to the cumulative number of lanes on the PHYs. The PHY release package includes the PCS versions that can be generated from the PHY included in the release.

For example, the release package for a x2 PHY includes the following PCS versions:

- x2 PCS (./upcs/rtl/dwc_*_upcs_x2_*_upcs_pcs_x2_x2.v)
- x4 PCS (./upcs/rtl/dwc_*_upcs_x2_*_upcs_pcs_x4_x2x2.v)
- x8 PCS (./upcs/rtl/dwc_*_upcs_x2_*_upcs_pcs_x8_x2x2x2x2.v)

The corresponding PIPE wrappers instantiating the PCS and the PHYs included in the package are as follows:

- x2 PIPE (./macro/rtl/dwc_*_upcs_x2_*_ x2_x2_pipe.v)
- x4 PIPE (./macro/rtl/dwc_*_upcs_x2_*_ x4_x2x2_pipe.v)
- x8 PIPE (./macro/rtl/dwc_*_upcs_x2_*_ x8_x2x2x2x2_pipe.v)

Combining different instances of the PHY to operate as a common link is referred to as PHY aggregation. The PCIe 4.0 PCS supports PHY aggregation through the pipe_laneX_phy_src_sel[1:0] input, which selects the PHY to be used to source the MPLL and reference clock inputs for each laneX (for an illustration, see Figure 2-1 on page 23). Using pipe_laneX_phy_src_sel[1:0], you can configure the PCS lanes to operate as one or more links—with each link operating with an independent rate (pipe_laneX_rate) and protocol (pipe_laneX_protocol)—provided that all lanes sourced by the same PHY are part of the same link. For details on the number of links supported, see "Clock Architecture and Link Selection" on page 246.

The following table lists some of the PHY aggregation options that can be obtained for various release packages.

Table 5-6 PHY Aggregation Options

	Bifurcation/Aggregation Options																	
	phy_laneX_phy_src_sel																**Description**	**Mnemonic**
X =	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
upcs_pipe_x16_x4x4x4x4	3	3	3	3	2	2	2	2	1	1	1	1	0	0	0	0	Four links with four PHYs	x4_x4_x4_x4
	2	2	2	2	2	2	2	2	1	1	1	1	0	0	0	0	Three links, of which one obtained aggregating phy2 and 3	[x4_x4]_x4_x4
	3	3	3	3	1	1	1	1	1	1	1	1	0	0	0	0	Three links, of which one obtained aggregating phy1 and 2	x4_[x4_x4]_x4
	3	3	3	3	2	2	2	2	0	0	0	0	0	0	0	0	Three links, of which one obtained aggregating phy0 and 1	x4_x4_[x4_x4]
	2	2	2	2	2	2	2	2	0	0	0	0	0	0	0	0	Two links, obtained aggregating phy0 and 1, and phy2 and 3, respectively.	[x4_x4]_[x4_x4]
	3	3	3	3	0	0	0	0	0	0	0	0	0	0	0	0	Two links, of which one obtained aggregating phy0, 1, and 2	x4_[x4_x4_x4]
	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	Two links, of which one obtained aggregating phy1, 2, and 3	[x4_x4_x4]_x4
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	One link (no bifurcation), obtained aggregating phy0, 1, 2, and 3	[x4_x4_x4_x4]

Note: To facilitate readability, some aggregation options have been omitted. Options obtainable by aggregating non-contiguous PHYs (for example PHY0 and PHY3) have been omitted. Also omitted are options where the source PHY of a link is not the lowest numbered PHY (for example, when aggregating PHY2 and PHY3 only list the case where PHY2 is the source PHY, the case where PHY3 is the source PHY is omitted).

Table 5-6 PHY Aggregation Options (Continued)

	Bifurcation/Aggregation Options																	
	phy_laneX_phy_src_sel																Description	Mnemonic
X =	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
upcs_pipe_x8_x2x2x2x2									3	3	2	2	1	1	0	0	Four links with four PHYs	x2_x2_x2_x2
									2	2	2	2	1	1	0	0	…	[x2_x2]_x2_x2
									3	3	1	1	1	1	0	0	…	x2_[x2_x2]_x2
									3	3	2	2	0	0	0	0	…	x2_x2_[x2_x2]
									2	2	2	2	0	0	0	0	…	[x2_x2]_[x2_x2]
									3	3	0	0	0	0	0	0	…	
									1	1	1	1	1	1	0	0	…	
									0	0	0	0	0	0	0	0	…	[x2_x2_x2_x2]
upcs_pipe_x8_x4x4									1	1	1	1	0	0	0	0	Two links with two PHYs	x4_x4
									0	0	0	0	0	0	0	0	One link (no bifurcation) aggregating phy0 and 1	[x4_x4]

Note: To facilitate readability, some aggregation options have been omitted. Options obtainable by aggregating non-contiguous PHYs (for example PHY0 and PHY3) have been omitted. Also omitted are options where the source PHY of a link is not the lowest numbered PHY (for example, when aggregating PHY2 and PHY3 only list the case where PHY2 is the source PHY, the case where PHY3 is the source PHY is omitted).

Table 5-6 PHY Aggregation Options (Continued)

	Bifurcation/Aggregation Options																	
	phy_laneX_phy_src_sel																**Description**	**Mnemonic**
X =	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
upcs_pipe_x4_x1x1x1x1													3	2	1	0	Four links with four PHYs	x1_x1_x1_x1
																	…	[x1_x1]_x1_x1
																	…	x1_[x1_x1]_x1
																	…	x1_x1_[x1_x1]
																	…	[x1_x1]_[x1_x1]
																	…	
																	…	
																	…	[x1_x1_x1_x1]
upcs_pipe_x4_x2x2													1	1	0	0	Two links with two PHYs	x2_x2
													0	0	0	0	One link (no bifurcation) aggregating phy0 and 1	[x2_x2]
upcs_pipe_x4_x4													0	0	0	0	No aggregation/bifurcation options	x4
upcs_pipe_x2_x1x1															1	0	Two links with two PHYs	x1_x1
															0	0	One link (no bifurcation) aggregating phy0 and 1	[x1_x1]
upcs_pipe_x2_x2															0	0	No aggregation/bifurcation options	x2
upcs_pipe_x1_x1																0	No aggregation/bifurcation options	x1

Note: To facilitate readability, some aggregation options have been omitted. Options obtainable by aggregating non-contiguous PHYs (for example PHY0 and PHY3) have been omitted. Also omitted are options where the source PHY of a link is not the lowest numbered PHY (for example, when aggregating PHY2 and PHY3 only list the case where PHY2 is the source PHY, the case where PHY3 is the source PHY is omitted).

5.9 Lane Disabling and Up/Down-Configuration

The PCIe 4.0 PCS supports active disabling and enabling of lanes in a link.

5.9.1 RX Standby Mode

A lane in an active link can be disabled by setting pipe_txX_elecidle = 1 and pipe_rxX_standby = 1 while keeping pipe_laneX_powerdown set to P0. This setting maintains the TX common mode on the lane while turning off the transmit driver and disabling the CDR—keeping the rest of receiver circuitry on. The disabled lanes can be enabled again with minimal supported wake-up time by setting pipe_txX_elecidle = 0 and pipe_rxX_standby = 0.

5.9.2 PIPE Lane Disable

A lane in an active link can be disabled by setting pipe_txX_elecidle = 1 and pipe_tx1_compliance = 1.

In this state, the PHY's TX and RX are placed in the following equivalent state(s) (TX and RX circuits off):

- PCIe: P2

This state is a significantly lower power state as compared to disabling the lane using pipe_txX_elecidle = 1 and pipe_rxX_standby = 1. The disabled lanes can be enabled again according to the requirements defined in the PIPE 4.4.1/4.3 specification, "Multi-lane PIPE – PCI Express Mode" section. Therefore, the MAC must either assert pipe_laneX_reset_n or set pipe_laneX_powerdown to P1 at the same time as exiting the disabled state.

Exception

The PIPE 4.4.1/4.3 specification states that the PHY should ignore the PIPE signals when a lane is disabled using pipe_txX_elecidle = 1 and pipe_tx1_compliance = 1.

When in such a disabled state, the PCIe 4.0 PCS supports the following two modes:

- **upcs_pipe_config[0] = 0**: The PCIe 4.0 PCS ignores PIPE interface signals. In this mode, the pipe_laneX_pcs_clk output is gated.
- **upcs_pipe_config[0] = 1**: The PCIe 4.0 PCS monitors the PIPE interface signal, pipe_laneX_powerdown.

  Therefore, if pipe_laneX_powerdown is set to the following states, the pipe_laneX_pcs_clk output is available.

  - PCIe: P0/P0s/P1

  This mode enables the MAC to continue using the pipe_laneX_pcs_clk output from a particular lane in a link to sample/launch PIPE signals for other active lanes in the same link, resulting in a simpler clock architecture in the MAC. Without this mode, the MAC is required to use the pipe_laneX_pcs_clk output of each lane to sample/launch data of the same lane. For details on this requirement, see "PCIe 4.0 PCS Output Clocks" on page 230.

5.9.3 P1.2 State

A lane can also be disabled by placing the lane in a P1.2 state (for information about P.1.2 entry and exit, see "PCIe L1 Substate Transitions" on page 241). If all other lanes in the link and all lanes on one PHY are in a P1.2 state, this mode is the most power-efficient mode, because in this case the reference clock for that PHY can be turned off.

5.10 PHY Configuration Overrides

The PCIe 4.0 PCS drives the configuration inputs to the PHY using predefined and prevalidated settings.

These settings can be overridden using primary inputs, protocol[0,1,2]_ext_*_[g1,g2,g3,g4], when phy_ext_ctrl_sel = 1.

- protocol[0,1,2] refer to PHY settings that correspond to pipe_laneX_protocol[1:0] = 0, 1, and 2, respectively.
- g[1,2,3,4] refer to PHY settings that correspond to (up to) four rates within the selected protocol.

Because protocol1 (pipe_laneX_protocol = 01) supports only two rates, protocol1_ext_*_[g3,g4] inputs do not exist. Similarly, because protocol2 (pcs_laneX_protocol = 10) supports only three rates, protocol2_ext_*_g4 inputs do not exist. These overrides provide you with the flexibility to change PHY configuration settings in the SoC implementation.

Some override settings are lane-specific, while others are specific to each PHY.

- For lane-specific configuration settings, the PCIe 4.0 PCS uses the per-lane protocol (pipe_laneX_protocol) and rate (pipe_laneX_rate) inputs to select the protocol and rate-specific configuration settings for lane X.
- For PHY-specific configuration settings (for example, MPLL controls for the PHY), the PCIe 4.0 PCS uses the protocol (pipe_laneX_protocol) and rate (pipe_laneX_rate) inputs for the lane with the lowest lane index and connected to the PHY, in order to select the protocol and rate-specific configuration settings for that PHY.

  For example, for the aggregation option of {x4 PHY0, x2 PHY1, x2 PHY2} in Table 5-6 on page 249, for pcs_lane0_protocol, pcs_lane4_protocol, and pcs_lane6_protocol, select the MPLL settings for PHY0, PHY1, and PHY2, respectively.

Note It is mandatory that you drive the protocol[0,1,2]_ext_*, phy_ext_ctrl_sel, protocol_ext_mpll{a,b}_recal_bank_sel, protocol_ext_mpll{a,b}_recal_bank_sel_ovrd_en, and protocol_ext_sup_misc_ovrd_en inputs using external, programmable registers in your SoC/ASIC with the phy_ext_ctrl_sel, protocol_ext_mpll{a,b}_recal_bank_sel_ovrd_en, and protocol_ext_sup_misc_ovrd_en default values set to 0.

5.11 Transmitter Equalization Settings

The PHY transmitter equalization settings are controlled by the pcs_txX_deemph[17:0], protocol[0,1,2]_ext_tx_eq_[pre,main,post]_[g1,g2,g3,g4], and protocol[0,1,2]_ext_tx_eq_ovrd_[g1,g2,g3,g4] inputs of the PCIe 4.0 PCS.

The pcs_txX_deemph[17:0] settings that correspond to supported equalization settings can be derived from settings of the txX_eq_main[5:0], txX_eq_pre[5:0], and txX_eq_post[5:0] inputs on the PCIe 4.0 PHY, with the following mappings or any legal value with current FS/LF (provided on pcs_txX_eq_fs/pcs_txX_eq_lf) for rules a, b, and c of the PCIe base specification:

- PCS pcs_txX_deemph[5:0] = PHY txX_eq_pre[5:0] / 4
- PCS pcs_txX_deemph[11:6] = PHY txX_eq_main[5:0]
- PCS pcs_txX_deemph[17:12] = PHY txX_eq_post[5:0] / 4

For more information about TX equalization settings, refer to the DesignWare Cores PCIe 4.0 PHY databook, “TX EQ Initialization and Presets” section.

For PCIe3/4 operation, the PCS maps the TxMargin and TxPreset inputs on the PIPE to the pcs_txX_eq_preset_coeff[17:0] PIPE signal—using the pcs_txX_eq_preset_coeff_req and pcs_txX_eq_preset_coeff_vld PIPE control signals. Note that in this case, the MAC must apply the value on pcs_txX_eq_preset_coeff[17:0] back to the PCS through the pcs_txX_deemph[17:0] input.

5.12 Lane Margining at Receiver

The PCIe 4.0 PCS supports lane margining at receiver based on the definition in PCI Express Base Specification, revision 4.0, version 1.0. The following table shows the specific Lane Margining features supported.

Table 5-7 Lane Margining Parameters

Parameter	Supported Value	Description
RXMarginingVoltageSupported	0	The PHY indicates where it supports voltage margining, encoded as follows: ■ 0: No support ■ 1: Support
RXMarginingSamplingRateVoltage	N/A PHY does not support voltage margining.	Percentage of bits margined during Voltage Margining mode, calculated as 1 / 64 * (Sampling_Rate[5:0] + 1) Valid values: 0–63
RXMarginingSamplingRateTiming	63	Percentage of bits margined during timing margining mode, calculated as 1 / 64 * (Sampling_Rate[5:0] + 1) Valid values: 0–63
RXMarginingIndependentLeftRight	1	The PHY indicates whether it supports independent left and right time margining, encoded as follows: ■ 0: No support ■ 1: Support
RXMarginingIndependentUpDown	0	The PHY indicates whether it supports independent up and down voltage margining, encoded as follows: ■ 0: No support ■ 1: Support
RXMarginingIndependentErrorSampler	0	The PHY indicates whether it supports an error sampler independent from the main sampler to enable higher BER's to be measured, encoded as follows: ■ 0: No support ■ 1: Support
RXMarginingVoltageSteps	N/A	Total number of voltage steps, minimum range ± 50 mV A value of zero indicates that voltage margining is not supported. Valid non-zero values: 32–127

Table 5-7 Lane Margining Parameters (Continued)

Parameter	Supported Value	Description
RXMarginingTimingSteps	9	Total number of timing steps, minimum range ± .2 UI Valid values: 8–63
RXMarginingMaxVoltageOffset	N/A	Offset at maximum step value as percentage of 1 volt Valid values: 5–50
RXMarginingMaxTimingOffset	20	Offset at maximum step value as percentage of nominal UI Valid values: 20–50
RXMarginingMaxLanes	0-15	Maximum number of lanes that can be margined simultaneously Valid values:1–32 Recommended value: Number of lanes the PHY supports
RXMarginingSampleReportingMethod	1	Indicates whether a sample frequency or a sample count is reported, encoded as follows: ■ 0: Sample count reported. ■ 1: Sample frequency reported.
RXMarginingMaxTimingOffsetChange	2	Maximum number of steps margin offset that can be changed with one command during timing margining Valid values: 1–127
RXMarginingMaxVoltageOffsetChange	N/A PHY does not support voltage margining.	Maximum number of steps margin offset that can be changed with one command during voltage margining Valid values: 1–127
RXMessageBusWriteBufferDepth	2	Number of write buffer entries that the PHY has implemented in order to receive writes from the MAC, where one entry can hold the 3 bytes of information associated with each write transaction.
TXMessageBusMinWriteBufferDepth	1	Minimum number of write buffer entries that the PHY expects the MAC to implement in order to receive writes from the PHY. Valid values: 0–8 The MAC can implement more than the minimum required by the PHY; however, there might not be any benefit in doing so.

When using a Synopsys controller, the controller uses MSampleReportingMethod = 1 and counts only logical errors in MErrorCount.

The PHY does not count errors when margining; therefore, the PCS does not implement Writing MAC Registers "Address 1h: RX Margin Status1" or "Address 1h: RX Margin Status2" as specified in the PIPE 4.4.1 specification. The controller is expected to count these errors and report them in the appropriate registers.

To communicate margining commands between the PHY and a controller, the Register Bus interface (RBI) is used. To set the PHY registers from controller, pcs_laneX_m2p_messagebus[7:0] is used. To set MAC registers from the PCIe 4.0 PCS, pcs_laneX_p2m_messagebus[7:0] is used. The PCIe 4.0 PCS supports only "write committed non posted" and "write_ack" commands. The PIPE 4.4.1 PCS supports all message bus commands.

If "write committed non posted" is requested by the controller through pcs_laneX_m2p_messagebus[7:0], the PCIe 4.0 PCS responds "write_ack" through pcs_laneX_p2m_messagebus[7:0]. If "write committed non posted" is requested by Enterprise 16G PCS through pcs_laneX_p2m_messagebus[7:0], the controller is expected to respond "write_ack" through pcs_laneX_m2p_messagebus[7:0].

The PHY does not support voltage margining. If a voltage change is requested, the PCS responds with "Margin Nak".

The PCS responds with "Margin Nak" in the following cases:

- "PHY REG 0x000[0]Start Margin" is updated from 0b to 1b when "PHY REG 0x000[1]Margin Voltage or Timing"= 0b (voltage).
- "PHY REG 0x000[1]Margin Voltage or Timing" is updated from 1b (timing) to 0b (voltage) when "PHY REG 0x000[0]Start Margin"= 1b.
- "PHY REG 0x001[6:0]Margin Offset" is updated from any value to any value when "PHY REG 0x000[0]Start Margin"= 1b and "PHY REG 0x000[1]Margin Voltage or Timing"= 0b (voltage).

The following is a basic lane margining sequence:

PIPE 4.3 Start Margining

1. Controller sets PHY register 0x001[6:0] "Rx Margin Offset" field to testing value. This value should be equal to or less than MNumTimingSteps.
2. Controller sets PHY register 0x000[2:1] "Rx Margin Direction" field to 00b (left) or 01b (right).
3. Controller sets PHY register 0x000[0] "Rx Margin Start" field to 1b.
4. PCIe 4.0 PCS requests PCIe 4.0 PHY to update the lane margining setting and waits for completion from PCIe 4.0 PHY.
5. PCIe 4.0 PCS sets MAC register 0x000[0] "Rx Margin Status" field to 1b.

PIPE 4.4.1 Start Margining

1. Controller uses write-uncommitted command to set PHY register 0x001[6:0] "Margin Offset" field to testing value (this value should be equal to or less than NumTimingSteps) and sets bit [7] "Margin Direction" field to 0b (left) or 1b (right).
2. Controller uses write-committed command to set PHY register 0x000[1] "Margin Voltage or Timing" field to 1'b1 to indicate Timing and sets bit [0] "Start Margin" field to 1b.

3. PCIe 4.0 PCS requests PCIe 4.0 PHY to update the lane margining setting and waits for completion from PCIe 4.0 PHY.
4. PCIe 4.0 PCS uses write-committed command to set MAC register 0x000[0] “Margin Status” field to 1b.

PIPE 4.4.1 or PIPE 4.3 Offset Change

1. Controller sets PHY register 0x001[6:0] “Rx Margin Offset” field to new testing value. This value should be equal to or less than MNumTimingSteps.
2. PCIe 4.0 PCS requests Enterprise 16G PHY to update the lane margining setting and waits for completion from PCIe 4.0 PHY.
3. PCIe 4.0 PCS sets MAC register 0x000[0] “Rx Margin Status” field to 1b.

PIPE 4.4.1 or PIPE 4.3 Stop Margining

1. Controller sets PHY register 0x000[0] “Rx Margin Start” field to 0b.
2. PCIe 4.0 PCS requests PCIe 4.0 PHY to update the lane margining setting and waits for the completion from PCIe 4.0 PHY.
3. PCIe 4.0 PCS sets MAC Register 0x000[0] “Rx Margin Status” field to 1b.

5.13 Power-Gating Support

For PCIe operation, the PCIe 4.0 PCS and RAW_PCS (in the PHY) support one always-on power domain "VDD" along with multiple power-gated domains. The power switches for the PCIe 4.0 PCS and RAW_PCS portion of the PHY are external to the IP and control the supply to the power-gated domains.

The Always-on (AON) hierarchy in the design (PCS + PHY) is as follows:

- dwc_*_upcs_*_pipe.dwc_*_upcs_pcs_x*.dwc_*_upcs_pipe_ctl_x*
- dwc_*_upcs_*_pipe.dwc_*_upcs_pcs_x*.dwc_*_upcs_clk_ctl
- dwc_*_phy_x*.dwc_*_pcs_raw_x*.dwc_*_pcs_raw_aon_x*

Figure 5-12 on page 262 shows control signal connectivity for power gating the PCIe 4.0 PCS and the PHY. The power switches for the Raw PCS and the PCIe 4.0 PCS can be implemented either as single switches (as shown in the figure) or distributed switches. In either case, there must be one power-stable signal generated by the single switch or daisy chained through distributed switches. Assert the upcs_pwr_stable input only after the supplies ramp up. Figure 5-11 on page 244 shows the entry and exit sequence for PIPE 4.4.1 with power gating enabled.

The Raw PCS generates power-gating enables for the Raw PCS switch(es) as well as the PMA switch. The phyN_pcs_pwr_stable and phyN_pma_pwr_stable inputs must be asserted only after the supplies ramp up. Power-Gating mode is enabled when the pg_mode_en input to the PCIe 4.0 PCS and the PHY is set to 1, and when the PCIe 4.0 PCS is placed into a P1.2 power state (for information about P1.2 entry and exit, see "PCIe L1 Substate Transitions" on page 241). When pg_mode_en = 0, the enables for the power switches are held asserted to a 1.

Power gating is supported by defining IPNAME_TOP_PG_PINS or by using the UPF flow to insert power pins. The UPF files are provided in the upcs/upf/ and phy/upf/ release directories. UPF version 2.0 and version 1.0 are both supported. These files define the power domains, power supplies, ports, nets, supply sets, and the power state table that is used for simulation and synthesis. For the power-gating methodology, the expectation is that the top-level wrappers and the always-on (AON) blocks are connected to the ungated power domain, while the majority of sub-blocks are on "islands" of power-gated logic. The following tables describe the power supplies and power state table. For details on voltages for each supply, refer to the PCIe 4.0 PHY databook.

Table 5-8 Power Supplies

Supply	Description
VDD	ASIC core supply
VSS	ASIC ground
phyN_GD	IP ground
phyN_VPH	High-voltage IO supply
phyN_VP	PMA low-voltage supply
phyN_VPDIG	PMA digital supply
VPTXX	PMA transmitter termination voltage supply

Table 5-8 Power Supplies (Continued)

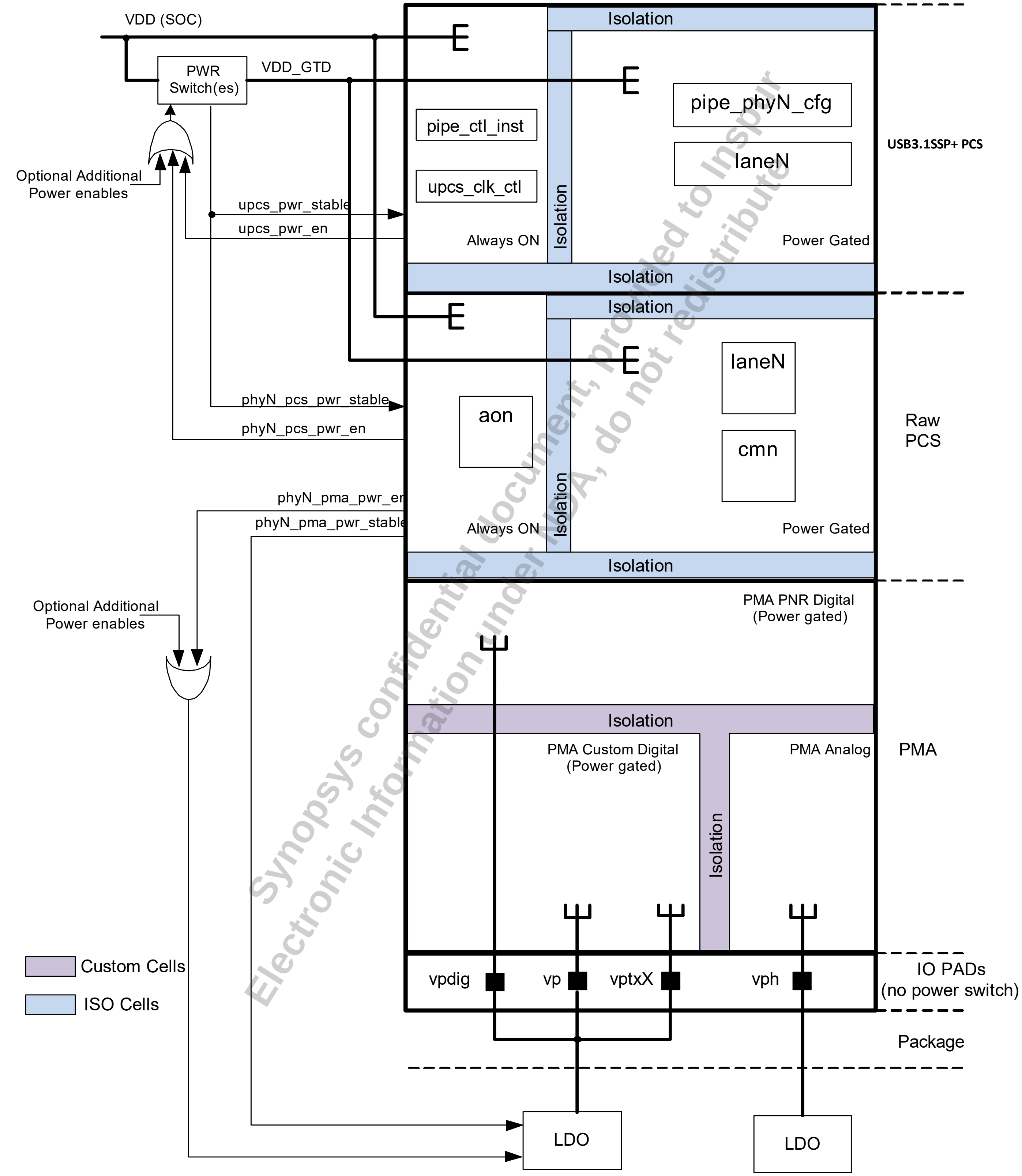
Supply	Description
phyN_VSSCORE_X	Ground for ESD protection

Table 5-9 Power State Table

Port	All_on	All_off (P1.2)
VDD	ON	ON
VSS	ON	ON
phyN_GD	ON	ON
RAW_VDD_GTD	ON	OFF
UPCS_VDD_GTD	ON	OFF
phyN_VPH	ON	OFF
phyN_VP	ON	OFF
phyN_VPDIG	ON	OFF
VPTXX	ON	OFF
VSSCORE_X	ON	ON

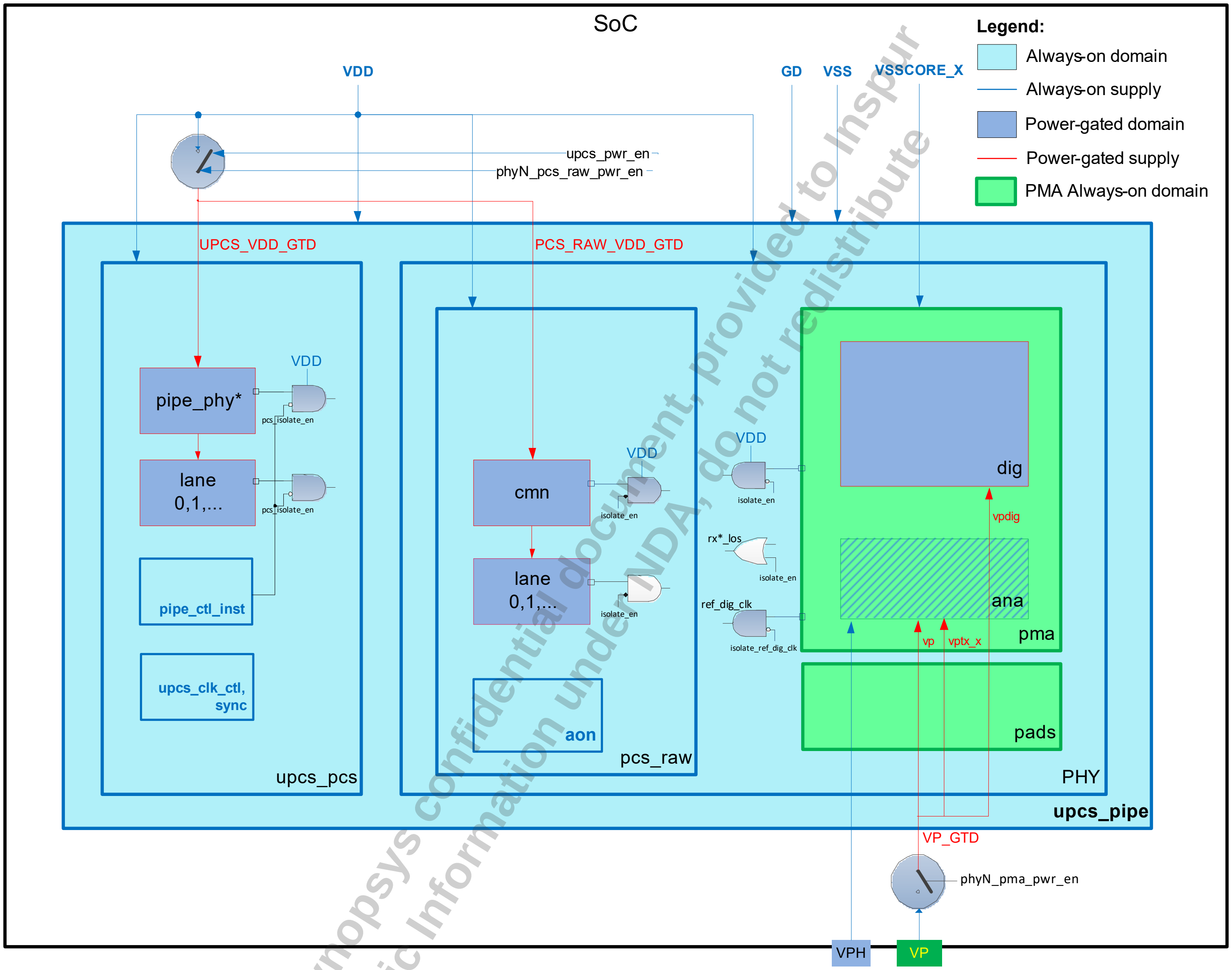
Figure 5-12 on page 262 shows the power-gating controls for the PCIe 4.0 PCS and PHY.

Figure 5-12 Power-Gating Controls for PCIe 4.0 PCS and PHY



The following figure depicts the power domain in the UPF file.

Figure 5-13 Power Domain Description in UPF File



When using PHY bifurcation, power gating has limits. For information about bifurcation, see "PHY Bifurcation Support" on page 247.

5.14 Tie-Off Values for Unused Lanes

If UPCS lanes are unused, they need to be tied-off to specific values. This allows other lanes to fully enter L1/P1 substates and allows power to be removed, where applicable.

Table 5-10 Tie-Off Values for Unused Lanes

Signal Name	Tie-Off Value
pipe_laneX_encdec_bypass	1'b0
pipe_laneX_if_width	2'd0
pipe_laneX_powerdown	4’b0100
pipe_laneX_rate	2'd0
pipe_laneX_reset_n	1'b0
pipe_laneX_m2p_messagebus	8'b0
pipe_rxX_es_mode	2'b0
pipe_rxX_blk_align_ctl	1'b0
pipe_rxX_eq_eval	1'b0
pipe_rxX_eq_in_prog	1'b0
pipe_rxX_eq_invld_req	1'b0
pipe_rxX_polarity	1'b0
pipe_rxX_sris_mode_en	1'b0
pipe_rxX_standby	1'b0
pipe_rxX_disable	1’b1
pipe_txX_disable	1’b1
pipe_txX_compliance	1'b1
pipe_txX_elecidle	1'b1
pipe_txX_data	40'h0
pipe_txX_datak	4'b0
pipe_txX_datavalid	1'b0
pipe_txX_deemph	18'd0
pipe_txX_detectrx	1'b0
pipe_txX_eq_preset	5'd0

Table 5-10 Tie-Off Values for Unused Lanes (Continued)

Signal Name	Tie-Off Value
pipe_txX_eq_preset_coeff_req	1'b0
pipe_txX_margin	3'd0
pipe_txX_startblock	1'b0
pipe_txX_swing	1'b0
pipe_txX_syncheader	4'b0

6 Simulation

The release package includes Verilog RTL for the PCIe 4.0 PCS, Raw PCS, and a Verilog behavioral model of the PMA. Contrary to the Verilog RTL for the PCIe 4.0 PCS and Raw PCS—which can be synthesized into a gate-level netlist by a synthesis tool—the PMA's Verilog behavioral model includes a netlist of generic GTECH gates and a non-synthesizable behavioral model of the analog circuits.

In the release package, the PCIe 4.0 PCS, Raw PCS Verilog RTL, and PMA Verilog behavioral model reside in the following directories:

- UPCS: ./upcs/rtl/
- Raw PCS: ./phy/rtl/
- PMA Verilog model: ./pma/behavior/

For details on the features that are not accurately modeled in the PMA behavioral model, refer to the DesignWare Cores PCIe 4.0 PHY databook.

The Verilog model can be simulated using Synopsys VCS. The following table lists the release directories where the PCIe 4.0 PCS, Raw PCS, and PMA model are stored.

Table 6-1 PCIe 4.0 PCS and PHY Model Database

Directory	Description
./phy/include	Contains macros used in the Raw PCS, PMA, and simulation files.
./upcs/include	Contains macros used in the PCS and simulation files.
./macro/rtl	Contains the top-level PIPE wrappers instantiating the PCS and the PHY. ■ dwc_* upcs_xN_ns xL_[xN][xN][xN][xN]_pipe.v: Top-level PIPE wrapper - N = number of lanes of the PMA - L = total number of lanes of the PIPE and PCS - [xN][xN][xN][xN] represents the aggregation option.
./upcs/rtl	Contains Verilog RTL files for the PCIe 4.0 PCS, which includes the following: ■ dwc_*_upcs_xN_ns_upcs_pcs_xL_[xN][xN][xN][xN].v: Top-level PCS module - N = number of lanes of the PMA - L = total number of lanes of the PCS - [xN][xN][xN][xN] represents the aggregation option.

Table 6-1 PCIe 4.0 PCS and PHY Model Database (Continued)

Directory	Description
./phy/rtl	Contains Verilog RTL for the Raw PCS files. ■ dwc_*_phy_xN_ns_pcs_raw_xN.v: Top-level RTL file for the xN Raw PCS ■ dwc_*_phy_xN_ns.v: PHY top-level wrapper instantiating PMA and Raw PCS
./pma/behavior	Contains Verilog RTL for the behavioral models for the PMA. ■ dwc_*_pma_xN_ns_gtech.v: Top-level file for the xN PMA instantiating GTECH model of the digital and behavioral model of analog sub-blocks.

For instructions on PHY-level simulations, refer to the DesignWare Cores PCIe 4.0 PHY databook.

Instructions for PIPE-level simulations are provided in the TESTBENCH.README file in the /upcs/testbench/ directory.

The TESTBENCH.README file includes the following sections:

- **Macro definitions**: These definitions are global variables that control all the test bench objects.
- **PLL configuration**: This section defines the PLL output clock.
- **Link definitions**: This section defines each link's sequence.
- **Example test_def.v**: This file is a sample test configuration file.
- **Porting the PIPE to another test bench**: This section provides a description of various simulation modes and explains how to invoke them.
- **Receiver detect**: This section provides instructions on how to force the decision of receiver detection.
- **Running simulations**: This section provides instructions on how to launch simulations from the test bench provided in the package.

For more details about each section, refer to TESTBENCH.README in the release package.

7 Synthesis

The PCIe 4.0 PCS is delivered as soft RTL, meaning that the product is hardened along with the rest of the RTL of the Raw PCS, controller, and ASIC. Synthesizing the PCIe 4.0 PCS requires using a Verilog-2001-compatible synthesis tool.

This chapter includes the following sections:

- "Synthesis Database" on page 270
- "PIPE (PCS and PHY) Constraints" on page 271

7.1 Synthesis Database

The release package includes the Design Compiler synthesis scripts in TCL format, which can be used to synthesize the PCIe 4.0 PCS and Raw PCS (PHY) logic using the timing model of the PMA (PHY). The following table describes the synthesis database structure of this PIPE-level wrapper.

Table 7-1 Synthesis Database Structure

Directory	Description
./upcs/synth	Synthesis directory for PIPE (PCIe 4.0 PCS, Raw PCS, and PMA): ■ **run_synth.sh**: Shell script to run synthesis using Design Compiler TCL scripts ■ **run_formal.sh**: Shell script to run logic equivalence checks between RTL and synthesized netlist of the PCIe 4.0 PCS and Raw PCS using Formality ■ **readme_synth.txt**: Readme containing instructions to configure and run the synthesis scripts You must update the variables in the scripts to match the installed technology library location (TECH_DIR, STD_DB). These scripts must be run in the ./upcs/synth/ directory.
./upcs/synth/scripts	Synthesis and Formality scripts: ■ **verilog_files.tcl**: List of Verilog files used for synthesis ■ **upcs_dft_insert.tcl**: Sourced from dwc_*_upcs_xN_ns_synth.tcl and used to insert DFT into the design. ■ **dwc_*_upcs_xN_ns_synth.tcl**: Design Compiler synthesis script to synthesize the PCIe 4.0 PCS and Raw PCS using timing .libs for the PMA ■ **upcs_con.tcl**: Design Compiler synthesis constraint file for the PCIe 4.0 PCS and Raw PCS in TCL format ■ **dwc_*_upcs_xN_ns_formal.tcl**: Formality script for running logic equivalence check on the synthesized netlist of the PCIe 4.0 PCS and Raw PCS

7.2 PIPE (PCS and PHY) Constraints

The synthesis constraints are provided in the form of a TCL script used by Synopsys's Design Compiler. This file (upcs_con.tcl) describes the design intent and surrounding constraints for synthesis, clocking, timing, power, test, and environmental and operating conditions. The constraints include the input and output delays of interface pins, clock definitions, and any timing path exceptions in the design, such as false paths or multicycle paths.

7.2.1 Hand-Placed Logic Constraints

The following table lists the modules in the PCIe 4.0 PCS that must be hand-instantiated as part of synthesis. These modules are typically used around asynchronous clock domain crossing (CDC) points within the design. It is essential that you instantiate these modules correctly, not just synthesize them as usual with the PCIe 4.0 PCS logic. Failure to do so may create glitch-generating and reconvergent logic that causes problems within the PCIe 4.0 PCS. For details on hand-instantiated modules used in the Raw PCS, refer to the associated DesignWare Cores PCIe 4.0 PHY databook.

To prevent these modules from being optimized during synthesis, you can either replace the contents of these modules with an instantiation of the appropriate standard cell from a digital library or synthesize each module standalone with a restricted library, so that only the correct cell is selected by the tool. In both cases, you must apply a "set_size_only" attribute to prevent re-optimization of the instance during regular PCIe 4.0 PCS synthesis.

The following table lists which cells require hand placement and/or "set_size_only" attribute.

Table 7-2 Hand-Instantiated Modules in PCIe 4.0 and SATA

Module Name	Description	Hand-Instantiated Cell Required	set_size_only Attribute Needed
pipe_gen_sync	Three-stage data signal synchronizer	Yes, see 7.2.2 for more information.	No
pipe_gen_bus_sync	Three-stage data-bus synchronizer	Yes, First three flops (d_s1,d_s2,d_s3) should be replaced	No
pipe_gen_rst_sync	Reset synchronizer	Yes, see 7.2.2 for more information.	No
pipe_SPECIAL_ani_clk_gate	Clock-gating cell	Yes	Yes
pipe_gen_and2	Two-input AND gate	Yes	Yes
pipe_gen_or2	Two-input OR gate	Yes	Yes
pipe_gen_mux	Two-input MUX	Yes	Yes
pipe_SPECIAL_clk_mux	Two-input clock mux	Yes	Yes
pipe_SPECIAL_dummy_scan_clk_mux	Two-input scan mux	Yes	Yes
pipe_SPECIAL_dummy_scan_rst_mux	Two-input scan reset mux	Yes	Yes
pipe_SPECIAL_clk_mux4	Instantiates pipe_SPECIAL_clk_mux	No	No
pipe_SPECIAL_clk_gate	Instantiates pipe_gen_mux & pipe_SPECIAL_ani_clk_gate	No	No
pipe_gen_buf	Generic random delay for simulation	No	No

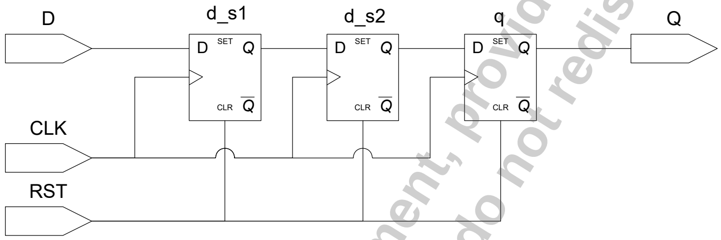
7.2.2 Synchronizers

The PCIe 4.0 PCS instantiates synchronizer modules used for clock domain crossing protection. There are two types of synchronizer modules used in the design as follows.

7.2.2.1 Data Synchronizers

For single-bit signals that must be passed from one clock domain to another, an instance of the "dwc_*pipe_gen_sync" module(s) is used to synchronize the source signal into the receiving clock domain. The following figure shows the structure of a single-bit data synchronizer.

Figure 7-1 dwc_*_pipe_gen_sync module



The input to the first stage of the synchronizer can be considered completely asynchronous, so all paths to the d_s1 register can be considered false_paths.

The second and third registers are d_s2 and q. To ensure stable synchronization, constrain the maximum delay to these cells, from d_s1/Q to d_s2/D and from d_s2/Q to q/Q, to 20% of the sampling clock frequency.

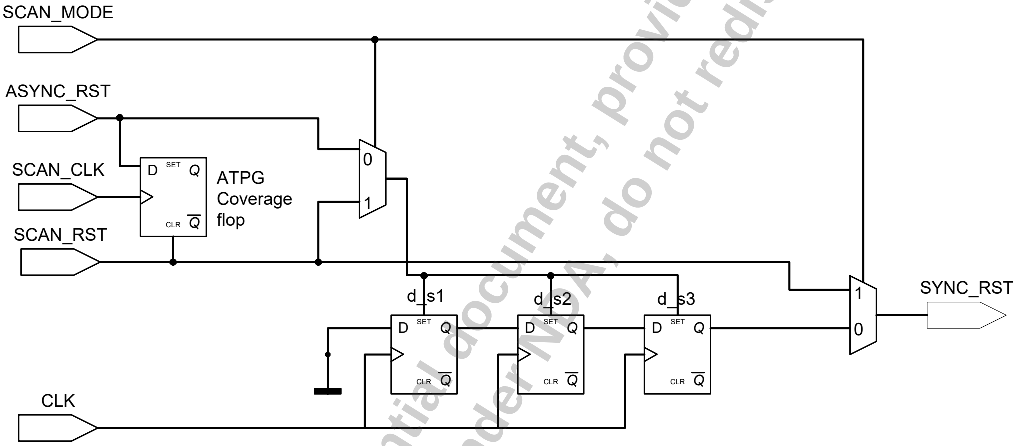
To ease timing constraint definition, this delay can be implemented as 200 ps plus the intended clock tree depth of the respective sampling clock for the associated module instances. If the paths are constrained explicitly from d_s1/Q to d_s2/D and from d_s2/Q to q/D, you can ignore the clock-tree delay.

7.2.2.2 Reset Synchronizers

For synchronizing the asynchronous reset signal, an instance of the "dwc_*_pipe_gen_rst_sync" module(s) is used. As shown in the following figure, any time an external or internal reset condition is passed from one clock domain to another domain, the reset condition must pass through a dwc_*_pipe_gen_rst_sync module to do the following:

- Asynchronously capture the reset signal.
- Stretch the reset pulse to be active for at least one full clock cycle in the receiving domain.
- Synchronously de-assert the reset signal with respect to the receiving domain.

Figure 7-2 dwc_*_pipe_gen_rst_sync module



Because assertion of a reset causes all flip-flops in the synchronizer module to capture the pulse, the de-assertion edge of the asynchronous reset input has no effect on the final flip-flop that drives the reset into the receiving domain. Therefore, all asynchronous paths through the d_s3 register of this module can be considered false.

The second and third registers in the dwc_*_gen_rst_sync module are d_s2 and d_s3. To ensure stable synchronization, constrain the maximum delay to these cells, from d_s1/Q to d_s2/D and from d_s2/Q to d_s3/D, to 20% of the sampling clock frequency.

To ease timing constraint definition, this delay can be implemented as 200 ps plus the intended clock tree depth of the respective sampling clock for the associated module instance. If the paths are constrained explicitly from d_s1/Q to d_s2/D and from d_s2/Q to d_s3/D, you can ignore the clock-tree delay.

Metastability behavior is modeled in the synchronizers and is activated by defining the ANI_SIM_MODE macro during simulations.