PHY Interface
For the
PCI Express, SATA, and USB 3.0
Architectures

Version 4.0

Intellectual Property Disclaimer

THIS SPECIFICATION IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER INCLUDING ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION, OR SAMPLE.

A COPYRIGHT LICENSE IS HEREBY GRANTED TO REPRODUCE AND DISTRIBUTE THIS SPECIFICATION FOR INTERNAL USE ONLY. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY OTHER INTELLECTUAL PROPERTY RIGHTS IS GRANTED OR INTENDED HEREBY.

INTEL CORPORATION AND THE AUTHORS OF THIS SPECIFICATION DISCLAIM ALL LIABILITY, INCLUDING LIABILITY FOR INFRINGEMENT OF PROPRIETARY RIGHTS, RELATING TO IMPLEMENTATION OF INFORMATION IN THIS DOCUMENT AND THE SPECIFICATION. INTEL CORPORATION AND THE AUTHORS OF THIS SPECIFICATION ALSO DO NOT WARRANT OR REPRESENT THAT SUCH IMPLEMENTATION(S) WILL NOT INFRINGE SUCH RIGHTS.

ALL SUGGESTIONS OR FEEDBACK RELATED TO THIS SPECIFICATION BECOME THE PROPERTY OF INTEL CORPORATION UPON SUBMISSION.

INTEL CORPORATION MAY MAKE CHANGES TO SPECIFICATIONS, PRODUCT DESCRIPTIONS, AND PLANS AT ANY TIME, WITHOUT NOTICE.

Notice: Implementations developed using the information provided in this specification may infringe the patent rights of various parties including the parties involved in the development of this specification. No license, express or implied, by estoppel or otherwise, to any intellectual property rights (including without limitation rights under any party's patents) are granted herein.

All product names are trademarks, registered trademarks, or service marks of their respective owners

Contributors

Jeff Morris
Andy Martwick
Brad Hosler
Matthew Myers
Bob Dunstan
Saleem Mohammad
Sue Vining
Tadashi Iwasaki
Yoichi Iizuka
Rahman Ismail
Bruce Tennant
Mikal Hunsaker
Jim Choate
Paul Mattos
Dan Froelich
Duane Quiet
Hajime Nozaki
Peter Teng
Karthi Vadivelu
Mineru Nishizawa
Takanori Saeki
David Vogel
Su Wei Lim
Frank Kavanagh

Dedicated to the memory of Brad Hosler, the impact of whose accomplishments made the Universal Serial Bus one of the most successful technology innovations of the Personal Computer era.

Table of Contents

Table of Figures

Formatted: Font: 10 pt, Bold

Table of Tables

1 Preface

1.1 Scope of this Revision

The PCI Express, SATA and USB SuperSpeed PHY Interface Specification has definitions of all functional blocks and signals. This revision includes support for PCI Express implementations conforming to the PCI Express Base Specification, Revision 3.0, SATA implementations conforming to the SATA specification, revision 3.0, and USB implementations conforming to the Universal Serial Bus Specification, Revision 3.0.

1.2 Revision History

Revision Number	Date	Description
0.1	7/31/02	Initial Draft
0.5	8/16/02	Draft for industry review
0.6	10/4/02	Provides operational detail
0.7	11/4/02	Includes timing diagrams
0.8	11/22/02	More operational detail. Receiver detection sequence changed.
0.9	12/16/02	Minor updates. Solid enough for implementations to be finalized.
0.95	4/25/03	Updates to reflect 1.0a Base Spec. Added multilane suggestions.
1.00	6/19/03	Stable revision for implementation.
1.70	11/6/05	First pass at Gen. 2 PIPE
1.81	12/4/2005	Fixed up areas based on feedback.
1.86	2/27/2006	Fixed up more areas based on feedback. Added a section on how to handle CLKREQ#.
1.87	9/28/2006	Removed references to Compliance Rate determination. Added sections for TX Margining and Selectable De-emphasis. Fixed up areas (6.4) based on feedback.
1.90	3/24/2007	Minor updates, mostly editorial.
2.00	7/21/2007	Minor updates, stable revision for implementation.
2.7	12/31/2007	Initial draft of updates to support the USB specification, revision 3.0.
2.71	1/21/2008	Updates for SKP handling and USB SuperSpeed PHY power management.
2.75	2/8/08	Additional updates for SKP handling.
2.90	8/11/08	Added 32 bit data interface support for USB SuperSpeed mode, support for USB SuperSpeed mode receiver equalization training, and support for USB SuperSpeed mode compliance patterns that are not 8b/10b encoded. Solid enough for implementation architectures to be finalized.
3.0	3/11/09	Final update
4.0	4/5/11	Draft 1 update adding SATA.
4.0	4/13/11	Draft 3 update adding PCI Express 3.0 rev .9.
4.0	9/1/11	Draft 6 update adding updates based on PCI Express 3.0 rev .9 feedback. Final 4.0 version.

2 Introduction

The **P**HY **I**nterface for the **P**CI **E**xpress, SATA, and USB SuperSpeed Architectures (**PIPE**) is intended to enable the development of functionally equivalent PCI Express, SATA and USB SuperSpeed PHY's. Such PHY's can be delivered as discrete IC's or as macrocells for inclusion in ASIC designs. The specification defines a set of PHY functions which must be incorporated in a PIPE compliant PHY, and it defines a standard interface between such a PHY and a Media Access Layer (MAC) & Link Layer ASIC. It is not the intent of this specification to define the internal architecture or design of a compliant PHY chip or macrocell. The PIPE specification is defined to allow various approaches to be used. Where possible the PIPE specification references the PCI Express base specification, SATA 3.0 Specification or USB 3.0 Specification rather than repeating its content. In case of conflicts, the PCI-Express Base Specification, SATA 3.0 specification and USB 3.0 Specification shall supersede the PIPE spec.

This spec provides some information about how the MAC could use the PIPE interface for various LTSSM states, Link states and other protocols. This information should be viewed as 'guidelines for' or as 'one way to implement' base specification requirements. MAC implementations are free to do things in other ways as long as they meet the corresponding specification requirements.

One of the intents of the PIPE specification is to accelerate PCI Express endpoint, SATA device and USB SuperSpeed device development. This document defines an interface to which ASIC and endpoint device vendors can develop. Peripheral and IP vendors will be able to develop and validate their designs, insulated from the high-speed and analog circuitry issues associated with the PCI Express, SATA or USB SuperSpeed PHY interfaces, thus minimizing the time and risk of their development cycles.

Figure 2-1 shows the partitioning described in this spec for the PCI Express Base Specification. Figure 2-2 shows the portioning described in this spec for the USB 3.0 Specification.

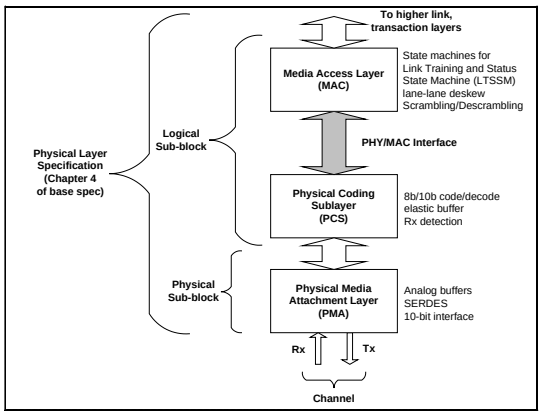


Figure 2-1: Partitioning PHY Layer for PCI Express

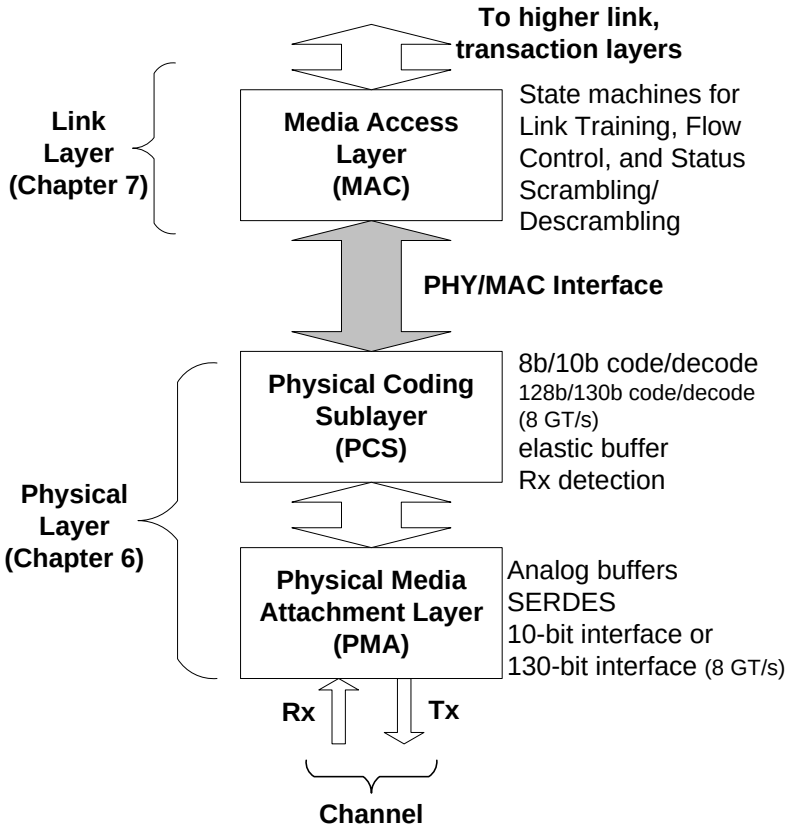


Figure 2-2 Partitioning PHY Layer for USB SuperSpeed

Formatted: Font: Bold

2.1 PCI Express PHY Layer

The PCI Express PHY Layer handles the low level PCI Express protocol and signaling. This includes features such as; data serialization and de-serialization, 8b/10b encoding/decoding, 128b/130b encoding/decoding (8 GT/s), analog buffers, elastic buffers and receiver detection. The primary focus of this block is to shift the clock domain of the data from the PCI Express rate to one that is compatible with the general logic in the ASIC.

Some key features of the PCI Express PHY are:

- Standard PHY interface enables multiple IP sources for PCI Express Logical Layer and provides a target interface for PCI Express PHY vendors.
- Supports 2.5GT/s only or 2.5GT/s and 5.0 GT/s, or 2.5 GT/s, 5.0 GT/s and 8.0 GT/s serial data transmission rate
- Utilizes 8-bit, 16-bit or 32 -bit parallel interface to transmit and receive PCI Express data
- Allows integration of high speed components into a single functional block as seen by the endpoint device designer
- Data and clock recovery from serial stream on the PCI Express bus
- Holding registers to stage transmit and receive data

- Supports direct disparity control for use in transmitting compliance pattern(s)
- 8b/10b encode/decode and error indication
- 128b/130b encode/decode and error indication
- Receiver detection
- Beacon transmission and reception
- Selectable Tx Margining, Tx De-emphasis and signal swing values

2.2 USB SuperSpeed PHY Layer

The USB SuperSpeed PHY Layer handles the low level USB SuperSpeed protocol and signaling. This includes features such as; data serialization and de-serialization, 8b/10b encoding/decoding/, analog buffers, elastic buffers and receiver detection. The primary focus of this block is to shift the clock domain of the data from the USB SuperSpeed rate to one that is compatible with the general logic in the ASIC.

Some key features of the USB SuperSpeed PHY are:

- Standard PHY interface enables multiple IP sources for USB SuperSpeed Link Layer and provides a target interface for USB SuperSpeed PHY vendors.
- Supports 5.0 GT/s serial data transmission rate
- Utilizes 8-bit, 16-bit or 32-bit parallel interface to transmit and receive USB SuperSpeed data
- Allows integration of high speed components into a single functional block as seen by the device designer
- Data and clock recovery from serial stream on the USB SuperSpeed bus
- Holding registers to stage transmit and receive data
- Supports direct disparity control for use in transmitting compliance pattern(s)
- 8b/10b encode/decode and error indication
- Receiver detection
- Low Frequency Periodic Signaling (LFPS) Transmission
- Selectable Tx Margining

2.3 SATA PHY Layer

The SATA PHY Layer handles the low level SATA protocol and signaling. This includes features such as; data serialization and deserialization, 8b/10b encoding/decoding, analog buffers, and elastic buffers. The primary focus of this block is to shift the clock domain of the data from the SATA rate to one that is compatible with the general logic in the ASIC.

Some key features of the SATA PHY are:

- Standard PHY interface enables multiple IP sources for SATA controllers and provides a target interface for SATA PHY vendors.
- Supports 1.5 GT/s only or 1.5 GT/s and 3.0 GT/s, or 1.5 GT/s, 3.0 GT/s and 6.0 GT/s serial data transmission rate
- Utilizes 8-bit, 16-bit, or 32-bit parallel interface to transmit and receive SATA data
- Allows integration of high speed components into a single functional block as seen by the device designer
- Data and clock recovery from serial stream on the SATA bus
- Holding registers to stage transmit and receive data
- 8b/10b encode/decode and error indication
- COMINIT and COMRESET transmission and reception

3 PHY/MAC Interface

Figure 3-1 shows the data and logical command/status signals between the PHY and the MAC layer. These signals are described in Section 5. Full support of PCI Express mode at all rates requires 56 control signals and 6 Status signals. Full support of USB SuperSpeed mode requires 16 control signals and 7 status signals. Full support of SATA at all rates requires 19 control signals and 6 Status signals. Refer to Section 6.1 for details on which specific signals are required for each operating mode.

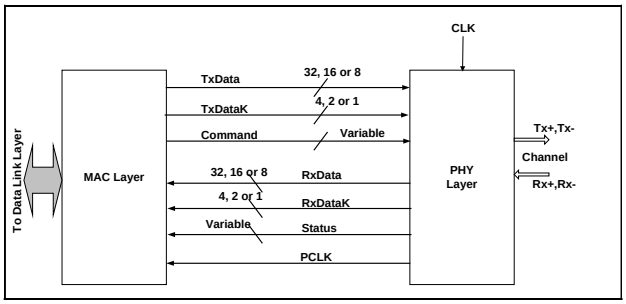


Figure 3-1: PHY/MAC Interface

This specification allows several different PHY/MAC interface configurations to support various signaling rates. For PIPE implementations that support only the 2.5 GT/s signaling rate in PCI Express mode implementers can choose to have 16 bit data paths with PCLK running at 125 MHz, or 8 bit data paths with PCLK running at 250 MHz.

PIPE implementations that support 5.0 GT/s signaling and 2.5 GT/s signaling in PCI Express mode, and therefore are able to switch between 2.5 GT/s and 5.0 GT/s signaling rates, can be implemented in several ways. An implementation may choose to have PCLK fixed at 250 MHz and use 8-bit data paths when operating at 2.5 GT/s signaling rate, and 16-bit data paths when operating at 5.0 GT/s signaling rate. Another implementation choice is to use a fixed data path width and change PCLK frequency to adjust the signaling rate. In this case, an implementation with 8-bit data paths would provide PCLK at 250 MHz for 2.5 GT/s signaling and provide PCLK at 500 MHz for 5.0 GT/s signaling. Similarly, an implementation with 16-bit data paths would provide PCLK at 125 MHz for 2.5 GT/s signaling and 250 MHz for 5.0 GT/s signaling.

For PIPE implementations that support only 5.0 GT/s (USB SuperSpeed mode) implementers can choose to have 32 bit data paths with PCLIK running at 125 MHz, or 16 bit data paths with PCLK running at 250 MHz,or 8 bit data paths with PCLK running at 500 MHz.

For SATA PIPE implementations that support only the 1.5 GT/s signaling rate implementers can choose to have 16 bit data paths with PCLK running at 75 MHz, or 8 bit data paths with PCLK running at 150, 300 or 600 MHz. The 300 and 600 Mhz options requires the use of TxDataValid and RxDataValid signals to toggle the use of data on the data bus.

SATA PIPE implementations that support 1.5 GT/s signaling and 3.0 GT/s signaling in SATA mode, and therefore are able to switch between 1.5 GT/s and 3.0 GT/s signaling rates, can be implemented in several ways. An implementation may choose to have PCLK fixed at 150 MHz and use 8-bit data paths when operating at 1.5 GT/s signaling rate, and 16-bit data paths when operating at 3.0 GT/s signaling rate. Another implementation choice is to use a fixed data path width and change PCLK frequency to adjust the signaling rate. In this case, an implementation with 8-bit data paths could provide PCLK at 150 MHz for 1.5 GT/s signaling and provide PCLK

at 300 MHz for 3.0 GT/s signaling. Similarly, an implementation with 16-bit data paths would provide PCLK at 75 MHz for 1.5 GT/s signaling and 150 MHz for 3.0 GT/s signaling. The full set of possible widths and PCLK rates for SATA mode are shown in Table 3-1. A PIPE compliant MAC or PHY is only required to support one option for each SATA transfer speed that it supports.

Mode	PCLK	Data Width
1.5 GT/s SATA	600 Mhz	8 bits* TxDataValid/RxDataValid is asserted every fourth PCLK to indicate valid data.
1.5 GT/s SATA	300 Mhz	8 bits* TxDataValid/RxDataValid is asserted every PCLK to indicate valid data.
1.5 GT/s SATA	150 Mhz	8 bits
1.5 GT/s SATA	75 Mhz	16 bits
1.5 GT/s SATA	37.5 Mhz	32 bits
3.0 GT/s SATA	300 Mhz	8 bits
3.0 GT/s SATA	150 Mhz	16 bits
3.0 GT/s SATA	75 Mhz	32 bits
3.0 GT/s SATA	600 Mhz	8 bits* TxDataValid/RxDataValid is toggled every PCLK to indicate valid data.
6.0 GT/s SATA	600 Mhz	8 bits
6.0 GT/s SATA	300 Mhz	16 bits
6.0 GT/s SATA	150 Mhz	32 bits

Table 3-1 SATA Mode - Possible PCLK rates and data widths

Note: In SATA Mode if the PHY elasticity buffer is operating in nominal empty mode – then RxDataValid may also be used when the EB is empty and no data is available.

The full set of possible widths and PCLK rates for PCI Express mode is shown in Table 3-2. A PIPE compliant MAC or PHY is only required to support one option for each PCI Express transfer speed that it supports.

Mode	PCLK	Data Width
2.5 GT/s PCI Express	1000 Mhz	8 bits* TxDataValid/RxDataValid signals are used to indicate

		valid data.
2.5 GT/s PCI Express	500 Mhz	8 bits* TxDataValid/RxDataValid signals are used to indicate valid data.
2.5 GT/s PCI Express	250 Mhz	8 bits
2.5 GT/s PCI Express	250 Mhz	16 bits* TxDataValid/RxDataValid signals are used to indicate valid data.
2.5 GT/s PCI Express	500 Mhz	16 bits* TxDataValid/RxDataValid signals are used to indicate valid data.
2.5 GT/s PCI Express	125 Mhz	16 bits
2.5 GT/s PCI Express	250 Mhz	32 bits* TxDataValid/RxDataValid signals are used to indicate valid data.
2.5 GT/s PCI Express	62.5 Mhz	32 bits
5.0 GT/s PCI Express	1000 Mhz	8 bits* TxDataValid/RxDataValid signals are used to indicate valid data.
5.0 GT/s PCI Express	500 Mhz	8 bits
5.0 GT/s PCI Express	500 Mhz	16 bits* TxDataValid/RxDataValid signals are used to indicate valid data.
5.0 GT/s PCI Express	250 Mhz	32 bits* TxDataValid/RxDataValid signals are used to indicate valid data.
5.0 GT/s PCI Express	250 Mhz	16 bits
5.0 GT/s PCI Express	125 Mhz	32 bits
8.0 GT/s PCI Express 3.0	250 Mhz	32 bits
8.0 GT/s PCI Express 3.0	500 Mhz	16 bits
8.0 GT/s PCI Express 3.0	1000 Mhz	8 bits

Table 3-2. PCI Express Mode - Possible PCLK rates and data widths

Note: When a MAC that implements the TxDataValid signal is using a mode that does not use TxDataValid the MAC shall keep TxDataValid asserted. When a PHY that implements RxDataValid is in a mode that does not use RxDataValid the PHY shall keep RxData valid asserted.

There may be PIPE implementations that support multiples of the above configurations. PHY implementations that support multiple configurations at the same rate must support the width and PCLK rate control signals. A PHY that supports multiple rates in either PCI Express Mode or SATA Mode must support configurations across all supported rates that are either fixed data path width or fixed PCLK rate.

4 PCI Express and USB PHY Functionality

Figure 4-1 shows the functional block diagram of the PHY. The functional blocks shown are not intended to define the internal architecture or design of a compliant PHY but to serve as an aid for signal grouping.

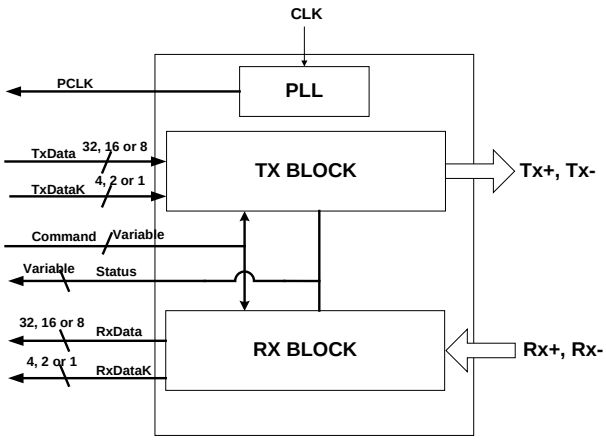


Figure 4-1: PHY Functional Block Diagram

Sections below provide descriptions of each of the blocks shown in Figure 4-1: PHY Functional Block Diagram. These blocks represent high-level functionality that is required to exist in the PHY implementation. These descriptions and diagrams describe general architecture and behavioral characteristics. Different implementations are possible and acceptable.

4.1 Transmitter Block Diagram (2.5 and 5.0 GT/s)

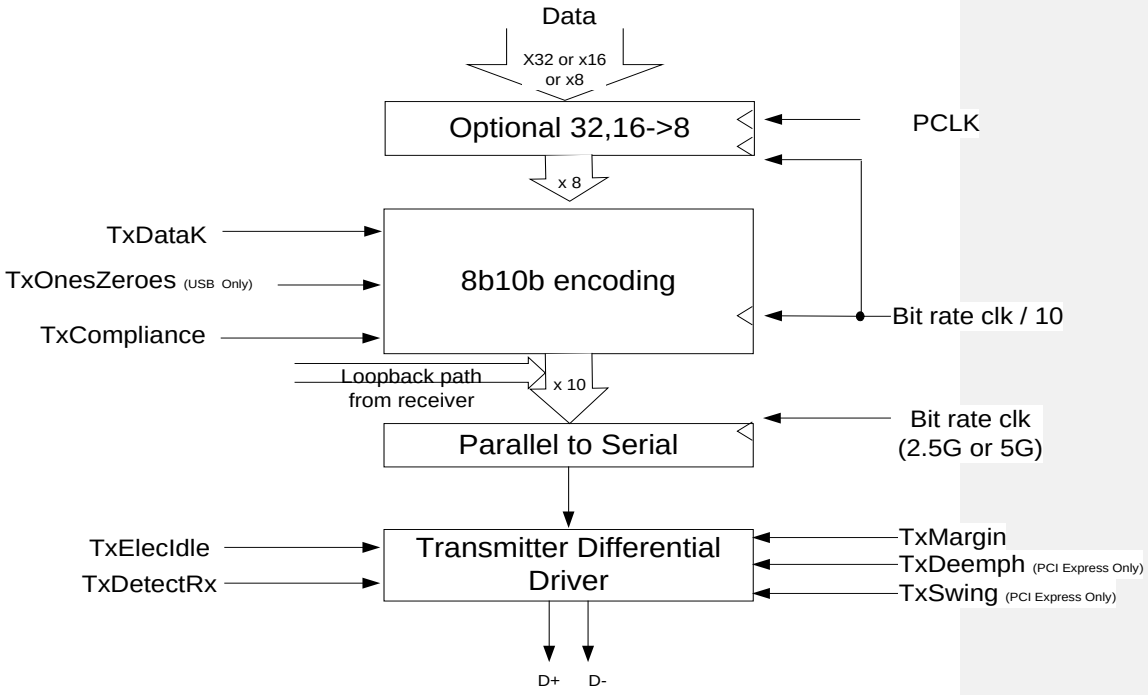


Figure 4-2: Transmitter Block Diagram

Formatted: Font: 10 pt

4.2 Transmitter Block Diagram (8.0 GT/s)

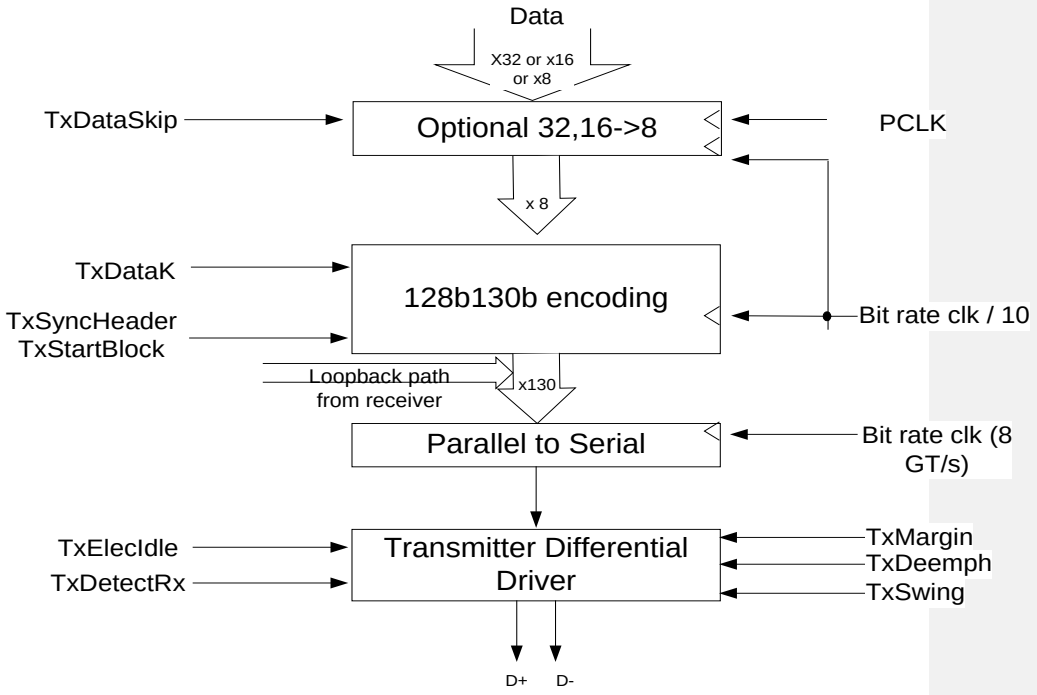


Figure 4-3: Transmitter Block Diagram (8.0 GT/s)

Formatted: Font: 10 pt

4.3 Receiver Block Diagram (2.5 and 5.0 GT/s)

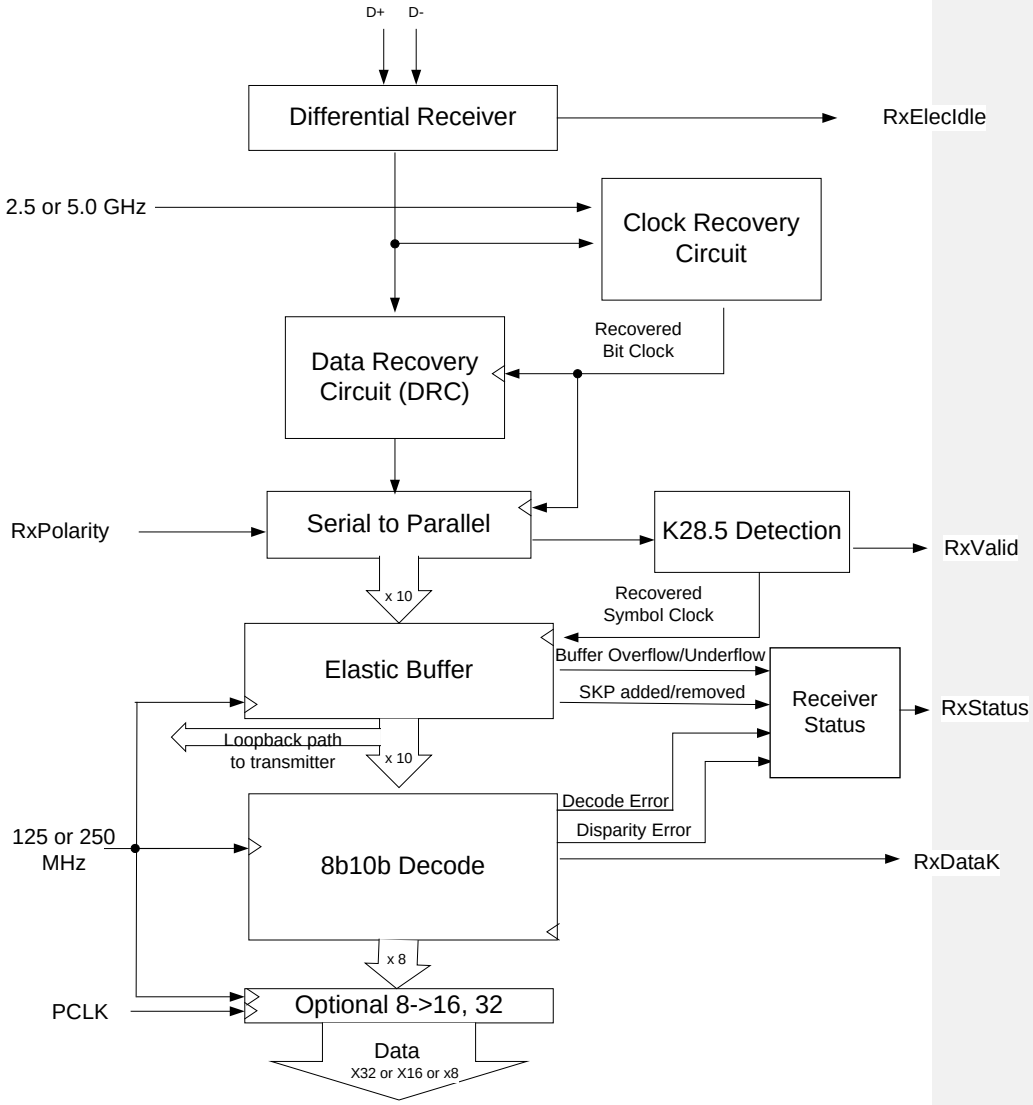


Figure 4-4: Receiver Block Diagram

Formatted: Font: 10 pt

4.4 Receiver Block Diagram (8.0 GT/s)

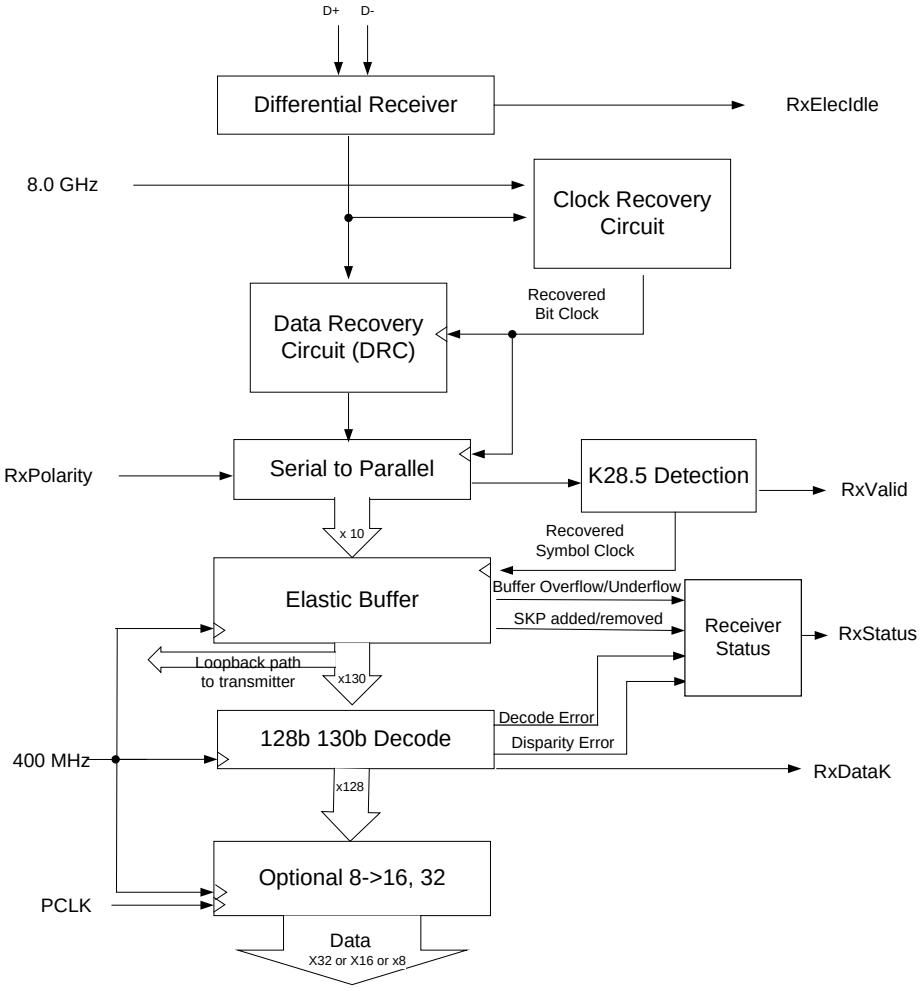


Figure 4-5: Receiver Block Diagram (8.0 GT/s)

Formatted: Font: 10 pt

4.5 Clocking

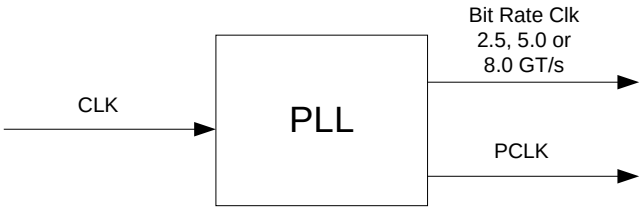


Figure 4-6: Clocking and Power Block Diagram

Formatted: Font: 10 pt

5 SATA PHY Functionality

Figure 4-1 shows the functional block diagram of a SATA PHY. The functional blocks shown are not intended to define the internal architecture or design of a compliant PHY but to serve as an aid for signal grouping.

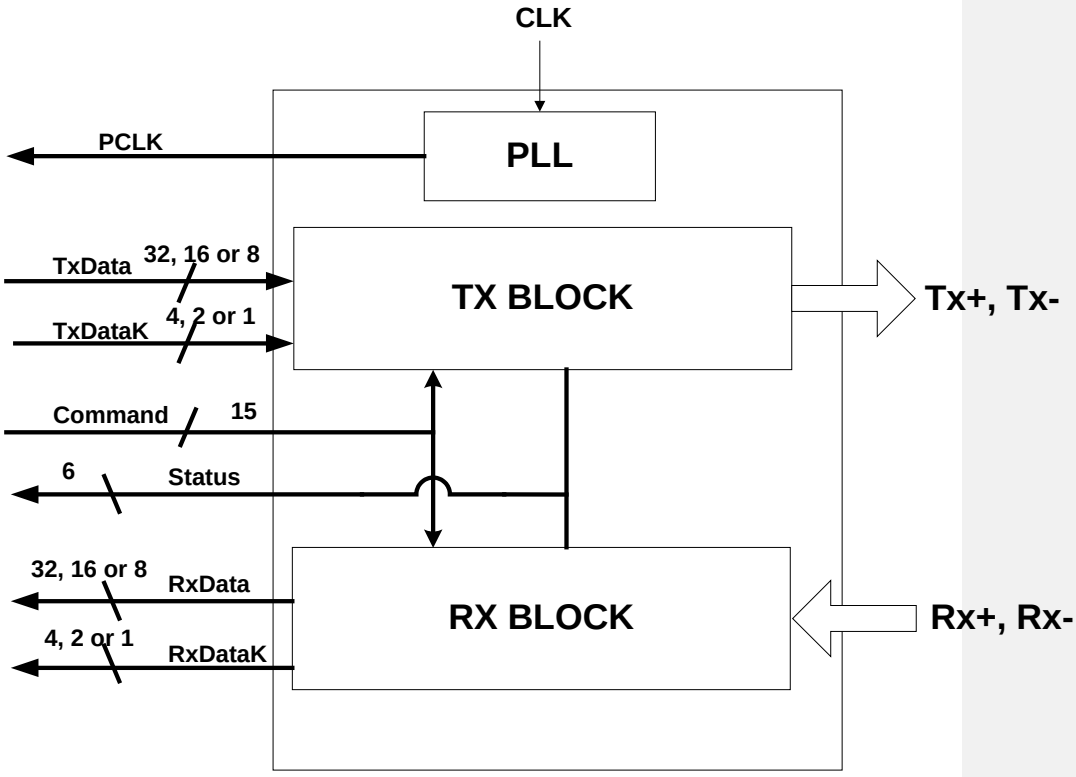


Figure 5-1: PHY Functional Block Diagram

Sections below provide descriptions of each of the blocks shown in Figure 5-1: PHY Functional Block Diagram. These blocks represent high-level functionality that is required to exist in the PHY implementation. These descriptions and diagrams describe general architecture and behavioral characteristics. Different implementations are possible and acceptable.

5.1 Transmitter Block Diagram (1.5, 3.0, and 6.0 GT/s)

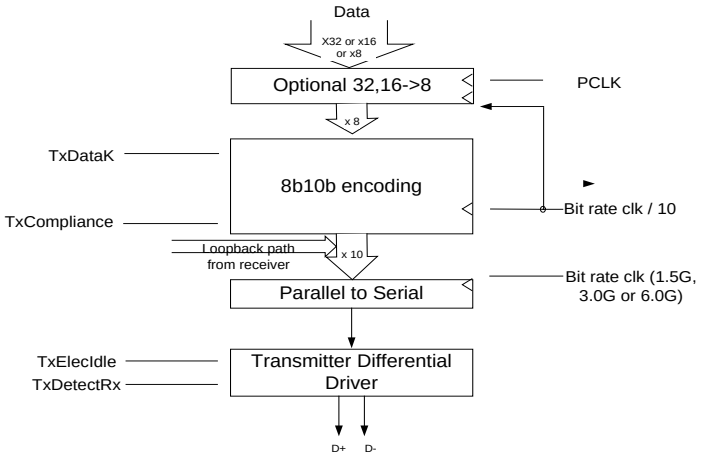


Figure 5-2: Transmitter Block Diagram (1.5, 3.0, and 6.0 GT/s)

Formatted: Font: 10 pt

5.2 Receiver Block Diagram (1.5, 3.0 and 6.0 GT/s)

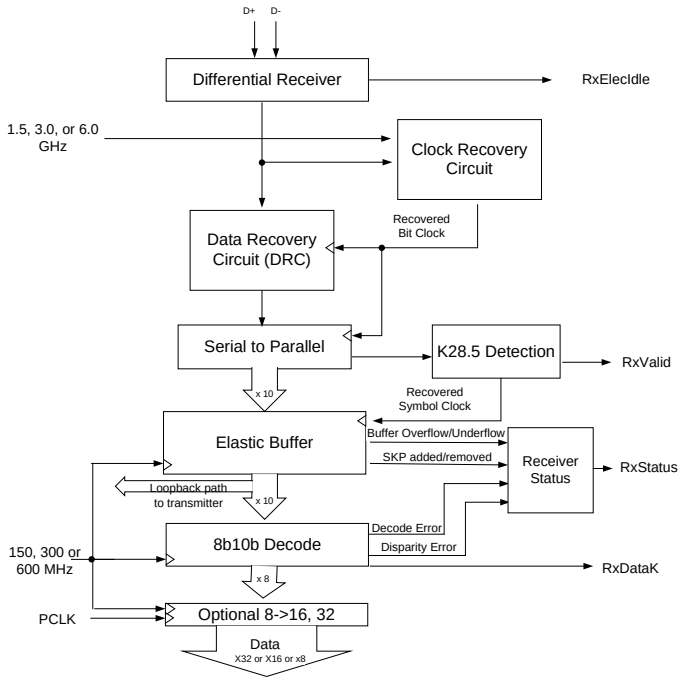


Figure 5-3: Receiver Block Diagram (1.5, 3.0 and 6.0 GT/s)

Formatted: Font: 10 pt

5.3 Clocking

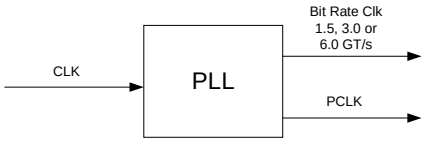


Figure 4-4: Clocking and Power Block Diagram

Formatted: Font: 10 pt, Bold

Formatted: Font: 10 pt

6 PIPE Interface Signal Descriptions

6.1 PHY/MAC Interface Signals

The PHY input and output signals are described in the following tables. Note that Input/Output is defined from the perspective of a PIPE compliant PHY component. Thus a signal described as an

"Output" is driven by the PHY and a signal described as an "Input" is received by the PHY. A basic description of each signal is provided. More details on their operation and timing can be found in following sections. All signals on the 'parallel' side of a PIPE implementation are synchronous with PCLK, with exceptions noted in the tables below.

Table 5-1: Transmit Data Interface Signals

Name	Direction	Active Level	Description
Tx+, Tx-	Output	N/A	The PCI Express, SATA or USB SuperSpeed differential outputs from the PHY. All transmitters shall be AC coupled to the media. See section 4.3.1.2 of the PCI Express Base Specification or section 3.2.1 of the USB 3.0 Specification.
TxData[31:0] for 32-bit interface TxData[15:0] for 16-bit interface TxData[7:0] for 8-bit interface	Input	N/A	Parallel PCI Express, SATA or USB SuperSpeed data input bus. For the 16-bit interface, 16 bits represent 2 symbols of transmit data. Bits [7:0] are the first symbol to be transmitted, and bits [15:8] are the second symbol. For the 32-bit interface, 32 bits represent the 4 symbols of transmit data. Bits [23:16] are the third symbol to be transmitted, and bits [31:24] are the fourth symbol. Bit zero is the first to be transmitted.
TxDataK[3:0] for 32-bit interface TxDataK[1:0] for 16-bit interface TxDataK for 8-bit interface	Input	N/A	Data/Control for the symbols of transmit data. For 32-bit interfaces, Bit 0 corresponds to the low-byte of TxData, Bit3 corresponds to the upper byte. For 16-bit interfaces, Bit 0 corresponds to the low-byte of TxData, Bit 1 to the upper byte. A value of zero indicates a data byte, a value of 1 indicates a control byte. Data bytes are scrambled and control bytes are not. Not used in PCI Express mode at 8 GT/s.
TxDataValid	Input	N/A	PCI Express Mode and SATA Mode: This signal allow the MAC to instruct the PHY to ignore the data interface for one clock cycle. A value of one indicates the phy will use the data, a value of zero indicates the phy will not use the data.
TxStartBlock	Input	N/A	PCI Express Mode: Only used at the 8.0 GT/s signaling rate. This signals allow the MAC to tell the PHY the starting byte for a 128b block. The starting byte for a 128b block must always start with Bit 0 of the data interface.

Table 5-2: Receive Data Interface Signals

Formatted: Index 1

Formatted

Formatted: Normal

Name	Direction	Active Level	Description
Rx+, Rx-	Input	N/A	The PCI Express, SATA or USB SuperSpeed differential inputs to the PHY.
RxData[31:0] for 32-bit interface RxData[15:0] for 16-bit interface or RxData[7:0] for 8-bit interface	Output	N/A	Parallel PCI Express, SATA or SuperSpeed USB data output bus. For 16-bit interface, 16 bits represents 2 symbols of receive data. Bits [7:0] are the first symbol received, and bits [15:8] are the second symbol. For the 32 bit interface, 32 bits represent the 4 symbols of receive data. Bits [23:16] are the third symbol received, and bits [31:24] are the fourth symbol received. Bit zero is the first bit received. When the PHY is in a SATA mode, the first valid data following an ALIGN primitive must appear as byte 0 in the receive data.
RxDataK[3:0] for 32-bit interface RxDataK[1:0] for 16-bit interface RxDataK for 8-bit interface	Output	N/A	Data/Control bit for the symbols of receive data. For 32-bit interfaces, Bit 0 corresponds to the low-byte of RxData, Bit3 corresponds to the upper byte. For 16-bit interface, Bit 0 corresponds to the low-byte of RxData[15:0], Bit 1 to the upper byte. A value of zero indicates a data byte; a value of 1 indicates a control byte. Not used in PCI Express mode at 8 GT/s. When the PHY is in a SATA mode, the first valid data following an ALIGN primitive must appear as byte 0 in the receive data.
RxDataValid	Output	N/A	PCI Express Mode and SATA Mode: This signal allows the PHY to instruct the MAC to ignore the data interface for one clock cycle. A value of one indicates the MAC will use the data, a value of zero indicates the MAC will not use the data. RxDataValid shall not assert when RXvalid is de-asserted.

Formatted: Index 1

Formatted

RxStartBlock	Output	N/A	PCI Express Mode: Only used at the 8.0 GT/s signaling rate. This signal allows the PHY to tell the MAC the starting byte for a 128b block. The starting byte for a 128b block must always start with Bit 0 of the data interface. Note: If there is an invalid sync header decoded on RxSyncHeader[1:0] and block alignment is still present (RxValid == 1), then the PHY will assert RxStartBlock with the invalid sync header on RxSyncHeader[1:0] RxStartBlock shall not assert when RxValid is de-asserted

Formatted

Formatted: Index 1

Formatted

Table 5-3: Command Interface Signals

<table>
<tr><th>Name</th><th>Direction</th><th>Active Level</th><th>Description</th></tr>
<tr><td>PHY Mode[1:0]</td><td>Input</td><td>N/A</td><td>Selects PHY operating mode.
<table><tr><td>Value</td><td>Description</td></tr><tr><td>0</td><td>PCI Express</td></tr><tr><td>1</td><td>USB SuperSpeed</td></tr><tr><td>2</td><td>SATA</td></tr><tr><td>3</td><td>Reserved</td></tr></table>Implementation of this signal is not required for PHYs that only support only a single mode.</td></tr>
<tr><td>Elasticity Buffer Mode</td><td>Input</td><td>N/A</td><td>Selects Elasticity Buffer operating mode.
<table><tr><td>Value</td><td>Description</td></tr><tr><td>0</td><td>Nominal Half Full Buffer mode</td></tr><tr><td>1</td><td>Nominal Empty Buffer Mode</td></tr></table>This signal is only required for a PHY that supports more than one elasticity buffer mode.

Implementation of this signal is required for PHYs that support USB SuperSpeed mode.</td></tr>
<tr><td>TxDetectRx/ Loopback</td><td>Input</td><td>High</td><td>Used to tell the PHY to begin a receiver detection operation or to begin loopback or to signal LFPS during P0 for USB Polling state. Refer to Sections 7.21 and 7.22 for details on the required values for all control signals to perform loopback and receiver detection operations and to signal Polling.LFPS.

Sata Mode:
Loopback support is optional for SATA PHYs. Loopback is only valid in Sata Mode when EncodeDecodeBypass is asserted. The RX elasticity buffer must be active during loopback. If the PHY runs out of data to transmit during loopback – it must transmit ALIGNs.

TxDetectRX is not used in SATA mode.</td></tr>
</table>

TxElecIdle	Input	High	Forces Tx output to electrical idle when asserted except in loopback. See Section 7.21 (PCI Express Mode) or Section 7.22 (USB SuperSpeed mode) or Section 7.24 (SATA Mode) for the full description and usage of this pin. ~~Sata Mode:~~ ~~Forces Tx output to electrical idle when asserted in all power states.~~ ~~When deasserted while in P0 (as indicated by the *PowerDown* signals), indicates that there is valid data present on the *TxData[..]* and *TxDataK[..]* pins and that the data must be transmitted.~~ Note: The MAC must always have TxDataValid asserted when TxElecIdle transitions to either asserted or deasserted. See section 7.3 for the definitions of PHY power states.
TX Pattern[1:0]	Input	N/A	Sata Mode: Controls which pattern the PHY sends at the Gen 1 rate when sending OOB or initialization signaling. The PHY transmits this pattern at the Gen 1 rate regardless of what rate the PHY is configured at. 0 ALIGN 1 D24.3 2 D10.2 3 Reserved See Section 7.24 for a more detailed description of the usage of these pins.
TxCompliance	Input	High	PCI Express Mode: Sets the running disparity to negative. Used when transmitting the PCI Express compliance pattern. Implementation of this signal is only required for PHYs that support PCI Express mode.

Formatted: Indent: Left: -0.25"

Formatted: Bullets and Numbering

<table>
<tr><td>TxOnesZeros</td><td>Input</td><td>High</td><td>USB SuperSpeed Mode: Used only when transmitting USB SuperSpeed compliance patterns CP7 or CP8. Causes the transmitter to transmit an alternating sequence of 50-250 ones and 50-250 zeros – regardless of the state of the TxData interface.
Implementation of this signal is only required for PHYs that support USB SuperSpeed mode.</td></tr>
<tr><td>RxPolarity</td><td>Input</td><td>High</td><td>~~USB SuperSpeed Mode and PCI Express Mode:~~
Tells PHY to do a polarity inversion on the received data.
<table>
<tr><td>Value</td><td>Description</td></tr>
<tr><td>0</td><td>PHY does no polarity inversion</td></tr>
<tr><td>1</td><td>PHY does polarity inversion</td></tr>
</table>
</td></tr>
<tr><td>RxEqTraining</td><td>Input</td><td>High</td><td>USB SuperSpeed Mode: Used to instruct the receiver to bypass normal operation to perform equalization training. While performing training the state of the RxData interface is undefined.
Implementation of this signal is only required for PHYs that support USB SuperSpeed mode.</td></tr>
<tr><td>Reset#</td><td>Input</td><td>Low</td><td>Resets the transmitter and receiver. This signal is asynchronous.

The PHY reports its default power state after reset as defined in section 7.2.</td></tr>
</table>

<table>
<tr>
<td>PowerDown[2:0]</td>
<td>Input</td>
<td>N/A</td>
<td>Power up or down the transceiver. Power states
PCI Express Mode:
<table>
<tr><th>[2]</th><th>[1]</th><th>[0]</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>P0, normal operation</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>P0s, low recovery time latency, power saving state</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>P1, longer recovery time latency, lower power state</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>P2, lowest power state</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>POWER_STATE_4 Phy specific</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>POWER_STATE_5 Phy specific</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>POWER_STATE_6 Phy specific</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>POWER_STATE_7 Phy specific</td></tr>
</table>
When transitioning from P2 to P1, the signaling is asynchronous (since PCLK is not running).

USB SuperSpeed Mode:
<table>
<tr><th>[2]</th><th>[1]</th><th>[0]</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>P0, normal operation</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>P1, low recovery time latency, power saving state</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>P2, longer recovery time latency, lower power state</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>P3, lowest power state</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>POWER_STATE_4 Phy specific</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>POWER_STATE_5 Phy specific</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>POWER_STATE_6 Phy specific</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>POWER_STATE_7 Phy specific</td></tr>
</table>
When transitioning from P3 to P0, the signaling is asynchronous (since PCLK is not running).</td>
</tr>
</table>

<table>
<tr>
<td>PowerDown[2:0]
Sata Mode</td>
<td>Input</td>
<td>N/A</td>
<td>
Sata Mode:

Power up or down the transceiver. Power states
[2] [1] [0] Description
0 0 0 POWER_STATE_0 Operational state.
0 0 1 POWER_STATE_1 Phy specific.
0 1 0 POWER_STATE_2 Phy specific.
0 1 1 POWER_STATE_3 Phy specific.
1 0 0 POWER_STATE_4 Phy specific.
1 0 1 POWER_STATE_5 Phy specific.
1 1 0 POWER_STATE_6 Phy specific.
1 1 1 POWER_STATE_7 Phy specific.

A PIPE compliant SATA PHY is recommended to support at least 4 states other than POWER_STATE_0. There must be at least one ~~additional~~ power state meeting the requirements shown in the following table

<table>
<tr><th>PCLK State</th><th>TX Common Mode State</th><th>Exit Latency to POWER_STATE_0</th></tr>
<tr><td>Off</td><td>Off</td><td>< 10 ms</td></tr>
<tr><td>Off</td><td>On</td><td>< 10 us</td></tr>
<tr><td>On</td><td>On</td><td>< 10 us</td></tr>
<tr><td>On</td><td>Off</td><td>< 300 us</td></tr>
</table>

Exit latency to POWER_STATE_0 is measured from when the MAC changes the Power down value to when the PHY deasserts PHY status. The actual PHY latency must provide enough margin from the indicated limits to enable compliant device behavior per the SATA specification. A MAC must map the available PHY states to SATA states.

Note: PLL shutdown is only possible if PowerDown is set to a state with PCLK off.

~~An informative table with value of all signals after reset will be added in the next revision.~~
</td>
</tr>
</table>

<table>
<tr>
<td>Rate[1:0]</td>
<td>Input</td>
<td>N/A</td>
<td>Control the link signaling rate.

PCI Express Mode:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0</td><td>Use 2.5 GT/s signaling rate</td></tr>
<tr><td>1</td><td>Use 5.0 GT/s signaling rate</td></tr>
<tr><td>2</td><td>Use 8.0 GT/s signaling rate</td></tr>
<tr><td>3</td><td>Reserved</td></tr>
</table>
Sata Mode:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0</td><td>Use 1.5 GT/s signaling rate</td></tr>
<tr><td>1</td><td>Use 3.0 GT/s signaling rate</td></tr>
<tr><td>2</td><td>Use 6.0 GT/s signaling rate</td></tr>
<tr><td>3</td><td>Reserved</td></tr>
</table>
PIPE implementations that only support one signaling rate do not implement this signal.</td>
</tr>
<tr>
<td>Width[1:0]</td>
<td>Input</td>
<td>N/A</td>
<td>Control the PIPE data path width
0 8 bits
1 16 bits
2 32 bits
3 Reserved

PIPE implementations that only support one option at each signaling rate do not implement this signal.

This field is optional for PHYs that only support USB Mode.</td>
</tr>
</table>

PCLK Rate[2:0]	Input	N/A	Control the PIPE PCLK rate SATA Mode: 0 37.5 Mhz 1 75 Mhz 2 150 Mhz 3 300 Mhz 4 600 Mhz 5 Reserved 6 Reserved 7 Reserved PCI Express Mode: 0 62.5 Mhz 1 125 Mhz 2 250 Mhz 3 500 Mhz 4 1000 Mhz 5 Reserved 6 Reserved 7 Reserved PIPE implementations that do not support more than one PCLK rate for any analog signaling rate do not implement this signal.
LocalTxPresetCoefficients[17:0]	Output	N/A	PCI Express Mode: These are the coefficients for the preset on the *LocalPresetIndex*[3:0] after a *GetLocalPresetCoeffcients* request: [5:0] C-1 [11:6] C0 [17:12] C+1 Valid on assertion of *LocalTxCoefficientsValid*. The MAC will reflect these coefficient values on the TxDeemph bus when MAC wishes to apply this preset. These signals are only used by a PHY that requires dynamic preset coefficient updates.

Formatted: Font: (Default) Arial, 11 pt

Formatted: Font: (Default) Arial

Formatted: Normal

<table>
<tr>
<td>TxDeemph[17:0]</td>
<td>Input</td>
<td>N/A</td>
<td>Selects transmitter de-emphasis.
When rate is 2.5 or 5.0 GT/s

<table>
<tr><td>Value</td><td>Description</td></tr>
<tr><td>0</td><td>-6dB de-emphasis</td></tr>
<tr><td>1</td><td>-3.5dB de-emphasis</td></tr>
<tr><td>2</td><td>No de-emphasis</td></tr>
<tr><td>3</td><td>Reserved</td></tr>
</table>

PIPE implementations that only support 2.5 GT/s do not implement this signal. PIPE PHY implementations that do not support low swing are not required to support the no-de-emphasis mode.

When the rate is 8.0 GT/s
[5:0] C_{-1}
[11:6] C_0
[17:12] C_{+1}
Note: The MAC must ensure that only supported values are used for TxDeemph.

This signal is not defined for Sata Mode.</td>
</tr>
<tr>
<td>RxPresetHint[2:0]</td>
<td>Input</td>
<td>N/A</td>
<td>PCI Express Mode:
Provides the RX preset hint for the receiver. These signals are only used at the 8.0 GT/s signaling rate.

Note: It is recommended that this value not be used by a PHY.</td>
</tr>
<tr>
<td>LocalFS[5:0]</td>
<td>Output</td>
<td>N/A</td>
<td>PCI Express Mode:
Provides the FS value for the PHY. These signals are only used by a PHY that requires dynamic preset coefficient updates.

This value shall be sampled by the MAC only when PhyStatus is pulsed after RESET# or on the first PhyStatus pulse after a rate change to 8 GT/s.</td>
</tr>
</table>

LocalLF[5:0] ~~FS[5:0]~~	Output~~In put~~	N/A~~N/ A~~	PCI Express Mode: Provides the LF value for the PHY. This signal is only used by a PHY that requires dynamic preset coefficient updates. This value must only be sampled by the MAC only when PhyStatus is pulsed after RESET# or on the first PhyStatus pulse after a rate change to 8 GT/s. ~~PCI Express Mode: Provides the FS value advertised by the link partner. A PHY may optionally consider this value when deciding how long to evaluate TX equalization settings of the link partner. These signals are only used at the 8.0 GT/s signaling rate.~~
LocalPresetIndex[3:0]	Input	N/A	PCI Express Mode: Index for local PHY preset coefficients requested by the MAC The preset index value is encoded as follows: 0000b – Preset P0. 0001b – Preset P1. 0010b – Preset P2. 0011b – Preset P3. 0100b – Preset P4. 0101b – Preset P5. 0110b – Preset P6. 0111b – Preset P7. 1000b – Preset P8. 1001b – Preset P9. 1010b – Preset P10. 1011b – Reserved 1100b – Reserved 1101b – Reserved 1110b – Reserved 1111b – Reserved. These signals are only used with a PHY that requires dynamic preset coefficient updates.

Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial
Formatted: Font: (Default) Arial
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt
Formatted: Normal
Formatted: Font: (Default) Arial
Formatted: Font: (Default) Arial
Formatted: Font: (Default) Arial
Formatted: Font: (Default) Times New Roman, 11 pt
Formatted: Normal
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial
Formatted: Normal

GetLocalPresetCoeffcients	Input	High	PCI Express Mode: A MAC holds this signal high for one PCLK cycle requesting a preset to co-efficient mapping for the preset on LocalPresetIndex[3:0] to coefficients on LocalTxPresetCoefficient[17:0] Maximum Response time of PHY is 128 nSec. Note. A MAC can make this request any time after reset. This signal is only used with a PHY that requires dynamic preset coefficient updates
LocalTxCoefficientsValid	Output	High	PCI Express Mode: A PHY holds this signal high for one PCLK cycle to indicate that the LocalTxPresetCoefficients[17:0] bus correctly represents the coefficients values for the preset on the LocalPresetIndex bus. This signal is only used by a PHY that requires dynamic preset coefficient updates
LF[5:0]	Input	N/A	PCI Express Mode: Provides the LF value advertised by the link partner. A PHY may optionally consider this value when deciding how long to evaluate TX equalization settings of the link partner. These signals are only used at the 8.0 GT/s signaling rate.
RxEqEval	Input	High	PCI Express Mode: The PHY starts evaluation of the far end transmitter TX EQ settings while this signal is held high by the MAC. This signal is only used at the 8.0 GT/s signaling rate.

Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt, Not Italic
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt, Not Italic
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt
Formatted: Normal
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt
Formatted: Font: (Default) Arial, 11 pt, Not Italic
Formatted: Normal

LinkEvaluationFeedbackFigureMerit[7:0]	Output	N/A	PCI Express Mode: Provides the PHY link equalization evaluation Figure of Merit value. The value is encoded as an integer from 0 to 255. A PHY does not implement these signals if it is does not provide link equalization evaluation feedback using the Figure of Merit format. These signals are only used at the 8.0 GT/s signaling rate.
LinkEvaluationFeedbackDirectionChange[5:0]	Output	N/A	PCI Express Mode: Provides the link equalization evaluation feedback in the direction change format. Feedback is provided for each coefficient: [1:0] C_{-1} [3:2] C_0 [5:4] C_1 The feedback value for each coefficient is encoded as follows: 00 - No change 01 – Increment 10 – Decrement 11 - Reserved A PHY does not implement these signals if it is does not provide link equalization evaluation feedback using the Direction Change format. Note: In 8.0 GT/s mode the MAC shall ignore the C_0 value and use the correct value per the PCI Express specification. These signals are only used at the 8.0 GT/s signaling rate.

InvalidRequest	Input	High	PCI Express Mode: Indicates that the Link Evaluation feedback requested a link partner TX EQ setting that was out of range. The MAC asserts this signal when it detects an out of range error until the next time it asserts RxEQEval. When a MAC asserts this signal it shall subsequently ask the PHY to perform an RxEQ evaluation using the last valid setting a second time. This signal is only used at the 8.0 GT/s signaling rate.

<table>
<tr>
<td>TxMargin[2:0]</td>
<td>Input</td>
<td>N/A</td>
<td>Selects transmitter voltage levels.
<table>
<tr><th>[2]</th><th>[1]</th><th>[0]</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>TxMargin value 0 = Normal operating range</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>TxMargin value 1 = 800-1200mV for Full swing* OR
400-700mV for Half swing*</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>TxMargin value 2 = required and vendor defined</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>TxMargin value 3 = required and vendor defined</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>TxMargin value 4 = required and 200-400mV for Full swing* OR 100-200mV for Half swing* if the last value or vendor defined</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>TxMargin value 5 = optional and 200-400mV for Full swing* OR 100-200mV for Half swing* if the last value OR vendor defined OR Reserved if no other values supported</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>TxMargin value 6 = optional and 200-400mV for Full swing* OR 100-200mV for Half swing* if the last value OR vendor defined OR Reserved if no other values supported</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>TxMargin value 7 = optional and 200-400mV for Full swing* OR 100-200mV for Half swing* if the last value OR Reserved if no other values supported</td></tr>
</table>
PIPE implementations that only support PCI Express mode and the 2.5GT/s signaling rate do not implement this signal.

This signal is not defined for SATA mode.</td>
</tr>
<tr>
<td>TxSwing</td>
<td>Input</td>
<td>N/A</td>
<td>PCI Express Mode:
Controls transmitter voltage swing level
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0</td><td>Full swing</td></tr>
<tr><td>1</td><td>Low swing (optional)</td></tr>
</table>
Implementation of this signal is optional if only Full swing is supported.
This signal is not used at the 8.0 GT/s signaling rate.</td>
</tr>
</table>

<table>
<tr><td>TxSyncHeader[1:0]</td><td>Input</td><td>N/A</td><td>PCI Express Mode:
Provides the sync header for the PHY to use in the next 130b block. The PHY reads this value when the TXStartBlock signal is asserted.
This signal is only used at the 8.0 GT/s signaling rate.</td></tr>
<tr><td>RxSyncHeader[1:0]</td><td>Output</td><td>N/A</td><td>PCI Express Mode:
Provides the sync header for the MAC to use with the next 128b block. The MAC reads this value when the RxStartBlock signal is asserted.
This signal is only used at the 8.0 GT/s signaling rate.

Note: The PHY shall pass blocks normally across the PIPE interface even if the decoded SyncHeader is invalid.</td></tr>
<tr><td>BlockAlignControl</td><td>Input</td><td>N/A</td><td>PCI Express Mode:
Controls whether the PHY performs block alignment.
This signal is only used at the 8.0 GT/s signaling rate.</td></tr>
<tr><td>RX Termination</td><td>Input</td><td>High</td><td>Controls presence of receiver terminations:
<table><tr><th>Value</th><th>Description</th></tr><tr><td>0</td><td>Terminations removed</td></tr><tr><td>1</td><td>Terminations present</td></tr></table>
Implementation of this signal is only required for PHYs that support USB SuperSpeed mode.</td></tr>
</table>

RxStandby	Input	N/A	SATA Mode: Controls whether the PHY RX is active when the PHY is in any power state with PCLK on.. 0 – Active 1 – Standby RxStandby is ignored when the PHY is in any power state where the high speed receiver is always off. PCI Express Mode: Controls whether the PHY RX is active when the PHY is in P0 or P0s. 0 – Active 1 – Standby RxStandby is ignored when the PHY is in P1 or P2.
RxStandbyStatus	Output	N/A	SATA Mode and PCI Express Mode: The PHY uses this signal to indicate its RxStandby state. 0 – Active 1 – Standby RxStandbyStatus reflects the state of the high speed receiver. The high speed receiver is always off in PHY states that do not provide PCLK. The PHY indicates in section 6.8 any other power states in which the high speed receiver is always off. PCI Express Mode: RxStandbyStatus is undefined when the power state is P1 or P2.

EncodeDecodeBypass	Input	N/A	PCI Express Mode and SATA Mode: Controls whether the PHY performs 8b/10b encode and decode. 0 – 8b/10b encode/decode performed normally by the PHY. 1 – 8b/10b encode/decode bypassed. The MAC can only change this signal during reset or in a power state other than POWER_STATE_0 (SATA Mode) or P0 (PCI Express Mode). SATA Mode: When EncodeDecodeBypass is one the TxDataK and RxDataK interfaces are not used and the data bus width is 10, 20, or 40 bits. PCI Express Mode: When EncodeDecodeBypass is one the TxDataK and RxDataK interfaces are not used. The data bus width is 10, 20, or 40 bits if rate is 2.5 or 5.0 GT/s. The data bus width is 8, 16, or 32 bits if the rate is 8.0 GT/s. The TxStartBlock and RxStartBlock signals are not used. Support of this signal in PCI Express Mode is optional for a PHY.

Table 5-4: Status Interface Signals

Name	Direction	Active Level	Description
RxValid	Output	High	Indicates symbol lock and valid data on *RxData* and *RxDataK*.
PhyStatus	Output	High	Used to communicate completion of several PHY functions including stable PCLK after Reset# deassertion, power management state transitions, rate change, and receiver detection. When this signal transitions during entry and exit from any PHY state where PCLK is not provided, then the signaling is asynchronous. In error situations (where the PHY fails to assert PhyStatus) the MAC can take MAC-specific error recovery actions.

AlignDetect	Output	High	Sata Mode: Indicates receiver detection of an Align. A PHY is only required to assert this signal when the Elasticity Buffer is running in nominal empty mode. The PHY shall only toggle this signal after obtaining bit and symbol lock. Each ALIGN received shall map to AlignDetect being asserted for one PCLK. The spacing between PCLK pulses for ALIGNs should map analog spacing of received ALIGNs as closely as possible. Hower there is no guarantee to have PCLK domain spacing between back to back AlignDetect pulses match the analog spacing exactly due to differences in the receive clock domain and the PCLK domain. For example: 1.5 GT/s with 8-bit data path PCLK=150MHz, the nominal spacing is 4 PCLK's. 3.0 GT/s with 8-bit data path PCLK=300MHz, the nominal spacing is 4 PCLK's. 6.0 GT/s with 16-bit data path PCLK=300MHz, the nominal spacing is every other PCLK. Due to differences in the PCLK and receive clocks, the nominal spacing can be off by one PCLK in either direction. In the example with PCLK rate being equal to Gen3 received clock rate, clock domain crossing could lead to AlignDetect being asserted for consecutive PCLK cycles without gap.

RxElecIdle	Output	High	Indicates receiver detection of an electrical idle. While deasserted with the PHY in P2 (PCI Express mode) or the PHY in P0, P1, P2, or P3 (USB SuperSpeed Mode), indicates detection of either: PCI Express Mode: a beacon. USB SuperSpeed Mode : LFPS This is an asynchronous signal. PCI Express Mode: It is required at the 5.0 GT/s and 8.0 GT/s rates that a MAC uses logic to detect electrical idle entry instead of relying on the RxElecIdle signal. Sata Mode: The time the signal is asserted must match the actual idle time on the analog bus within -16/+0 ns.

<table>
<tr><td>RxStatus[2:0]</td><td>Output</td><td>N/A</td><td>Encodes receiver status and error codes for the received data stream when receiving data.
<table>
<tr><th>[2]</th><th>[1]</th><th>[0]</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>Received data OK</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>PCI Express Mode: 1 SKP added
USB SuperSpeed Mode: 1 SKP Ordered Set added
Sata Mode: 1 ALIGN added
Asserted with first byte of Align that was added. An align may only be added in conjunction with receiving one or more aligns in the data stream and only when the elasticity buffer is operating in half full mode</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>PCI Express Mode: 1 SKP removed
USB SuperSpeed Mode: 1 SKP Ordered Set removed
SATA Mode: 1 or more ALIGNs removed
This status is asserted with first non ALIGN byte following an ALIGN. This status message is applicable to both EB buffer modes.</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>PCI Express and USB SuperSpeed Modes:
Receiver detected
SATA Mode: Misalign
Signaled on the first symbol of an ALIGN that was received misaligned in elasticity buffer nominal half full mode. Signaled on the first data following an align in elasticity buffer nominal empty mode.</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>Both 8B/10B (128B/130B[1]) decode error and (optionally) Receive Disparity error Note: This error is never reported if EncodeDecodeBypass is asserted.</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>Elastic Buffer overflow</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>Elastic Buffer underflow.
This error code is not used if the elasticity buffer is operating in the nominal buffer empty mode.</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>Receive disparity error (Reserved if Receive Disparity error is reported with code 0b100)
Not used if EncodeDecodeBypass is asserted.</td></tr>
</table></td></tr>
</table>

[1] Disparity errors are not reported when the rate is 8.0 GT/s.

PowerPresent	Output	High	USB SuperSpeed Mode: Indicates the presence of VBUS. Implementation of this signal is only required for PHYs that support USB SuperSpeed mode.

Errors in SKP ordered sets shall be reported as 128/130 decode errors. An error in a SKP ordered set shall be reported if there is an error in the first 4N+1 symbols of the skip ordered set.

6.2 External Signals

Table 5-5: External Signals

Name	Direction	Active Level	Description
CLK	Input	Edge	This differential Input is used to generate the bit-rate clock for the PHY transmitter and receiver. Specs for this clock signal (frequency, jitter, …) are implementation dependent and must be specified for each implementation. This clock may have a spread spectrum modulation.
PCLK	Output	Rising Edge	Parallel interface differential data clock. All data movement across the parallel interface is synchronized to this clock. This clock operates at 37.5MHz , 75 MHz, 125MHz, 150 MHz, 250 MHz, 300 MHz, 500 MHz , 600 MHz, or 1000 MHz depending on the *Rate* and *PHY Mode* control inputs and the data interface width. The rising edge of the clock is the reference for all signals. Spread spectrum modulation on this clock is allowed.
Max PCLK	Output	Rising Edge	Parallel interface data clock. This fixed rate clock operates at the following rate: PCI Express Mode: Max rate supported — Max PCLK 2.5 GT/s — 250 MHz. 5.0 GT/s — 500 MHz. 8.0 GT/s — 1000 MHz. This clock is provided whenever PCLK is active. SATA Mode: Max rate supported — Max PCLK 1.5 GT/s — 150 MHz. 3.0 GT/s — 300 MHz. 6.0 GT/s — 600 MHz. This clock is provided whenever PCLK is active. Spread spectrum modulation on this clock is allowed. This signal is optional for SATA~~PCI Express~~ and USB SuperSpeed mode.

<table>
<tr>
<td>DataBusWidth[1:0]</td>
<td>Output</td>
<td>N/A</td>
<td>This field reports the width of the data bus that the PHY is configured for.

This field is optional.

<table>
<tr><th>[1]</th><th>[0]</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>32-bit mode</td></tr>
<tr><td>0</td><td>1</td><td>16-bit mode</td></tr>
<tr><td>1</td><td>0</td><td>8-bit mode</td></tr>
<tr><td>1</td><td>1</td><td>Reserved</td></tr>
</table>
</td>
</tr>
</table>

7 PIPE Operational Behavior

7.1 Clocking

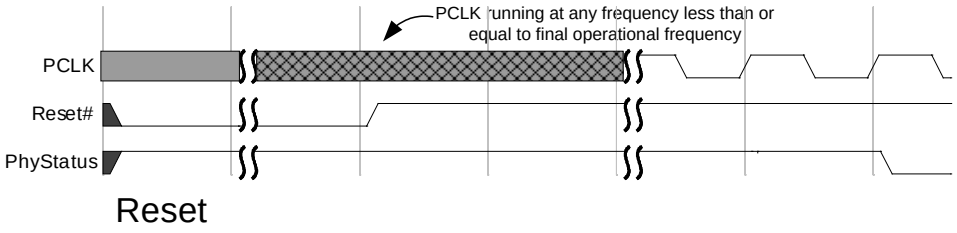
There are three clock signals used by the PHY Interface component. The first (*CLK*) is a reference clock that the PHY uses to generate internal bit rate clocks for transmitting and receiving data. The specifications for this signal are implementation dependent and must be fully specified by vendors. The specifications may vary for different operating modes of the PHY. This clock may have spread spectrum modulation that matches a system reference clock (for example, the spread spectrum modulation could come from REFCLK from the Card Electro-Mechanical Specification).

The second clock (*PCLK*) is an output from the PHY and is the parallel interface clock used to synchronize data transfers across the parallel interface. This clock runs at a rate dependent on the *Rate*, *PCLK Rate*, and *PHY Mode* control inputs and data interface width. The rising edge of this clock is the reference point. This clock may also have spread spectrum modulation.

The third clock (*MAX PCLK*) is a constant frequency clock with a frequency determined by the maximum SATA signaling rate supported by the PHY and is only required in SATA Mode.

7.2 Reset

When the MAC wants to reset the PHY (e.g.; initial power on), the MAC must hold the PHY in reset until power and *CLK* to the PHY are stable. The PHY signals that *PCLK* is valid (i.e. PCLK has been running at its operational frequency for at least one clock) and the PHY is in the specified power state by the deassertion of *PhyStatus* after the MAC has stopped holding the PHY in reset. While *Reset#* is asserted the MAC should have *TxDetectRx/Loopback* deasserted, *TxElecIdle* asserted, *TxCompliance* deasserted, *RxPolarity* deasserted, *PowerDown* = P1 (PCI Express mode) or *PowerDown* = P2 (USB SuperSpeed Mode) or *PowerDown* set to the default value reported by the PHY (SATA Mode), *TxMargin* = 000b, *TxDeemp* = 1, *PHY Mode* set to the desired PHY operating mode, and *Rate* set to 2.5GT/s signaling rate for a PHY in PCI Express mode or 5.0 GT/s for a PHY in USB SuperSpeed 3.0 mode or any rate supported by the PHY in SATA mode. The state of *TxSwing* during *Reset#* assertion is implementation specific. *RxTermination* is asserted in USB SuperSpeed 3.0 mode.



Reset

7.3 Power Management – PCI Express Mode

The power management signals allow the PHY to minimize power consumption. The PHY must meet all timing constraints provided in the PCI Express Base Specification regarding clock recovery and link training for the various power states. The PHY must also meet all terminations requirements for transmitters and receivers.

Four standard power states are defined, P0, P0s, P1, and P2. P0 state is the normal operational state for the PHY. When directed from P0 to a lower power state, the PHY can immediately take whatever power saving measures are appropriate. A PHY is allowed to implement up to 4 additional PHY specific power states. A MAC may use any of the PHY specific states as long as the PCI Express base specification requirements are still met.

In states P0, P0s and P1, the PHY is required to keep *PCLK* operational. For all state transitions between these three states and any PHY specific states where *PCLK* is operational, the PHY indicates successful transition into the designated power state by a single cycle assertion of *PhyStatus*. Transitions into and out of P2 or a PHY specific state where *PCLK* is not operational are described below. For all power state transitions, the MAC must not begin any operational sequences or further power state transitions until the PHY has indicated that the initial state transition is completed.

Mapping of PHY power states to states in the Link Training and Status State Machine (LTSSM) found in the base specification are included below. A MAC may alternately use PHY specific states as long as the base specification requirements are still met.

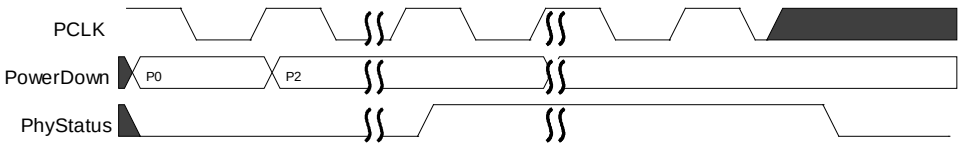
- P0 state: All internal clocks in the PHY are operational. P0 is the only state where the PHY transmits and receives PCI Express signaling.
  P0 is the appropriate PHY power management state for most states in the Link Training and Status State Machine (LTSSM). Exceptions are listed below for each lower power PHY state.
- P0s state: *PCLK* output must stay operational. The MAC may move the PHY to this state only when the transmit channel is idle.
  P0s state can be used when the transmitter is in state *Tx_L0s.Idle*.

  While the PHY is in either P0 or P0s power states, if the receiver is detecting an electrical idle, the receiver portion of the PHY can take appropriate power saving measures. Note that the PHY must be capable of obtaining bit and symbol lock within the PHY-specified time (N_FTS with/without common clock) upon resumption of signaling on the receive channel. This requirement only applies if the receiver had previously been bit and symbol locked while in P0 or P0s states.
- P1 state: Selected internal clocks in the PHY can be turned off. *PCLK* output must stay operational. The MAC will move the PHY to this state only when both transmit and receive channels are idle. The PHY must not indicate successful entry into P1 (by asserting *PhyStatus*) until PCLK is stable and the operating DC common mode voltage is stable and within specification (as per the base spec).
  P1 can be used for the *Disabled* state, all *Detect* states, and *L1.Idle* state of the Link Training and Status State Machine (LTSSM).
- P2 state: Selected internal clocks in the PHY can be turned off. The parallel interface is in an asynchronous mode and *PCLK* output is turned off. The MAC must ensure that the PHY

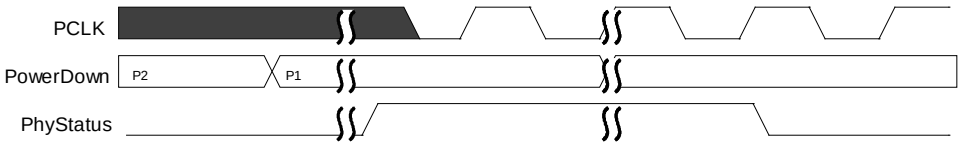
is in 2.5 GT/s signaling mode prior to moving the PHY to P2 state or direct the signaling mode change and PHY power state change at the same time.

When transitioning into P2, the PHY must assert *PhyStatus* before *PCLK* is turned off and then deassert *PhyStatus* when PCLK is fully off and when the PHY is in the P2 state. When transitioning out of P2, the PHY asserts *PhyStatus* as soon as possible and leaves it asserted until after *PCLK* is stable. PHYs should be implemented to minimize power consumption during P2 as this is when the device will have to operate within Vaux power limits (as described in the PCI Express Base Specification).

P2 state can be used in LTSSM states *L2.Idle* and *L2.TransmitWake*.



P2 Entry



P2 Exit

There is a limited set of legal power state transitions that a MAC can ask the PHY to make. Referencing the main state diagram of the LTSSM in the base spec and the mapping of LTSSM states to PHY power states described in the preceding paragraphs, those legal transitions are: P0 to P0s, P0 to P1, P0 to P2, P0s to P0, P1 to P0, and P2 to P0. The base spec also describes what causes those state transitions.

Transitions two and from any pair of PHY power states including at least one PHY specific power state are also allowed by PIPE (unless otherwise prohibited). However, a MAC must ensure that PCI Express specification timing requirements are met.

7.4 Power Management – USB SuperSpeed Mode

The power management signals allow the PHY to minimize power consumption. The PHY must meet all timing constraints provided in the USB 3.0 Specification regarding clock recovery and link training for the various power states. The PHY must also meet all termination requirements for transmitters and receivers.

Four power states are defined, P0, P1, P2, and P3. The P0 state is the normal operational state for the PHY. When directed from P0 to a lower power state, the PHY can immediately take whatever power saving measures are appropriate.

In states P0, P1 and P2, the PHY is required to keep *PCLK* operational. For all state transitions between these three states, the PHY indicates successful transition into the designated power state

by a single cycle assertion of *PhyStatus*. Transitions into and out of P3 are described below. For all power state transitions, the MAC must not begin any operational sequences or further power state transitions until the PHY has indicated that the initial state transition is completed.

Mapping of PHY power states to states in the Link Training and Status State Machine found in the USB specification are included below.

- P0 state: All internal clocks in the PHY are operational. P0 is the only state where the PHY transmits and receives USB SuperSpeed signaling.
  P0 is the appropriate PHY power management state for all cases where the link is in U0 and all other link state except those listed below for P1, P2, and P3.
- P1 state: *PCLK* output must stay operational. The MAC will move the PHY to this state only when the PHY is transmitting idles and receiving idles. The P1 state can be used for the *U1* link state.
- P2 state: Selected internal clocks in the PHY can be turned off. *PCLK* output must stay operational. The MAC will move the PHY to this state only when both transmit and receive channels are idle. The PHY must not indicate successful entry into P2 (by asserting *PhyStatus*) until PCLK is stable and the operating DC common mode voltage is stable and within specification (as per the base spec).
- P2 can be used for the *U2, Rx.Detect, and SS.Inactive*.
- P3 state: Selected internal clocks in the PHY can be turned off. The parallel interface is in an asynchronous mode and *PCLK* output is turned off. When transitioning into P3, the PHY must assert *PhyStatus* before *PCLK* is turned off and then deassert *PhyStatus* when PCLK is fully off and when the PHY is in the P3 state. When transitioning out of P3, the PHY asserts *PhyStatus* as soon as possible and leaves it asserted until after *PCLK* is stable. PHYs should be implemented to minimize power consumption during P3 as this is when the device will have to operate within power limits described in the USB 3.0 Specification.
- The P3 state shall be used in states *SS.disabled and U3.*
- There is a limited set of legal power state transitions that a MAC can ask the PHY to make. Referencing the main state diagram in the USB spec and the mapping of link states to PHY power states described in the preceding paragraphs, those legal transitions are: P0 to P1, P0 to P2, P0 to P3, P1 to P0, P2 to P0, P3 to P0, and P1 to P2. The base spec also describes what causes those state transitions.

7.5 Power Management – SATA Mode

The power management signals allow the PHY to minimize power consumption. The PHY must meet all timing constraints provided in the SATA Specification regarding clock recovery and link training for the various power states. The PHY must also meet all termination requirements for transmitters and receivers.

A minimum of five power states are defined, POWER_STATE_0 and a minimum of four additional states that meet minimum requirements defined in section 6.1. POWER_STATE_0 state is the normal operational state for the PHY. When directed from POWER_STATE_0 to a lower power state, the PHY can immediately take whatever power saving measures are appropriate.

For all state transitions between POWER_STATE_0 and lower power states that provide PCLK, the PHY indicates successful transition into the designated power state by a single cycle assertion of *PhyStatus*. The PHY must complete transmitting all data transferred across the PIPE interface before the change in the PowerDown signals before assertion of *PhyStatus*. Transitions into and out of power states that do not provide PCLK are described below. For all power state transitions, the MAC must not begin any operational sequences or further power state transitions

until the PHY has indicated that the initial state transition is completed. Power state transitions between two power states that do not provide PCLK are not allowed.

Mapping of PHY power states to link states in the SATA specification is MAC specific.

- POWER_STATE_0 : All internal clocks in the PHY are operational. POWER_STATE_0 is the only state where the PHY transmits and receives SATA signaling.
  POWER_STATE_0 is the appropriate PHY power management state for most link states in the SATA specificationWhen transitioning into a power state that does not provide *PCLK* , the PHY must assert *PhyStatus* before *PCLK* is turned off and then deassert *PhyStatus* when PCLK is fully off and when the PHY is in the low power state. The PHY must leave PCLK on for at least one cycle after asserting *PhyStatus.* When transitioning out of a state that does not provide *PCLK* , the PHY asserts *PhyStatus* as soon as possible and leaves it asserted until after *PCLK* is stable.

Transitions between any pair of PHY power states (except two states that do not provide PCLK) are allowed by PIPE. However, a MAC must ensure that SATA specification timing requirements are met.

7.6 Changing Signaling Rate, PCLK Rate, or Data Bus Width

7.6.1 PCI Express Mode

The signaling rate of the link, PCLK rate, or the Data Bus Width can be changed only when the PHY is in the P0 or P1 power state and *TxElecIdle* and *RxStandby are* asserted. The rate can be changed, or the width can be changed, or the PCLK rate can be changed, or any combination of the rate and width and PCLK rate, can be changed simultaneously. When the MAC changes the *Rate* signal, and/or the *Width* signal, and/or the *PCLK rate* signal,the PHY performs the rate change and/or the width change and/or the PCLK rate change and signals its completion with a single cycle assertion of *PhyStatus*. The MAC must not perform any operational sequences, power state transitions, deassert *TxElecIdle* or *RxStandby,* or further signaling rate changes until the PHY has indicated that the signaling rate change has completed.

There are instances where LTSSM state machine transitions indicate both a speed change and/or with and/or PCLK rate change and a power state change for the PHY. One of these instances is when the LTSSM transitions to Detect. In this case, the MAC must change (if necessary) the signaling rate to 2.5 GT/s and change the width and/or PCLK rate if necessary before changing the power state to P1. Another instance is when the LTSSM transitions to L2.Idle. Again, the MAC must change (if necessary) the signaling rate to 2.5 GT/s and change the width and/or PCLK rate if necessary before changing the power state to P2.

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. If a PHY supports this, the MAC must change the rate and/or width and/or PCLK rate at the same PCLK edge that it changes the *PowerDown* signals. This can happen when transition the PHY from P0 to either P1 or P2 states. The completion mechanisms are the same as previously defined for the power state changes and indicate not only that the power state change is complete, but also that the rate and/or width and/or PCLK rate change is complete.

7.6.2 SATA Mode

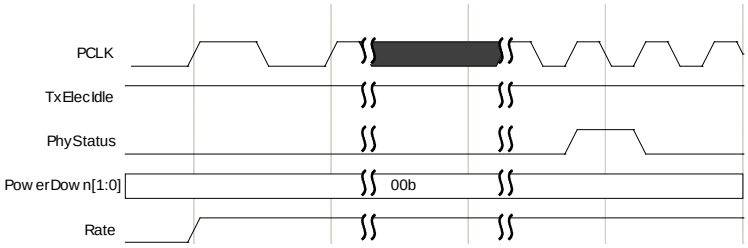
The signaling rate of the link, PCLK rate, or the Data Bus Width can be changed only when the PHY is in POWER_STATE_0 and *TxElecIdle* and *RxStandby are asserted*, or in a lowpower state where PCLK is provided. The rate can be changed, or the width can be changed, or the PCLK rate can be changed, or any combination of the rate and width and PCLK rate, can be changed simultaneously. When the MAC changes the *Rate* signal, and/or the *Width* signal, and/or the *PCLK rate* signal, the PHY performs the rate change and/or the width change and/or the PCLK rate change and signals its completion with a single cycle assertion of *PhyStatus*. The MAC must not perform any operational sequences, power state transitions, deassert *TxElecIdle* or *RxStandby*, or further signaling rate and/or width changes until the PHY has indicated that the change has completed.

There are instances where conditions indicate both a speed change and/or width and/or PCLK rate change and a power state change for the PHY. In such cases the MAC must change the signaling rate and/or width and/or PCLK rate, before changing the power state.

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. If a PHY supports this, the MAC must change the rate and/or width and/or PCLK rate at the same PCLK edge that it changes the *PowerDown* signals. The completion mechanisms are the same as previously defined for the power state changes and indicate not only that the power state change is complete, but also that the rate and/or width and/or PCLK rate change is complete.

7.6.3 Fixed data path implementations

The figure below shows logical timings for implementations that change PCLK frequency when the MAC changes the signaling rate. Implementations that change the *PCLK* frequency when changing signaling rates must change the clock such that the time the clock is stopped (if it is stopped) is minimized to prevent any timers using *PCLK* from exceeding their specifications. Also during the clock transition period, the frequency of *PCLK* must not exceed the PHY's defined maximum clock frequency. The amount of time between when *Rate* is changed and the PHY completes the rate change is a PHY specific value. These timings also apply to implementations that keep the data path fixed by using options that make use of the TxDataValid and RxDataValid signals.

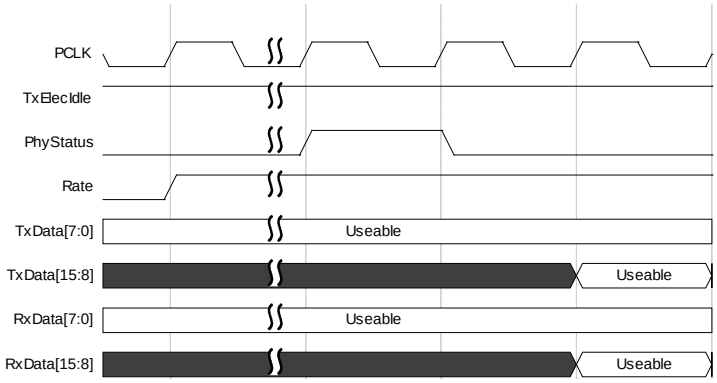


Rate change with fixed data path

7.6.4 Fixed PCLK implementations

The figure below shows logical timings for implementations that change the width of the data path for different signaling rates. PCLK may be stopped during a rate change. These timings

also apply to fixed PCLK implementations that make use of the TxDataValid and RxDataValid signals.

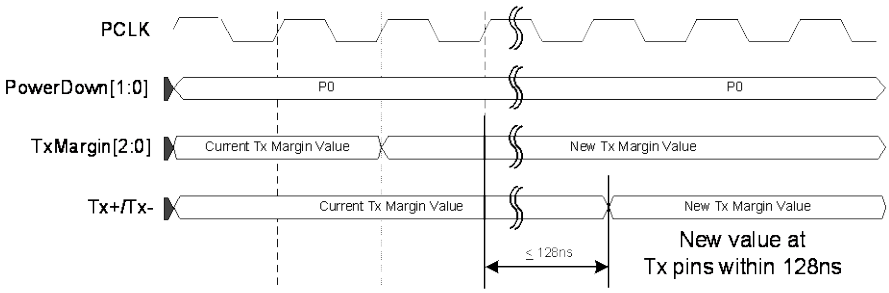


Rate change with fixed PCLK frequency

7.7 Transmitter Margining – PCI Express Mode and USB SuperSpeed Mode

While in the P0 power state, the PHY can be instructed to change the value of the voltage at the transmitter pins. When the MAC changes *TxMargin[2:0]*, the PHY must be capable of transmitting with the new setting within 128 ns.

There is a limited set of legal *TxMargin[2:0]* and *Rate* combinations that a MAC can select. Refer to the PCIe Base Specification for a complete description of legal settings when the PHY is in PCI Express Mode. Refer to the USB specification for a complete description of the legal settings when the PHY is in USB SuperSpeed mode.



Selecting Tx Margining value

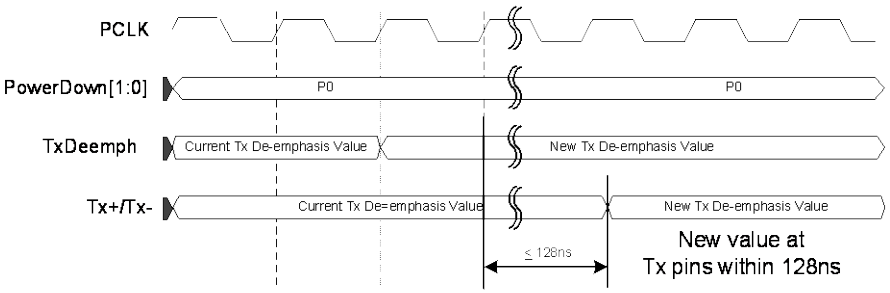
7.8 Selectable De-emphasis – PCI Express Mode

While in the P0 power state and transmitting at 5.0GT/s or 8.0 GT/s, the PHY can be instructed to change the value of the transmitter equalization. When the signaling rate is 5.0 GT/s and the

MAC changes *TxDeemph*, the PHY must be capable of transmitting with the new setting within 128 ns. When the signaling rate is 8.0 GT/s and the MAC changes *TxDeemph*, the PHY must be capable of transmitting with the new setting within 256 ns.

There is a limited set of legal *TxDeemph* and *Rate* combinations that a MAC can select. Refer to the PCIe Base Specification for a complete description.

The MAC must ensure that *TxDeemph* is selecting -3.5db whenever *Rate* is selecting 2.5 GT/s.

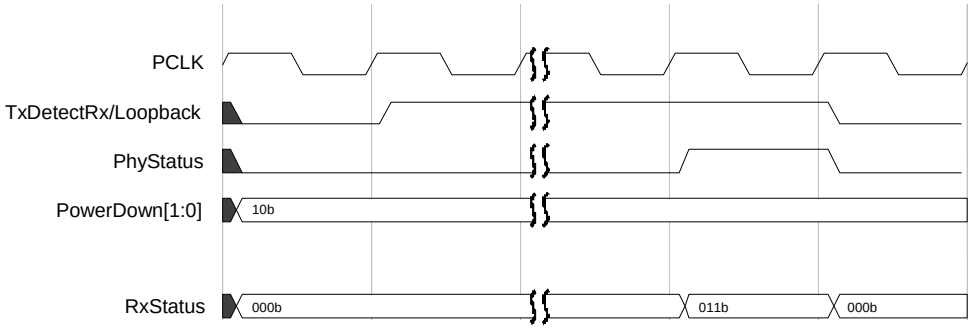


Selecting Tx De-emphasis value

7.9 Receiver Detection – PCI Express Mode and USB SuperSpeed Mode

While in the P1 power state and PCI Express mode or in the P2 or P3 power state and USB SuperSpeed mode, the PHY can be instructed to perform a receiver detection operation to determine if there is a receiver at the other end of the link. Basic operation of receiver detection is that the MAC requests the PHY to do a receiver detect sequence by asserting *TxDetectRx/Loopback*. When the PHY has completed the receiver detect sequence, it asserts *PhyStatus* for one clock and drives the *RxStatus* signals to the appropriate code. After the receiver detection has completed (as signaled by the assertion of *PhyStatus*), the MAC must deassert *TxDetectRx/Loopback* before initiating another receiver detection, a power state transition, or signaling a rate change.
Once the MAC has requested a receiver detect sequence (by asserting *TxDetectRx/Loopback*), the MAC must leave *TxDetectRx/Loopback* asserted until after the PHY has signaled completion by the assertion of *PhyStatus*. When receiver detection is performed in USB SuperSpeed mode with the PHY in P3 the PHY asserts PhyStatus and signals the appropriate receiver detect value until the MAC deasserts TxDetectRx/Loopback.

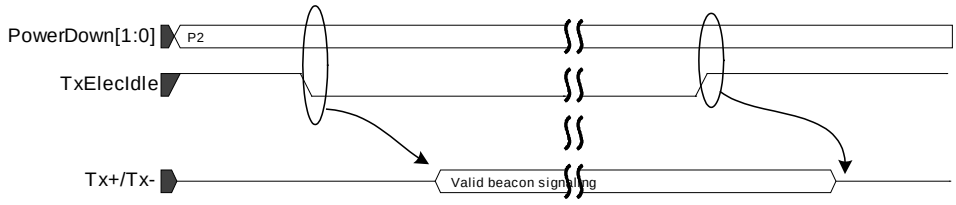
Detected Condition	*RxStatus* code
Receiver not present	000b
Receiver present	011b



Receiver Detect - Receiver present

7.10 Transmitting a beacon – PCI Express Mode

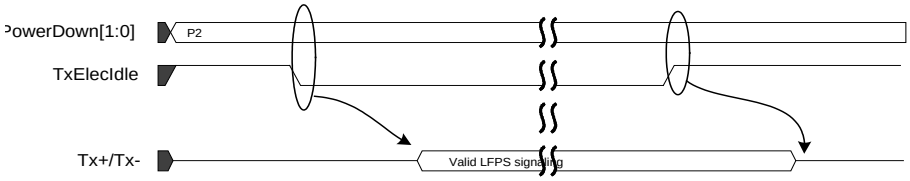
When the PHY has been put in the P2 power state, and the MAC wants to transmit a beacon, the MAC deasserts *TxElecIdle* and the PHY should generate a valid beacon until *TxElecIdle* is asserted. The MAC must assert *TxElecIdle* before transitioning the PHY to P0.



Beacon Transmit

7.11 Transmitting LFPS – USB SuperSpeed Mode

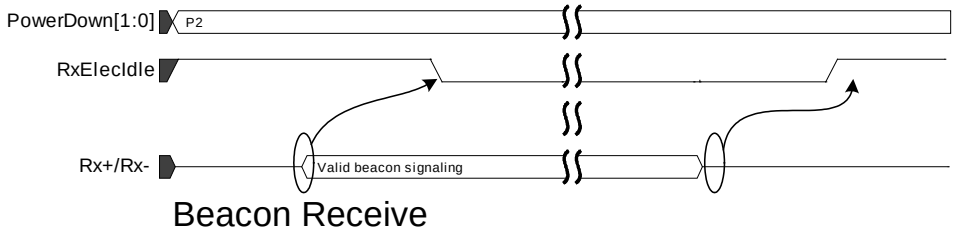
When the PHY is in P1, P2, or P3 and the MAC wants to transmit LFPS, the MAC deasserts *TxElecIdle* and the PHY should generate valid LFPS until *TxElecIdle* is asserted. The MAC must assert *TxElecIdle* before transitioning the PHY to P0. The length of time *TxElecIdle* is deasserted is varied for different events. When the PHY is in P0 and the MAC wants to transmit LFPS for Polling, the MAC must assert both *TxElecIdle* and *TxDetectRx/Loopback* for the duration defined for a Polling.LFPS burst. Refer to chapter 5 in the USB 3.0 specification for more details.



LFPS Transmit

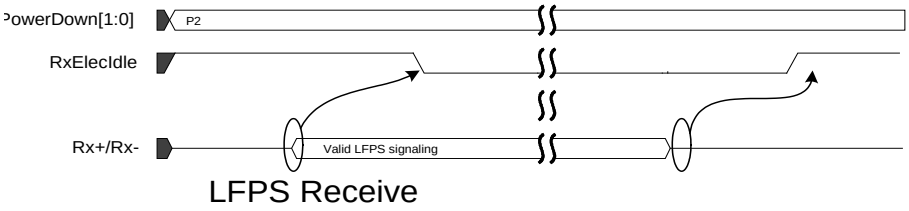
7.12 Detecting a beacon – PCI Express Mode

The PHY receiver must monitor at all times (except during reset) for electrical idle. When the PHY is in the P2 power state, and *RxElecIdle* is deasserted, then a beacon is being detected.



7.13 Detecting Low Frequency Periodic Signaling – USB SuperSpeed Mode

The PHY receiver must monitor at all times (except during reset and when RX terminations are removed) for LFPS. When the PHY is in the P0, P1, P2, or P3 power state, and *RxElecIdle* is deasserted, then LFPS is being detected. The length of time *RxElecIdle* is deasserted indicates the length of time Low Frequency Periodic Signaling is detected. Refer to chapter 5 in the USB 3.0 specification for more details on the length of Low Frequency Periodic Signaling (LFPS) for various events.



7.14 Clock Tolerance Compensation

The PHY receiver contains an elastic buffer used to compensate for differences in frequencies between bit rates at the two ends of a Link. The elastic buffer must be capable of holding enough symbols to handle worst case differences in frequency and worst case intervals between symbols that can be used for rate compensation for the selected PHY mode as shown in Table 7-1

Phy Mode	Worst Case Frequency Offset	Symbol Depth – Nominal Half Full Buffer	Symbol Depth – Nominal Empty Buffer
PCI Express	600 ppm	7 symbols	N/A
USB SuperSpeed	5600 ppm	16 symbols (8 SKP ordered sets)	8 symbols (4 SKP ordered sets)
SATA	5700~~600~~ ppm	15 symbols	8 symbols

Table 7-1 Minimum Elasticity Buffer Size

For the Nominal Empty buffer model the PHY attempts to keep the elasticity buffer as close to empty as possible. In Nominal Empty mode the PHY uses the RxDataValid interface to tell the

MAC when no data is available. The Nominal Empty buffer model provides a smaller worst case and average latency then the Nominal Half Full buffer model, but requires the MAC to support the RxDataValid signal.

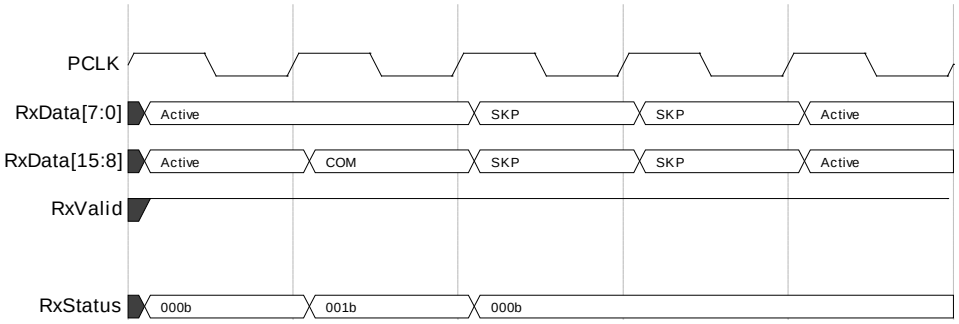
Two models are defined for the elastic buffer operation in the PHY. The PHY may support one or both of these models. The Nominal Empty buffer model is only supported in USB SuperSpeed or SATA Mode.
For the Nominal Half Full buffer model, the PHY is responsible for inserting or removing SKP symbols, ordered sets, or ALIGNs in the received data stream to avoid elastic buffer overflow or underflow. The PHY monitors the receive data stream, and when a Skip ordered-set or ALIGN is received, the PHY can add or remove one SKP symbol (PCI Express Mode) or one SKP ordered set (USB SuperSpeed Mode) or one ALIGN ~~symbol~~ from each SKP or ALIGN as appropriate to manage its elastic buffer to keep the buffer as close to half full as possible. In USB SuperSpeed mode the PHY shall only add or remove SKP ordered sets. Whenever a SKP symbol or ordered set is added to or removed, the PHY will signal this to the MAC using the *RxStatus[2:0]* signals. These signals have a non-zero value for one clock cycle and indicate whether a SKP symbol or ordered set was added to or removed from the received SKP ordered-set(s). In PCI Express Mode, *RxStatus* shall be asserted during the clock cycle when the COM symbol of the SKP ordered-set is moved across the parallel interface. In SATA Mode whenever a ALIGN symbol is added or removed, the PHY will signal this to the MAC using the *RxStatus[2:0]* signals. These signals have a non-zero value for one clock cycle and indicate whether an ALIGN was added or removed. *RxStatus* shall be asserted during the clock cycle when the first symbol of the ALIGN is moved across the parallel interface.

In USB SuperSpeed mode for the Nominal Empty buffer model the PHY attempts to keep the elasticity buffer as close to empty as possible. This means that the PHY will be required to insert SKP ordered sets into the received data stream when no SKP ordered sets have been received. The Nominal Empty buffer model provides a smaller worst case and average latency then the Nominal Half Full buffer model, but requires the MAC to support receiving SKP ordered sets any point in the data stream.

In SATA mode for the Nominal Empty buffer model the PHY attempts to keep the elasticity buffer as close to empty as possible. In Nominal Empty mode the PHY uses the RxDataValid interface to tell the MAC when no data is available. The Nominal Empty buffer model provides a smaller worst case and average latency then the Nominal Half Full buffer model, but requires the MAC to support the RxDataValid signal.

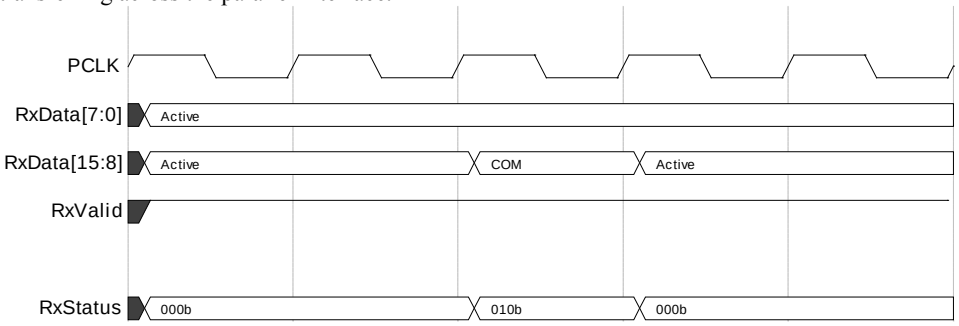
Starting with this version of the specification a PHY and MAC are allowed to support the Nominal Empty buffer model in USB mode using the RxDataValid signal. Inserting SKPs in the data stream when no SKPs have been received is not recommended. ~~deprecated.~~

The figure below shows a sequence where a PHY operating in PCI Express Mode added a SKP symbol in the data stream.



Clock Correction - Add a SKP

The figure below shows a sequence where a PHY operating in PCI Express mode removed a SKP symbol from a SKP ordered-set that only had one SKP symbol, resulting in a 'bare' COM transferring across the parallel interface.



Clock Correction - Remove a SKP

7.15 Error Detection

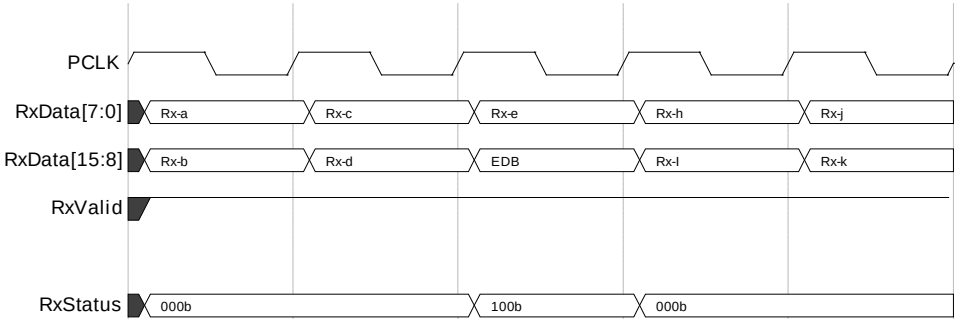
The PHY is responsible for detecting receive errors of several types. These errors are signaled to the MAC layer using the receiver status signals (*RxStatus[2:0]*). Because of higher level error detection mechanisms (like CRC) built into the Data Link layer there is no need to specifically identify symbols with errors, but reasonable timing information about when the error occurred in the data stream is important. When a receive error occurs, the appropriate error code is asserted for one clock cycle at the point in the data stream across the parallel interface closest to where the error actually occurred. There are four error conditions (five for SATA mode) that can be encoded on the *RxStatus* signals. If more than one error should happen to occur on a received byte (or set of bytes transferred across a 16-bit interface or 32-bit interface), the errors should be signaled with the priority shown below.

1. 8B/10B decode error
2. Elastic buffer overflow
3. Elastic buffer underflow (Cannot occur in Nominal Empty buffer model)
4. Disparity errors
5. Misalign (SATA mode only)

If an error occurs during a SKP ordered-set or ALIGN, such that the error signaling and SKP or ALIGN added/removed signaling on *RxStatus* would occur on the same CLK, then the error signaling has precedence.

7.15.1 8B/10B Decode Errors

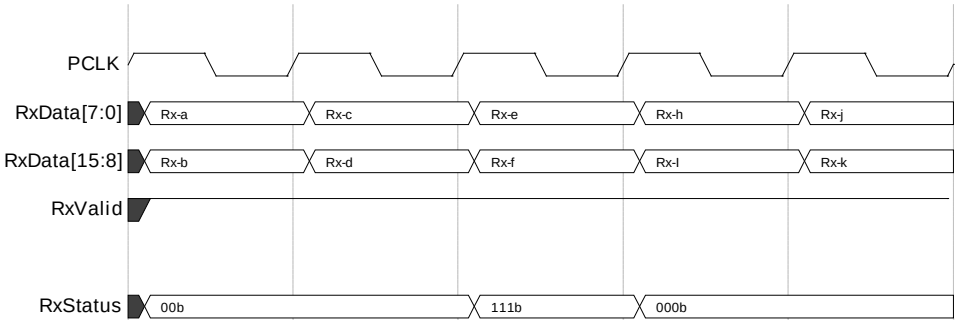
For a detected 8B/10B decode error, the PHY should place an EDB symbol (for PCIe or SATA) or SUB symbol (for USB) in the data stream in place of the bad byte, and encode *RxStatus* with a decode error during the clock cycle when the effected byte is transferred across the parallel interface. In the example below, the receiver is receiving a stream of bytes Rx-a through Rx-z, and byte Rx-f has an 8B/10B decode error. In place of that byte, the PHY places an EDB (for PCIe or SATA) or SUB (for USB) on the parallel interface, and sets *RxStatus* to the 8B/10B decode error code. Note that a byte that can't be decoded may also have bad disparity, but the 8B/10B error has precedence. Also note that for a 16-bit or 32-bit interface, if the bad byte is on the lower byte lane, one of the other bytes may have bad disparity, but again, the 8B/10B error has precedence.



8B/10B Decode Error

7.15.2 Disparity Errors

For a detected disparity error, the PHY should assert *RxStatus* with the disparity error code during the clock cycle when the affected byte is transferred across the parallel interface. For 16-bit interfaces, it is not possible to discern which byte (or possibly both) had the disparity error. In the example below, the receiver detected a disparity error on either (or both) Rx-e or Rx-f data bytes, and indicates this with the assertion of *RxStatus*. Optionally, the PHY can signal disparity errors as 8B/10B decode error (using code 0b100). (MACs often treat 8B/10B errors and disparity errors identically.). When operating in USB SuperSpeed mode signaling disparity errors is optional.

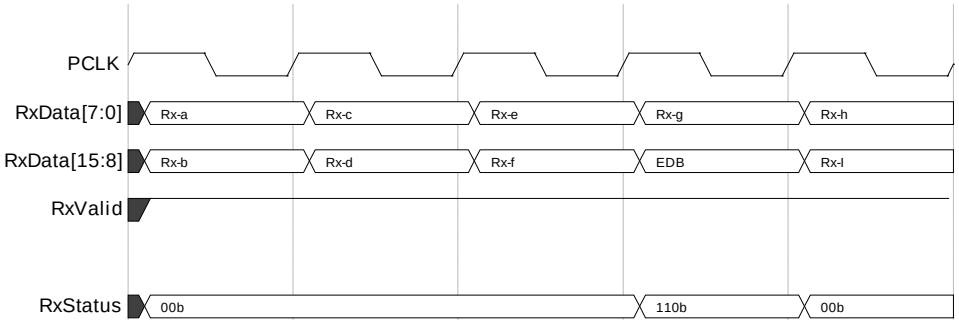


Disparity Error

7.15.3 Elastic Buffer Errors

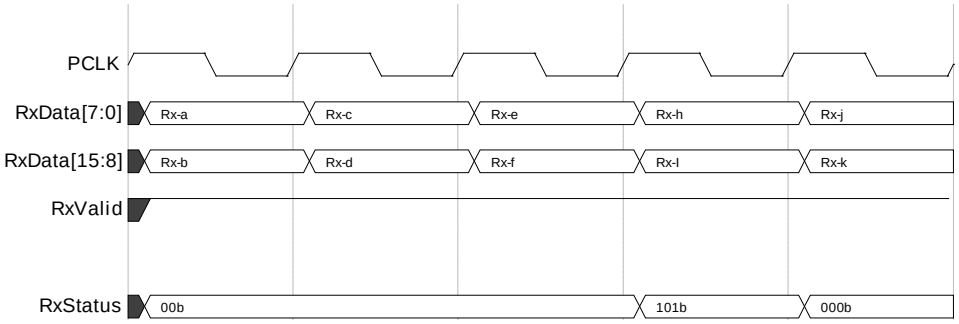
For elastic buffer errors, an underflow should be signaled during the clock cycle or clock cycles when a spurious symbol is moved across the parallel interface. The symbol moved across the interface should be the EDB symbol (for PCIe or SATA) or SUB symbol (for USB). In the timing diagram below, the PHY is receiving a repeating set of symbols Rx-a thru Rx-z. The elastic buffer underflows causing the EDB symbol (for PCIe) or SUB symbol (for USB) to be inserted between the Rx-g and Rx-h Symbols. The PHY drives *RxStatus* to indicate buffer underflow during the clock cycle when the EDB (for PCIe) or SUB (for USB) is presented on the parallel interface.

Note that underflow is not signaled when the PHY is operating in Nominal Empty buffer mode. In this mode SKP ordered sets are moved across the interface whenever data needs to be inserted or the RxDataValid signal is used. The RxDataValid method is preferred.



Elastic Buffer Underflow

For an elastic buffer overflow, the overflow should be signaled during the clock cycle where the dropped symbol or symbols would have appeared in the data stream. For the 16-bit interface it is not possible, or necessary, for the MAC to determine exactly where in the data stream the symbol was dropped. In the timing diagram below, the PHY is receiving a repeating set of symbols Rx-a thru Rx-z. The elastic buffer overflows causing the symbol Rx-g to be discarded. The PHY drives *RxStatus* to indicate buffer overflow during the clock cycle when Rx-g would have appeared on the parallel interface.



Elastic Buffer Overflow

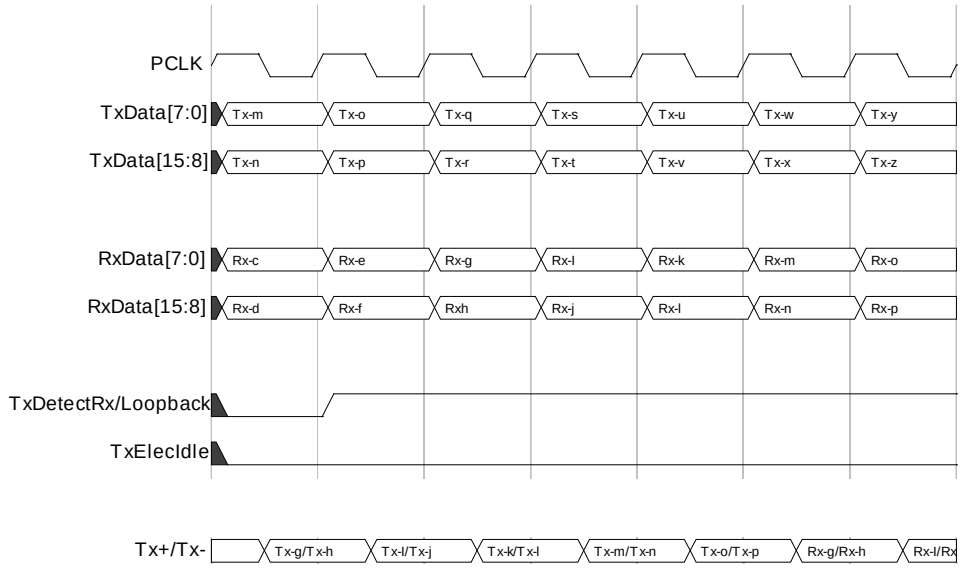
7.16 Loopback

- For USB SuperSpeed and PCI Express Modes he PHY must support an internal loopback as described in the corresponding base specification.
- For SATA the PHY may optionally support an internal loopback mode when EncodeDecodeBypass is asserted.

The PHY begins to loopback data when the MAC asserts *TxDetectRx/Loopback* while doing normal data transmission (i.e. when *TxElecIdle* is deasserted). The PHY must, within the specified receive and transmit latencies, stop transmitting data from the parallel interface, and begin to loopback received symbols. While doing loopback, the PHY continues to present received data on the parallel interface.

The PHY stops looping back received data when the MAC deasserts *TxDetectRx/Loopback*. Transmission of data on the parallel interface must begin within the specified transmit latency.

The timing diagram below shows example timing for beginning loopback. In this example, the receiver is receiving a repeating stream of bytes, Rx-a thru Rx-z. Similarly, the MAC is causing the PHY to transmit a repeating stream of bytes Tx-a thru Tx-z. When the MAC asserts *TxDetectRx/Loopback* to the PHY, the PHY begins to loopback the received data to the differential *Tx+/Tx-* lines. Timing between assertion of *TxDetectRx/Loopback* and when Rx data is transmitted on the Tx pins is implementation dependent.

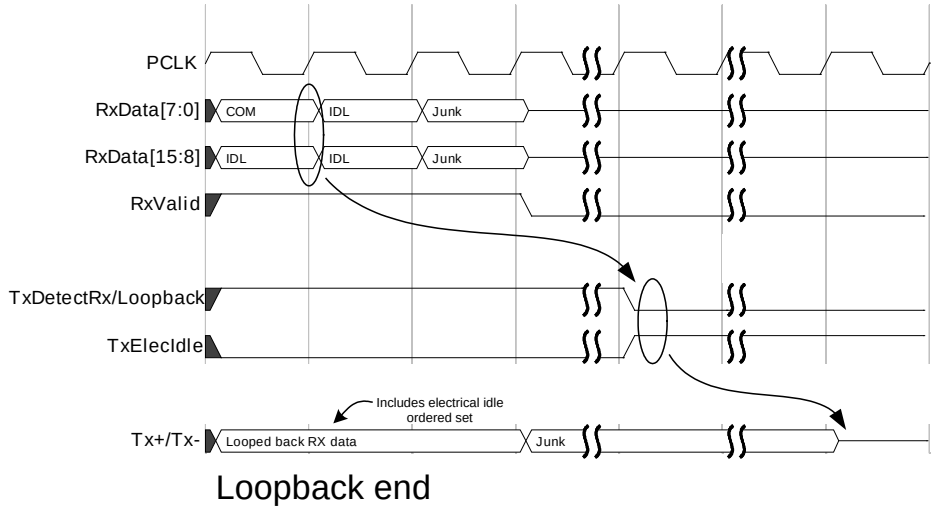


Loopback start

The next timing diagram shows an example of switching from loopback mode to normal mode when the PHY is operating in PCI Express Mode.

In PCI Express Mode, when the MAC detects an electrical idle ordered-set, the MAC deasserts *TxDetectRx/Loopback* and asserts *TxElecIdle*. The PHY must transmit at least three bytes of the electrical idle ordered-set before going to electrical idle. (Note, transmission of the electrical idle ordered-set should be part of the normal pipeline through the PHY and should not require the PHY to detect the electrical idle ordered-set). The base spec requires that a Loopback Slave be able to detect and react to an electrical idle ordered set within 1ms. The PHY's contribution to this time consists of the PHY's Receive Latency plus the PHY's Transmit Latency (see section 6.13).

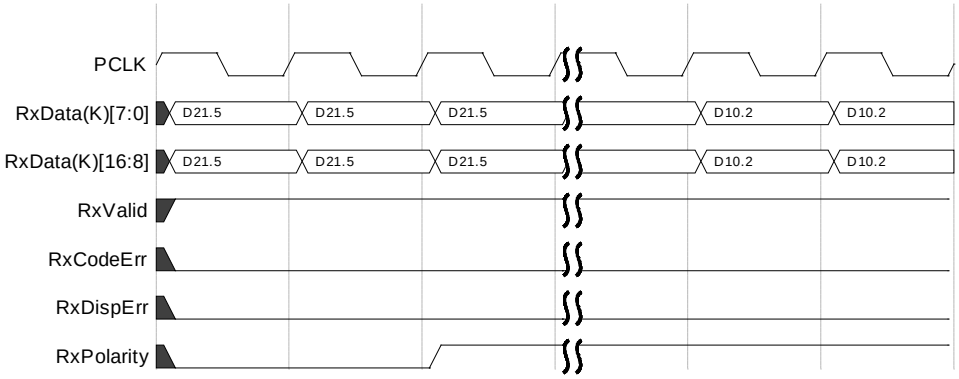
When the PHY is operating in USB SuperSpeed Mode, the device shall only transition out of loopback on detection of LFPS signaling (reset) or when VBUS is removed. When valid LFPS signaling is detected, the MAC transitions the PHY to the P2 power state in order to begin the LFPS handshake.



Loopback end

7.17 Polarity Inversion – PCI Express and USB SuperSpeed Modes

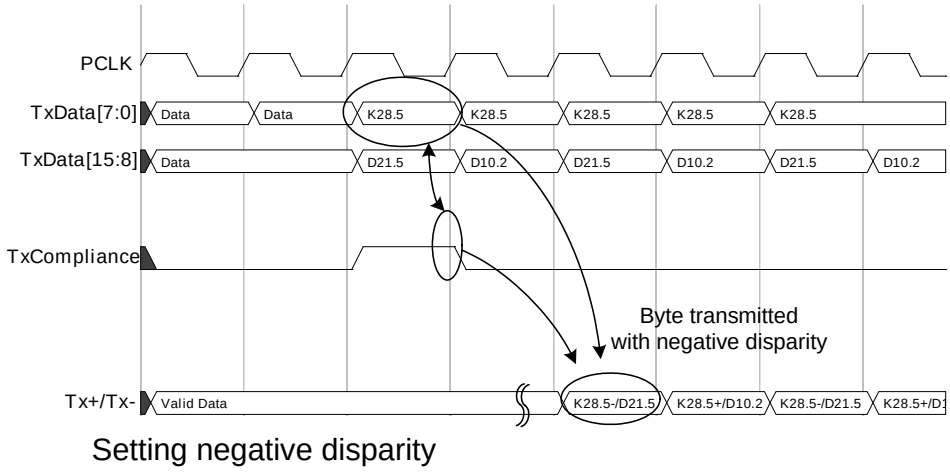
To support lane polarity inversion, the PHY must invert received data when *RxPolarity* is asserted. Inversion can happen in many places in the receive chain, including somewhere in the serial path, as symbols are placed into the elastic buffer, or as symbols are removed from the elastic buffer. Inverted data must begin showing up on *RxData[]* within 20 PCLKs of when *RxPolarity* is asserted.



Polarity inversion

7.18 Setting negative disparity (PCI Express Mode)

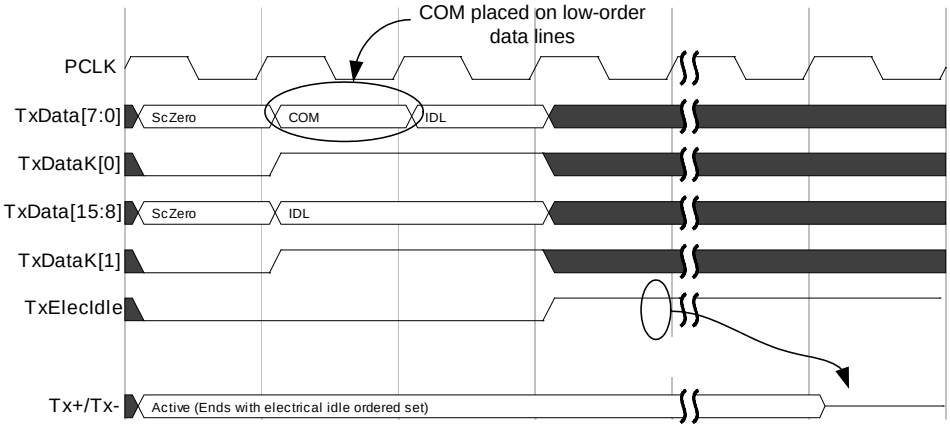
To set the running disparity to negative, the MAC asserts *TxCompliance* for one clock cycle that matches with the data that is to be transmitted with negative disparity. For a 16-bit interface, the low order byte will be the byte transmitted where running disparity is negative. The example shows how *TxCompliance* is used to transmit the PCI Express compliance pattern in PCI Express mode. *TxCompliance* is only used in PCI Express mode.



Setting negative disparity

7.19 Electrical Idle – PCI Express Mode

The base spec requires that devices send an Electrical Idle ordered set before Tx+/Tx- goes to the electrical idle state. For a 16-bit interface or 32-bit interface, the MAC must always align the electrical idle ordered set on the parallel interface so that the COM symbol is on the low-order data lines (*TxDataK[7:0]*).



Electrical Idle

7.20 Link Equalization Evaluation

While in the P0 power state, the PHY can be instructed to perform evaluation of the current TX equalization settings of the link partner. Basic operation of the equalization evaluation is that the MAC requests the PHY to evaluate the current equalization settings by asserting *RxEqEval*. When the PHY has completed evaluating the current equalization settings, it asserts *PhyStatus* for one clock and drives the *LinkEvaluationFeedback* signals to the appropriate feedback response.

After link equalization evaluation has completed (as signaled by the assertion of *PhyStatus*), the MAC must deassert *RxEqEval* before initiating another evaluation.
Once the MAC has requested link equalization evaluation (by asserting *RxEqEval*), the MAC must leave *RxEqEval* asserted until after the PHY has signaled completion by the assertion of *PhyStatus* unless the MAC needs to abort the evaluation due to high level timeouts or error conditions. To abort an evaluation the MAC de-asserts RxEqEval before the PHY has signaled completion. If the MAC aborts the evaluation the PHY must signal completion as quickly as possible. The MAC ignores returned evaluation values in an abort scenario.

Note: If a race condition occurs where the MAC aborts by deasserting RxEqEval on same cycle as the PHY asserts PhyStatus then the PHY shall not take any further action.

Formatted: Font: (Default) Times New Roman, 11 pt

Formatted: Font: (Default) Times New Roman, 11 pt

Formatted: Font: (Default) Times New Roman, 11 pt

Formatted: Font: (Default) Times New Roman, 11 pt

7.21 Implementation specific timing and selectable parameter support

PHY vendors (macrocell or discrete) must specify typical and worst case timings for the cases listed in the table below.

Transmit Latency	Time for data moving between the parallel interface and the PCI Express, SATA or USB SuperSpeed serial lines. Timing is measured from when the data is transferred across the parallel interface (i.e. the rising edge of *PCLK*) and when the first bit of the equivalent 10-bit symbol is transmitted on the *Tx+/Tx-* serial lines. The PHY reports the latency for each operational mode the PHY supports. Note: If the transmit latency is different when EncodeDecodeBypass is asserted – the PHY must report this latency separately.
Receive Latency	Time for data moving between the parallel interface and the PCI Express, SATA or USB SuperSpeed serial lines. Timing is measured from when the first bit of a 10-bit symbol is available on the *Rx+/Rx-* serial lines to when the corresponding 8-bit data is transferred across the parallel interface (i.e. the rising edge of *PCLK*). The PHY reports the latency for each operational mode the PHY supports. Note: If the receive latency is different when EncodeDecodeBypass is asserted – the PHY ust report this latency separately.
Power State After Reset	The PHY power state immediately following reset. The state after reset needs to provide PCLK and have common mode off. Reporting this parameter is required if the PHY supports either SATA mode or PCI Express mode at 8 GT/s.
Loopback enable latency	Amount of time it takes the PHY to begin looping back receive data. Timed from when *TxDetectRx/Loopback* is asserted until the receive data is being transmitted on the serial pins. The PHY reports the latency for each operational mode the pHY supports.
Transmit Beacon – PCI Express	Timed from when the MAC directs the PHY to send a

Mode.	beacon (power state is P2 and *TxElecIdle* is deasserted) until the beacon signaling begins at the serial pins.
Receive Beacon – PCI Express Mode	Timed from when valid beacon signaling is present at the receiver pins until *RxElecIdle* is deasserted.
Transmit LFPS – USB SuperSpeed Mode	Timed from when the MAC directs the PHY to send LFPS signaling until the LFPS signaling begins at the serial pins. Times are reported for each possible P state if the times are different for different power states.
Receive LFPS – USB SuperSpeed Mode	Timed from when valid LFPS signaling is present at the receiver pins until *RxElecIdle* is deasserted.
N_FTS with common clock (PCI Express Mode)	Number of FTS ordered sets required by the receiver to obtain reliable bit and symbol lock when operating with a common clock.
N_FTS without common clock (PCI Express Mode)	Number of FTS ordered sets required by the receiver to obtain reliable bit and symbol lock when operating without a common clock.
PHY lock time	Amount of time for the PHY receiver to obtain reliable bit and symbol lock after valid symbols are present at the receiver. The PHY reports the time for each operational mode the PHY supports.
P0s to P0 transition time PCI Express Mode.	Amount of time for the PHY to return to P0 state, after having been in the P0s state. Time is measured from when the MAC sets the *PowerDown* signals to P0 until the PHY asserts *PhyStatus*. PHY asserts *PhyStatus* when it is ready to begin data transmission and reception.
P1 to P0 transition time. PCI Express Mode.	Amount of time for the PHY to return to P0 state, after having been in the P1 state. Time is measured from when the MAC sets the *PowerDown* signals to P0 until the PHY asserts *PhyStatus*. PHY asserts *PhyStatus* when it is ready to begin data transmission and reception.
P2 to P1 transition time PCI Express Mode.	Amount of time for the PHY to go to P1 state, after having been in the P2 state. Time is measured from when the MAC sets the *PowerDown* signals to P1 until the PHY deasserts *PhyStatus*.
P1 to P0 transition time. USB SuperSpeed Mode.	Amount of time for the PHY to return to P0 state, after having been in the P1 state. Time is measured from when the MAC sets the *PowerDown* signals to P0 until the PHY asserts *PhyStatus*. PHY asserts *PhyStatus* when it is ready to begin data transmission and reception.
P2 to P0 transition time. USB SuperSpeed Mode.	Amount of time for the PHY to return to P0 state, after having been in the P2 state. Time is measured from when the MAC sets the *PowerDown* signals to P0 until the PHY asserts *PhyStatus*. PHY asserts *PhyStatus* when it is ready to begin data transmission and reception.
P3 to P0 transition time USB SuperSpeed Mode.	Amount of time for the PHY to go to P0 state, after having been in the P3 state. Time is measured from when the MAC sets the *PowerDown* signals to P0 until the PHY deasserts *PhyStatus*. PHY asserts *PhyStatus* when it is

<table>
<tr><td></td><td>ready to begin data transmission and reception.</td></tr>
<tr><td>Power state transition times between two power states that provide PCLK.</td><td>Amount of time for the PHY to transition to a new power state. Time is measured from when the MAC sets the PowerDown signals to POWER_STATE_X until the PHY asserts PhyStatus. PHY asserts PhyStatus when it is ready to begin data transmission and reception. The PHY reports this transition between each pair of power states it supports in each PHY mode it supports.</td></tr>
<tr><td>Power state transition times between a power state without PCLK and a power state with PCLK.</td><td>Amount of time for the PHY to go to a power state providing PCLK, after having been in a power state that does not provide PCLK. Time is measured from when the MAC sets the PowerDown signals to the new power state until the PHY deasserts PhyStatus. The PHY reports this time for each possible transition between a power state that does not provide PCLK and a power state that does provide PCLK. The PHY reports this transition time between each pair of power states it supports in each PHY mode it supports.</td></tr>
<tr><td>Reset to ready time</td><td>Timed from when Reset# is deasserted until the PHY deasserts PhyStatus.</td></tr>
<tr><td>Supported power states.</td><td>The PHY lists each power state it supports for each PHY mode it supports. For each power state supported it reports whether PCLK is provided, the exit latency, and the common mode state.
Note: This is done for all states not already listed separately.</td></tr>
<tr><td>Simultaneous Rate and Power State Change</td><td>The PHY reports if it supports simultaneous rate and power state changes for each PHY mode it supports.</td></tr>
<tr><td>Data Rate change time. PCI Express Mode and SATA Mode.</td><td>Amount of time the PHY takes to perform a data rate change. Time is measured from when the MAC changes Rate to when the PHY signals rate change complete with the single clock assertion of PhyStatus. There may be separate values for each possible change between different supported rates for each supported PHY mode.</td></tr>
<tr><td>Transmit Margin values supported. PCI Express Mode and USB SuperSpeed Mode.</td><td>Transmitter voltage levels.
<table>
<tr><td>[2]</td><td>[1]</td><td>[0]</td><td>Description</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>TxMargin value 0 =</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>TxMargin value 1 =</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>TxMargin value 2 =</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>TxMargin value 3 =</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>TxMargin value 4 =</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>TxMargin value 5 =</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>TxMargin value 6 =</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>TxMargin value 7 =</td></tr>
</table></td></tr>
<tr><td>Max Equalization Settings for C_{-1}</td><td>Reports the maximum number of settings supported by the PHY for the 8.0 GT/s equalization. The maximum number of settings must be less than 64.</td></tr>
<tr><td>Max Equalization Settings for C_0</td><td>Reports the maximum number of settings supported by the PHY for the 8.0 GT/s equalization. The maximum number of settings must be less than 64.</td></tr>
</table>

Max Equalization Settings for C_1	Reports the maximum number of settings supported by the PHY for the 8.0 GT/s equalization. The maximum number of settings must be less than 64.
Default Equalization settings for full swing preset Pn.	Reports the recommended setting values for C_{-1}, C_0, C_1 for each full swing preset.
Default Equalization settings for half swing preset Pn.	Reports the recommended setting values for C_{-1}, C_0, C_1 for each half swing preset.
Dynamic Preset Coefficient Update Support	A PHY indicates if it dynamically updates coefficients.
Figure of Merit range	If the PHY reports link equalization feedback in the Figure of Merit format it reports the maximum value it will report. The maximum value must be less than 256.
Figure of Merit for BER target	If the PHY reports link equalization feedback in the Figure of Merit format it reports the minimum value that the PHY estimates corresponds to a link BER of E-12.
Default Link Partner Preset[3:0]	If the PHY prefers the link parter to start with a specific preset during link evaluation it reports the preferred starting preset. The default link partner preset value is encoded as follows: 0000b – Preset P0. 0001b – Preset P1. 0010b – Preset P2. 0011b – Preset P3. 0100b – Preset P4. 0101b – Preset P5. 0110b – Preset P6. 0111b – Preset P7. 1000b – Preset P8. 1001b – Preset P9. 1010b – Preset P10. 1011b – Reserved 1100b – Reserved 1101b – Reserved 1110b – Reserved 1111b – No Preference.
Beacon Support	The PHY indicates whether it supports beacon transmission. Beacon transmission is optional. 1: Beacon transmission is supported. 0: Beacon transmission is not supported.
EncodeDecodeBypassSupport[3:0]	The PHY indicates whether it supports optional EncodeDecodeBypass mode at each signaling rate. [0] Support at 2.5 GT/s [1] Support at 5.0 GT/s [2] Support at 8.0 GT/s [3] Reserved The support value for each rate is encoded as follows:

	0 - No support for EncodeDecodeBypass 1 – Support for EncodeDecodeBypass
NoDeemphasisSupport[1:0]	The PHY indicates whether it supports an optional No De-emphasis signaling mode at 2.5 and 5.0 GT/s signaling rates. [0] Support at 2.5 GT/s [1] Support at 5.0 GT/s The support value for each rate is encoded as follows: 0 – No support for a no de-emphasis signaling mode. 1 – Support for a no de-emphasis signaling mode.
SupportedLFPresets	List of presets the PHY supports at 8 GT/s for half swing in addition to the 5 required by the base spec.

7.22 Control Signal Decode table – PCI Express Mode

The following table summarizes the encodings of four of the seven control signals that cause different behaviors depending on power state. For the other three signals, *Reset#* always overrides any other PHY activity. *TxCompliance* and *RxPolarity* are only valid, and therefore should only be asserted, when the PHY is in P0 and is actively transmitting. Note that these rules only apply to lanes that have not been 'turned off' as described in section 8 (Multi-lane PIPE).

PowerDown[1:0]	*TxDetectRx/ Loopback*	*TxElecIdle*	Description
P0: 00b	0	0	PHY is transmitting data. MAC is providing data bytes to be sent every clock cycle.
	0	1	PHY is not transmitting and is in electrical idle.
	1	0	PHY goes into loopback mode.
	1	1	Illegal. MAC should never do this.
P0s: 01b	Don't care	0	Illegal. MAC should always have PHY doing electrical idle while in P0s. PHY behavior is undefined if *TxElecIdle* is deasserted while in P0s or P1.
		1	PHY is not transmitting and is in electrical idle. Note that any data transferred across the PIPE interface before *TxElecIdle* is asserted, but not yet signaled on the analog interface is signaled before the analog interface becomes idle.
P1: 10b	Don't care	0	Illegal. MAC should always have PHY doing electrical idle while in P1. PHY behavior is undefined if *TxElecIdle* is deasserted while in P0s or P1.
	0	1	PHY is idle.
	1	1	PHY does a receiver detection operation.
P2: 11b	Don't care	0	PHY transmits Beacon signaling

Formatted: Normal

		1	PHY is idle.

7.23 Control Signal Decode table – USB SuperSpeed Mode

The following table summarizes the encodings of four of the seven control signals that cause different behaviors depending on power state. For the other three signals, *Reset#* always overrides any other PHY activity. *RxPolarity* is only valid, and therefore should only be asserted, when the PHY is in P0 and is actively transmitting.

PowerDown[1:0]	*TxDetectRx/ Loopback*	*TxElecIdle*	Description
P0: 00b	0	0	PHY is transmitting data. MAC is providing data bytes to be sent every clock cycle.
	0	1	PHY is not transmitting and is in electrical idle. Note that any data transferred across the PIPE interface before *TxElecIdle* is asserted, but not yet signaled on the analog interface is signaled before the analog interface becomes idle.
	1	0	PHY goes into loopback mode.
	1	1	PHY transmits LFPS signaling.
P1: 01b	Don't care	0	PHY transmits LFPS signaling
		1	PHY is not transmitting and is in electrical idle.
P2: 10b or P3: 11b	Don't care	0	PHY transmits LFPS signaling
	0	1	PHY is idle.
	1	1	PHY does a receiver detection operation.

7.24 Control Signal Decode table – SATA Mode

The following table summarizes the encodings of the control signals that cause different behaviors in POWER_STATE_0. For other control signals, *Reset#* always overrides any other PHY activity.

Note: The PHY transmit latency reported in section 7.20**.** must be consistent for all the different behaviors in POWER_STATE_0. This means that the amount of time OOB signaling is present on the analog TX pair must be the same as the time OOB signaling was indicated on the PIPE interface.

PowerDown[2:0]	*TxDetectRx/ Loopback*	*TxElecIdle*	Description
POWER_STATE_0: 00b	0	0	PHY is transmitting data. MAC is providing data bytes to be sent every clock cycle.
	0	1	PHY is not transmitting and is in electrical idle. Note that any data transferred across the PIPE interface before *TxElecIdle* is asserted, but not yet signaled on the analog interface is signaled before the analog interface becomes idle.
	1	0	PHY goes into loopback mode.

	1	1	PHY transmits OOB signaling with pattern determined by TX Pattern. Note that a PHY must ensure the transition between OOB signaling and data signaling is performed smoothly on a symbol boundary on the analog interface.
Power Stater other than POWER_STATE_0	Don't care	Don't care	PHY is not transmitting and is in electrical idle.
			PHY is not transmitting and is in electrical idle.

7.25 Required synchronous signal timings

To improve interoperability between MACs and PHYs from different vendors the following timings for synchronous signals are required:

Setup time for input signals	No greater than 25% of cycle time
Hold time for input signals	0ns
PCLK to data valid for outputs	No greater than 25% of cycle time

7.26 128b/130b Encoding and Block Synchronization

For every 128 bits that are moved across the PIPE TxData interface at the 8.0 GT/s rate the PHY must transmit 130 bits. The MAC must use the TxDataValid signal periodically to allow the PHY to transmit the built up backlog of data. For example – if the TxData bus is 16 bits wide and PCLK is 500 Mhz then every 8 blocks the MAC must deassert TxDataValid for one PCLK to allow the PHY to transmit the 16 bit backlog of built up data. The buffers used by the PHY to store TX data related to the 128/130b encoding rate mismatch must be empty when the PHY comes out of reset and must be empty whenever the PHY exits electrical idle (since TX buffers are flushed before entry to idle). The PHY must use RxDataValid in a similar fashion.
TxDataValid and RxDataValid must be de-asserted for one clock every N blocks when the PIPE interface is operating at 8 GT/s, where N is 4 for an 8 bit wide interface, 8 for a 16 bit wide interface, and 16 for a 32 bit wide interface. The MAC must first de-assert TxDataValid at the end of the Nth transmitted block following reset or exit from electrical idle. The PHY must first de-assert RxDataValid at the end of the Nth received block transmitted across the PIPE interface following reset or exit from electrical idle.
There are situations, such as upconfigure, when a MAC must start transmissions on idle lanes while some other lanes are already active. In any such situation the MAC must wait until the cycle after TxDataValid is de-asserted to allow the PHY to transmit the backlog of data due to 128b/130b to start transmissions on previously idle lanes.

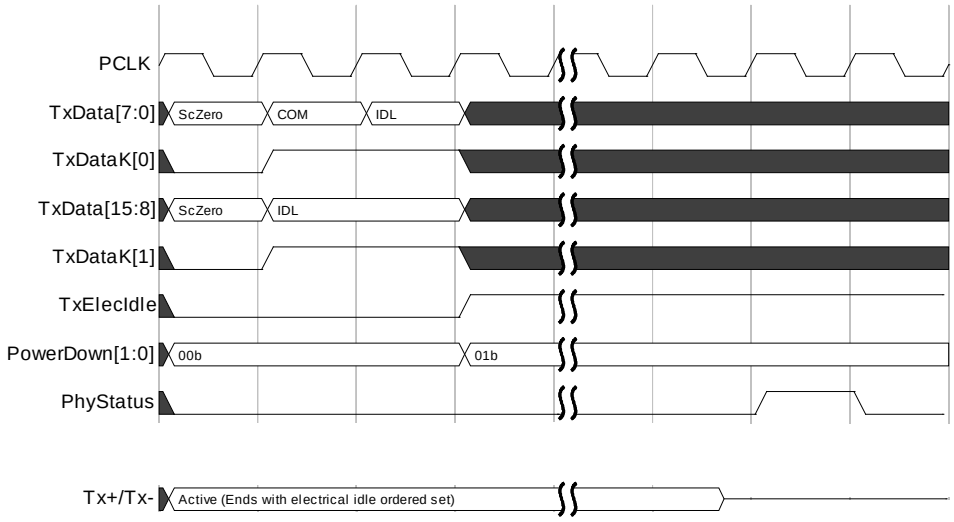
8 Sample Operational Sequences

These sections show sample timing sequences for some of the more common PCI Express, SATA and USB SuperSpeed operations. These are *sample* sequences and timings and are not required operation.

8.1 Active PM L0 to L0s and back to L0 – PCI Express Mode

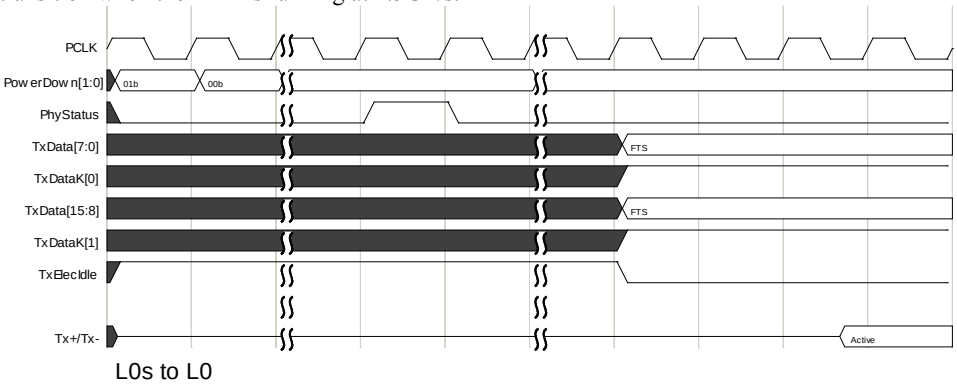
This example shows one way a PIPE PHY can be controlled to perform Active State Power Management on a link for the sequence of the link being in L0 state, transitioning to L0s state, and then transitioning back to L0 state.

When the MAC and higher levels have determined that the link should transition to L0s, the MAC transmits an electrical idle ordered set and then has the PHY transmitter go idle and enter P0s. Note that for a 16-bit or 32-bit interface, the MAC should always align the electrical idle on the parallel interface so that the COM symbol is in the low-order position (*TxDataK[7:0]*).



L0 to L0s

To cause the link to exit the L0s state, the MAC transitions the PHY from the P0s state to the P0 state, waits for the PHY to indicate that it is read to transmit (by the assertion of *PhyStatus*), and then begins transmitting Fast Training Sequences (FTS). Note, this is an example of L0s to L0 transition when the PHY is running at 2.5GT/s.

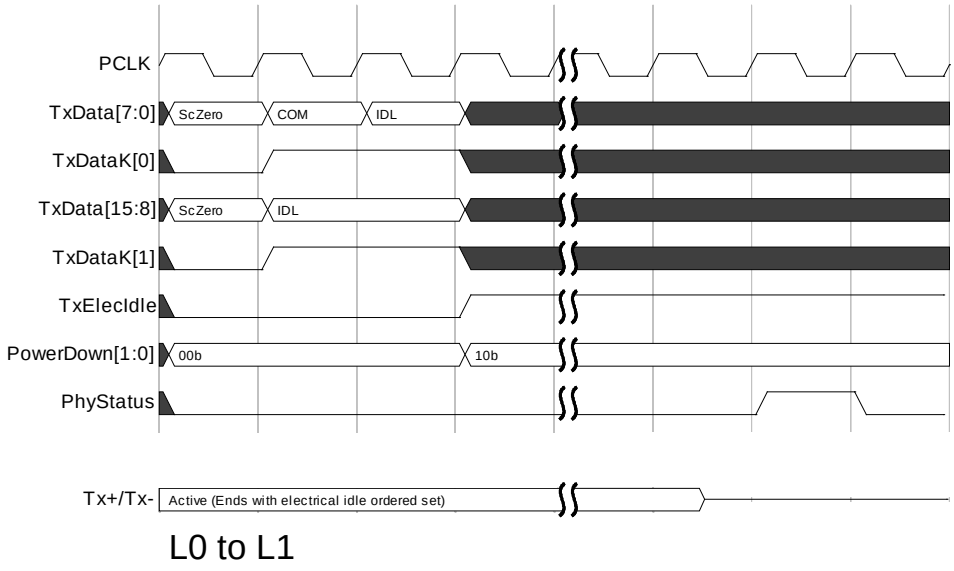


L0s to L0

8.2 Active PM to L1 and back to L0 - – PCI Express Mode

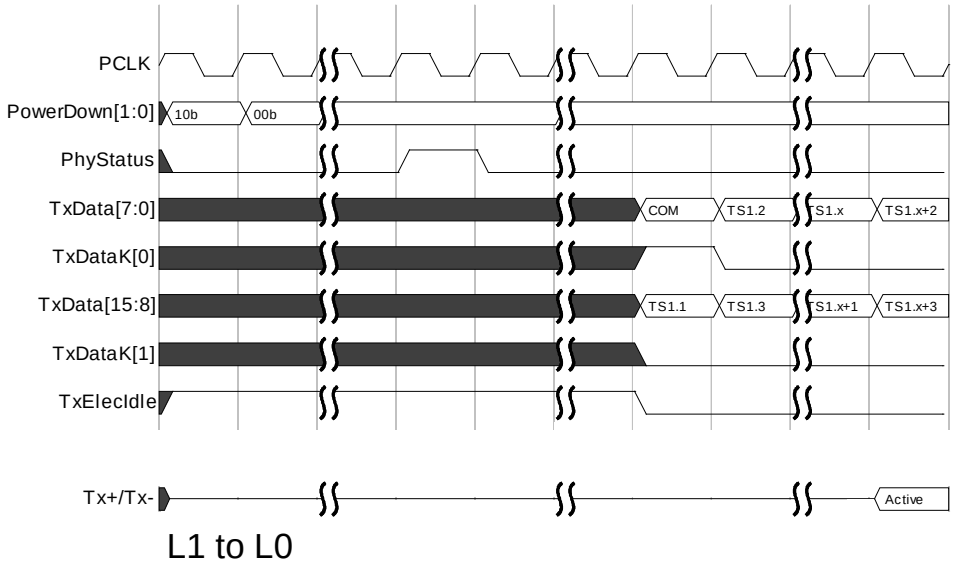
This example shows one way a PIPE PHY can be controlled to perform Active State Power Management on a link for the sequence of the link being in L0 state, transitioning to L1 state, and then transitioning back to L0 state. This example assumes that the PHY is on an endpoint (i.e. it is facing upstream) and that the endpoint has met all the requirements (as specified in the base spec) for entering L1.

After the MAC has had the PHY send PM_Active_State_Request_L1 messages, and has received the PM_Request_ACK message from the upstream port, it then transmits an electrical idle ordered set, and has the PHY transmitter go idle and enter P1.



L0 to L1

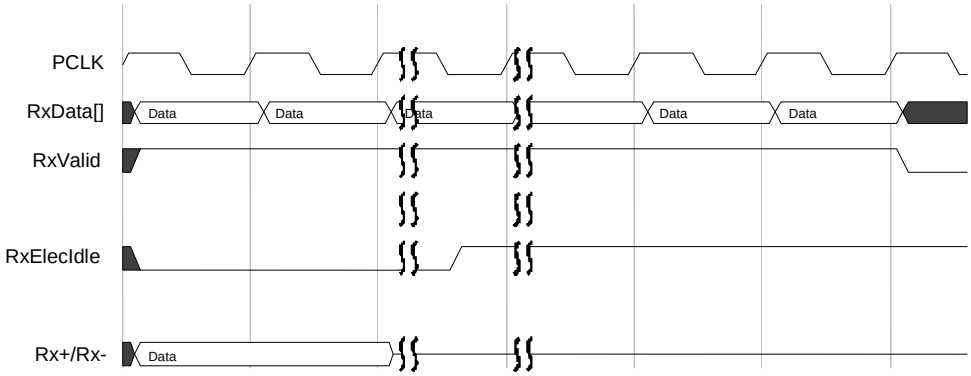
To cause the link to exit the 1 state, the MAC transitions the PHY from the P1 state to the P0 state, waits for the PHY to indicate that it is ready to transmit (by the assertion of *PhyStatus*), and then begins transmitting training sequence ordered sets (TS1s). Note, this is an example when the PHY is running at 2.5GT/s.



8.3 Receivers and Electrical Idle – PCI Express Mode Example

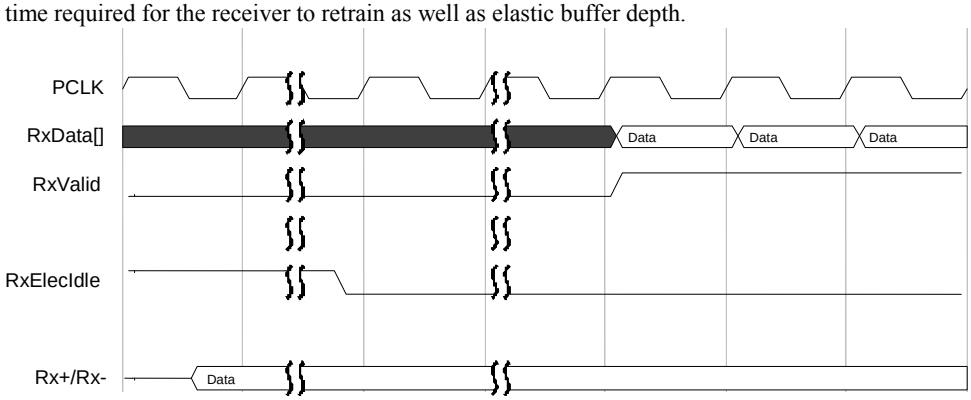
This section only applies to a PHY operating to 2.5GT/s. Note that when operating at 5.0 GT/s or 8 GT/s signaling rates, *RxElecIdle* may not be reliable. MACs should refer to the PCI Express Revision 3.0 Base Specification or USB 3.0 Specification for methods of detecting entry into the electrical idle condition. Refer to Table 5-4 for the definition of *RxElecIdle* when operating at 5.0 GT/s. This section shows some examples of how PIPE interface signaling may happen as a receiver transitions from active to electrical idle and back again. In these transitions there may be a significant time difference between when *RxElecIdle* transitions and when *RxValid* transitions.

The first diagram shows how the interface responds when the receive channel has been active and then goes to electrical idle. In this case, the delay between *RxElecIdle* being asserted and *RxValid* being deasserted is directly related to the depth of the implementations elastic buffer and symbol synchronization logic. Note that the transmitter that is going to electrical idle may transmit garbage data and this data will show up on the *RxData[]* lines. The MAC should discard any symbols received after the electrical idle ordered-set until RxValid is deasserted.



Receiver Active to Idle

The second diagram shows how the interface responds when the receive channel has been idle and then begins signaling again. In this case, there can be significant delay between the deassertion of *RxElecIdle* (indicating that there is activity on the *Rx+/Rx-* lines) and *RxValid* being asserted (indicating valid data on the *RxData[]* signals). This delay is composed of the time required for the receiver to retrain as well as elastic buffer depth.



Receiver Idle to Active

8.4 Using CLKREQ# with PIPE – PCI Express Mode

CLKREQ# is used in some implementations by the downstream device to cause the upstream device to stop signaling on REFCLK. When REFCLK is stopped, this will typically cause the CLK input to the PIPE PHY to stop as well. The PCI Express CEM spec allows the downstream device to stop REFCLK when the link is in either L1 or L2 states. For implementations that use CLKREQ# to further manage power consumption, PIPE compliant PHYs can be used as follows:

The general usage model is that to stop REFCLK the MAC puts the PHY into the P2 power state, then deasserts CLKREQ#. To get the REFCLK going again, the MAC asserts CLKREQ#, and then after some PHY and implementation specific time, the PHY is ready to use again.

CLKREQ# in L1

If the MAC is moving the link to the L1 state and intends to deassert CLKREQ# to stop REFCLK, then the MAC follows the proper sequence to get the link to L1, but instead of finishing by transitioning the PHY to P1, the MAC transition the PHY to P2. Then the MAC deasserts CLKREQ#.

When the MAC wants to get the link alive again, it can:

- Assert CLKREQ#
- Wait for REFCLK to be stable (implementation specific)
- Wait for the PHY to be ready (PHY specific)
- Transition the PHY to P0 state and begin training.

CLKREQ# in L2

If the MAC is moving the link to the L1 state and intends to deassert CLKREQ# to stop REFCLK, then the MAC follows the proper sequence to get the link to L2. Then the MAC deasserts CLKREQ#.

When the MAC wants to get the link alive again, it can:

- Assert CLKREQ#
- Wait for REFCLK to be stable (implementation specific)
- Wait for the PHY to be ready (PHY specific)
- Transition the PHY to P0 state and begin training.

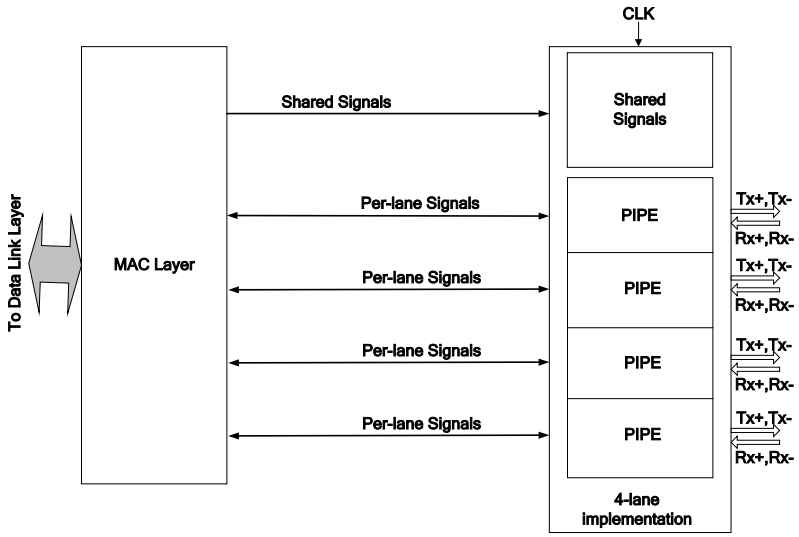
Delayed CLKREQ# in L1

The MAC may want to stop REFCLK after the link has been in L1 and idle for awhile. In this case, the PHY is in the P1 state and the MAC must transition the PHY into the P0 state, and then the P2 state before deasserting CLKREQ#. Getting the link operational again is the same as the preceding cases.

9 Multi-lane PIPE – PCI Express Mode

This section describes a suggested method for combining multiple PIPEs together to form a multi-lane implementation. It describes which PIPE signals can be shared between each PIPE of a multi-lane implementation, and which signals should be unique for each PIPE. There are two types of PHYs. "Variable" PHYs that are designed to support multiple links of variable maximum widths and "Fixed" PHYs that are designed to support a fixed number of links with fixed maximum widths.

The figure shows an example 4-lane implementation of a multilane PIPE solution. The signals that can be shared are shown in the figure as "Shared Signals" while signals that must be replicated for each lane are shown as 'Per-lane signals'.



4-lane PIPE implementation

The MAC layer is responsible for handling lane-to-lane deskew and it may be necessary to use the per-lane signaling of SKP insertion/removal to help perform this function.

	Shared Signals	Per-Lane Signals or Shared Signals	Per-lane Signals
	CLK Max PCLK	EncodeDecodeBypass BlockAlignControl FS[5:0] LF[5:0] TxSwing TxMargin[2:0] TxDetectRx/Loopback Rate Width[1:0] PCLK Rate[2:0] Reset#	TxData[], TxDataK[] RxData[], RxDataK[] TxStartBlock TxElecIdle TxCompliance RxPolarity RxValid RxElecIdle RxStatus[2:0] RxDataValid RxStartBlock TxDeemph[17:0]

		TxDataValid PCLK	PowerDown[1:0] PhyStatus RxPresetHint[2:0] --- RxEqEval LinkEvaluationFeedbackFigureMerit[7:0] LinkEvaluationFeedbackDirectionChange[7:0] InvalidRequest TxSyncHeader[1:0] RxSyncHeader[1:0] RxStandby RxStandbyStatus

A MAC must use all "Per-Lane Signals or Shared Signals" that are inputs to the PHY consistently on all lanes in the link. A PHY must ensure that PCLK and Max PCLK are synchronized across all lanes in the link. It is recommended that a MAC be designed to support both PHYs that implement all signals per lane and those that implement the "Per-Lane or Shared Signals" per link. A "Variable" PHY must implement the signals in "Per-Lane Signals or Shared Signals" per lane. A "Fixed" PHY may implement the signals in "Per-Lane Signals or Shared Signals" as either Shared or Per-Lane. A "Fixed" PHY should implement all the signals in "Per-Lane Signals or Shared Signals" consistently as either Shared or Per-Lane.

In cases where a multi-lane has been 'trained' to a state where not all lanes are in use (like a x4 implementation operating in x1 mode), a special signaling combination is defined to 'turn off' the unused lanes allowing them to conserve as much power as the implementation allows. This special 'turn off' signaling is done using the *TxElecIdle* and *TxCompliance* signals. When both are asserted, that PHY can immediately be considered 'turned off' and can take whatever power saving measures are appropriate. The PHY ignores any other signaling from the MAC (with the exception of *Reset#* assertion) while it is 'turned off'. Similarly, the MAC should ignore any signaling from the PHY when the PHY is 'turned off'. There is no 'handshake' back to the MAC to indicate that the PHY has reached a 'turned off' state.

There are two normal cases when a lane can get turned off:

1. During LTSSM Detect state, the MAC discovers that there is no receiver present and will 'turn off' the lane.
2. During LTSSM Configuration state (specifically Configuration.Complete), the MAC will 'turn off' any lanes that didn't become part of the configured link.

As an example, both of these cases could occur when a x4 device is plugged into a x8 slot. The upstream device (the one with the x8 port) will not discover receiver terminations on four of its lanes so it will turn them off. Training will occur on the remaining 4 lanes, and let's suppose that the x8 device cannot operate in x4 mode, so the link configuration process will end up settling on x1 operation for the link. Then both the upstream and downstream devices will 'turn off' all but the one lane configured in the link.

When the MAC wants to get 'turned off' lanes back into an operational state, there are two cases that need to be considered:

1. If the MAC wants to reset the multi-lane PIPE, it asserts *Reset#* and drives other interface signals to their proper states for reset (see section 6.2). Note that this stops signaling 'turned off' to all lanes because *TxCompliance* is deasserted during reset. The multi-lane PHY asserts *PhyStatus* in response to *Reset#* being asserted, and will deassert *PhyStatus* when *PCLK* is stable.

2. When normal operation on the active lanes causes those lanes to transition to the LTSSM Detect state, then the MAC sets the *PowerDown[1:0]* signals to the P1 PHY power state at the same time that it deasserts 'turned off' signaling to the inactive lanes. Then as with normal transitions to the P1 state, the multi-lane PHY will assert *PhyStatus* for one clock when all internal PHYs are in the P1 state and *PCLK* is stable.

