OPEN

Compute Project

This document to be released under the following OCP copyright release:

COPYRIGHT LICENSE AGREEMENT

This Agreement ("Agreement") is entered into on the date set forth below, (the "Effective Date") by and between the Open Compute Project Foundation a Delaware corporation ("OCP") and the entity identified below ("Licensor").

WHEREAS, Licensor is the owner of and/or has certain rights in or to the works of authorship identified in the attached Exhibits (collectively, the "Work").

WHEREAS, OCP desires to copy, distribute, make derivative works of and publish the Work and derivate works thereof, including without limitation in one or more OCP publications and/or on OCP's website, and Licensor will benefit from OCP's use of the Work as described in this Agreement.

NOW THEREFORE, in consideration of the promises in this Agreement, the parties agree as follows:

1. Structure of the Agreement. There may be multiple Exhibits to this Agreement. Each Exhibit will be signed by an authorized representative and will be governed by and subject to the terms set forth in this Agreement, with the licenses applicable to the Work described therein effective as of the date the Exhibit is signed.

2. License. Licensor hereby grants to OCP a non-exclusive, transferable (in accordance with Section 7 below), royalty free, fully-paid, perpetual, irrevocable, worldwide license, under Licensor's copyrights in the Work, with the right to sublicense, to use, reproduce, create derivative works, distribute, and publicly display and perform the Work and derivative works thereof, in whole or in part, as a separate work or as part of a collective work. The foregoing will apply to all mediums now known or hereafter existing.

3. Ownership of the Work / Other Rights Reserved. Except for the foregoing license, as between OCP and Licensor, Licensor retains all right, title and interest in and to the Work and all intellectual property rights therein. Licensor hereby reserves all rights not expressly granted in this Agreement. No additional licenses or rights whatsoever (including without limitation any patent licenses) are granted by implication, exhaustion, estoppel or otherwise.

4. Representations and Indemnification. Licensor represents to OCP that: (i) Licensor is the sole and exclusive owner of the Work and all copyrights therein or Licensor has the right and authority to grant the licenses set forth in this Agreement and (ii) OCP's exercise of the licenses set forth in this Agreement will not result in any infringement of any third party's copyrights or the misappropriation of any third party's trade secrets. Licensor agrees to indemnify and hold OCP harmless from and against any losses, damages, liabilities, settlement amount, costs and expenses (including reasonable attorneys' fees) incurred by OCP in connection with any breach of the foregoing representations. This Section will survive the termination of this Agreement.

5. Term and Termination. This Agreement will commence on the Effective Date and will terminate upon the written agreement of the parties or by written notice by OCP.

6. Governing Law and Forum. This Agreement shall be solely and exclusively governed, construed and enforced in accordance with the laws of the Texas, USA, without reference to conflict of laws principles. Any suit, action or proceeding arising from or relating to this

Agreement must be brought, solely and exclusively, in courts located in Travis County, Texas and each party irrevocably consents to the jurisdiction and venue of any such court.

7. Assignment. OCP may assign this Agreement (a) with the consent of Licensor, not to be unreasonably withheld or delayed, or (b) upon notice, but without such consent, in connection with a merger, acquisition, change of control, or sale of substantially all the assets of OCP. This Agreement shall be binding upon and inure to the benefit of the parties and their successors and permitted assigns.

8. Mutual Limits on Liability. Except as set forth below, in no event shall either party be liable to the other party in any manner, under any theory of liability, whether in contract, tort (including negligence), or other theory, for any indirect, consequential, incidental, exemplary, punitive, statutory or special damages, including lost profits, regardless of whether such party was advised of or was aware of the possibility of such damages. Except as set forth below, in no event shall the total, cumulative liability of either party regarding any and all claims and causes of action, under any theory of liability, whether in contract, tort (including negligence), or otherwise, exceed One Thousand Dollars ($1,000). The limitations set forth in this Section will not apply to liability arising under Section 4 (Representations and Indemnification) above. This Section will survive termination of this Agreement.

9. Entire Agreement. This Agreement constitutes the entire agreement between the parties with respect to its subject matter and it supersedes all prior or contemporaneous oral or written agreements and representations concerning the subject matter herein. This Agreement may be amended only in a written document signed by both parties. This Agreement shall not be interpreted or construed against the party preparing it.

10. Counterparts and Facsimile Signatures. This Agreement may be executed in counterparts all of which taken together shall constitute one single agreement between the parties. A facsimile transmission of the executed signature page of this Agreement shall constitute due and proper execution of this Agreement by the applicable party.

Contents

Bunch of Wires Specification

The Open Domain-Specific Architecture BoW Workstream

Mark Kuemerle, Ramin Farjad, Ken Poulton, Suresh Subramaniam, Bapi Vinnakota

1. Introduction

The Bunch of Wires (BoW) is a very simple, open and interoperable physical interface between two chiplets or chip-scale-packages (CSP) on a common package. This document specifies the BoW interface.

1.1. Objectives

The BoW interface is a set of backward compatible die-to-die parallel interfaces that provides the flexibility to trade off throughput/chipedge for, design complexity, cost, packaging technology. Backward as defined by every new mode being compatible with the "Basic mode" defined in this specification. This document specifies a "Terminated mode" in addition to the Basic mode. It is also possible that future technology enhancements will enable modes that further increase throughput per chipedge. Examples of this include previous discussions on a Turbo mode using simultaneous bidirectional communication (not covered in this document). Each future version of this specification is expected to be compatible with at least two previous significant versions.

The use of the BoW is expected to be confined to connect die placed closed to one another within the same package. Signal loss in transmission is expected to be lower than for off-package protocols. The definition of the BoW interface aims to meet the following design objectives:

- Inexpensive to implement
- Portable across process nodes ranging from 28nm to 5nm
- Portable across multiple bump pitches
- Have the flexibility to support advancing packaging technologies
- Be unencumbered by technology license costs
- Be very low power (< 1 pJ/bit) as defined by Tx IO Pad, wire and Rx IO Pad.
- Offer very low latency: <5 ns without FEC, <15 ns with FEC. Latency as defined from the PCS parallel interface at the source, through Tx interface, channel, Rx interface received at the PCS parallel interface at the receiver. Based on experience, the 5 ns target meets the latency requirements of high-performance applications and has been demonstrated to be achievable.
- Offer the following unidirectional throughput/chipedge target range:
  - Minimum Targets
    - 100 Gbps/mm with all packaging options
    - 1 Tbps/mm with advanced packaging options
    - Enable higher throughput at very short reach < 1 mm.

  - As reference examples, with a data rate of 4 Gbps/wire
    * achieve 200+ Gbps/mm with a bump pitch of 130um and with a die edge stack depth no greater than 2 routing layers with organic laminate packaging.
    * achieve 1+ Tbps/mm with a bump pitch of 50 um and with a die edge stack depth no greater than 4 routing layers with advanced packaging.

1.2. Advantages

The Bunch of Wires interface provides several key advantages for chiplet-based systems:

- Can operate at higher data rates per pin than existing parallel standards
  - -or- lower data rates for compatibility with existing parallel standards
- Can be implemented in legacy technologies (process nodes) with generally available IP
- Terminated mode can be implemented in less effort than a traditional SERDES
- Does not require silicon-based interconnect
- Is not constrained or intended to be used with a specific bump pitch
  - Two BoW interfaces can each be implemented at different bump pitches and can be directly connected on an organic substrate, through fanout technology or through silicon based interconnect.

While the advantages and simplicity are excellent benefits, the BoW interface does require more package routing traces than other serial based XSR or USR interconnect. This drives BoW implementations that need the highest bandwidth to use fine bump pitches and 'stacked' BoW implementations, adding some complexity and cost in test and packaging. Lower bandwidth implementations are free to use more standard packaging technology with coarse bump pitch.

1.3. Scope

The scope of this document and of any contributions to this document are limited to:

1. The specification of the BoW interface that specifies the following functionality:
   a. Operating modes
   b. Physical design
   c. Recommended bump patterns and signal ordering
   d. Initialization, operation and management
   e. Methods to verify and validate compliance with this specification
   f. Performance estimates
   g. Functions or design practices that may be necessary to meet the design objectives
   h. Test, testability and test enablement

2. The following activities are outside the scope of this document and contributions to this document:
   a. Physical implementations of the interface

b. Integration of the interface with system-level data flow e.g. adapting a standard PHY-layer abstraction such as PIPE interface to the BoW
c. The actual use of this interface in systems
d. The use of this interface outside a package

3. The following activities are intended to be addressed in subsequent versions of this specification:

    a. Security

1.4. Language

- "Shall" or "must" indicates a requirement. Failure to meet the requirement results in non-compliance
- "Should" indicates a strong suggestion, but not a requirement. Failure to implement the suggestion does not result in non-compliance.
- "May" indicates an option.
- The lack of one of the above verbs indicates the material is informative.
- "Reference" indicates a reference design that is provided as example for explanation, but is not a requirement.

1.5. Compliance Summary

The specifications must be met over process variation, supply voltage range and temperature range (PVT). Each implementation must document its supported supply voltage range and temperature range.

Table XX will summarize the compliance points that shall be met in order to meet the BoW requirements. Each of the compliance points is discussed in the specification.

This section will also include a discussion on interoperability?.

to be completed in subsequent draft

2. BoW Overview

Chiplet-based designs require physical and logical connectivity between the die in a single package. They also require the use of more complex packaging to accommodate the multiple die sharing a package. This section provides an overview of the BoW physical interface, its use in a multi-chiplet design, and how with the Open Domain-Specific Architecture stack it can be used as an underlay for popular transaction protocols.

2.1. BoW Application

The BoW is intended to be an energy-efficient easy-to-use PHY interface between a pair of die as shown in Figure 1. The BoW

The BoW is intended to be a low-power die-to-die interface in a chiplet-based package. As shown in Figure [#fig-bow_overviews], the BoW PHYs between two wires are physically connected through wires on a substrate or interposer. A BoW PHY will not have enough drive strength for off-package interfaces. A BoW PHY is also not expected to be used for buses that are entirely on

die. *Provides physical digital connectivity between the die* Can be used with multiple packaging technologies including organic laminate, interposers, bridges and RDL * Requires a separate protocol for the logical transport of information between the die

interposer in a dotted line extend the package out a bit - or show it as a thicker line

The basic BoW PHY unit is a 16-bit slice. A BoW PHY slice transmits and receives 16 bits of data between die. Figure 2 details the wires between BoW PHY instances on connected die in one direction. Since the BoW is a source-synchronous PHY, the transmitting PHY transmits a CLK signal with the data. A BoW PHY optionally has two additional wires that may be used for data or functions to improve data correctness - Forward Error Correction (FEC) and Data Bus Inversion (DBI). The connectivity between die is specified in detail below in Section 3

Bapi: Need Figure 2 to have Tx and Rx.

A BoW slice consists of either a transmit (Tx) or a receive (Rx) component. In Figure 3, the Tx and Rx paths each use a separate set of bumps at the die edge and wires between the connected die. On die, the BoW datapath is a mixture of analog and digital circuitry that transports. Each BoW slice consists of a bits from a link layer to the die edge. On one side, the on-die BoW datapath terminates in bumps, at the edge of the die connected to wires on the substrate. Internally the datapath is connected to a MAC. The datapath also expected to include serializer (deserializer) logic on transmit (receive) that is not specified in this document.

The BoW control path stores information to initialize and operate the datapath in status registers. The control path outputs information to manage the datapath. The interface between the control and datapath is specific to an implementation. The BoW PHY may be implemented with the control and datapath physically commingled. Implementing them separately allows the control logic to be implemented in a separate block for multi-slice macros. The control path may also receive input from an external digital control and management bus such as I2C to coordinate interface operation with other components in the package.

Any BoW implementation must receive a clock input from a PLL, which may be shared across multiple slices. A BoW Rx slice is expected to forward the clock received from the far side to the core logic. A BoW Tx slice is expected to forward its PLL clock to the die's core logic. This will place all clock domain transitions in the core logic.

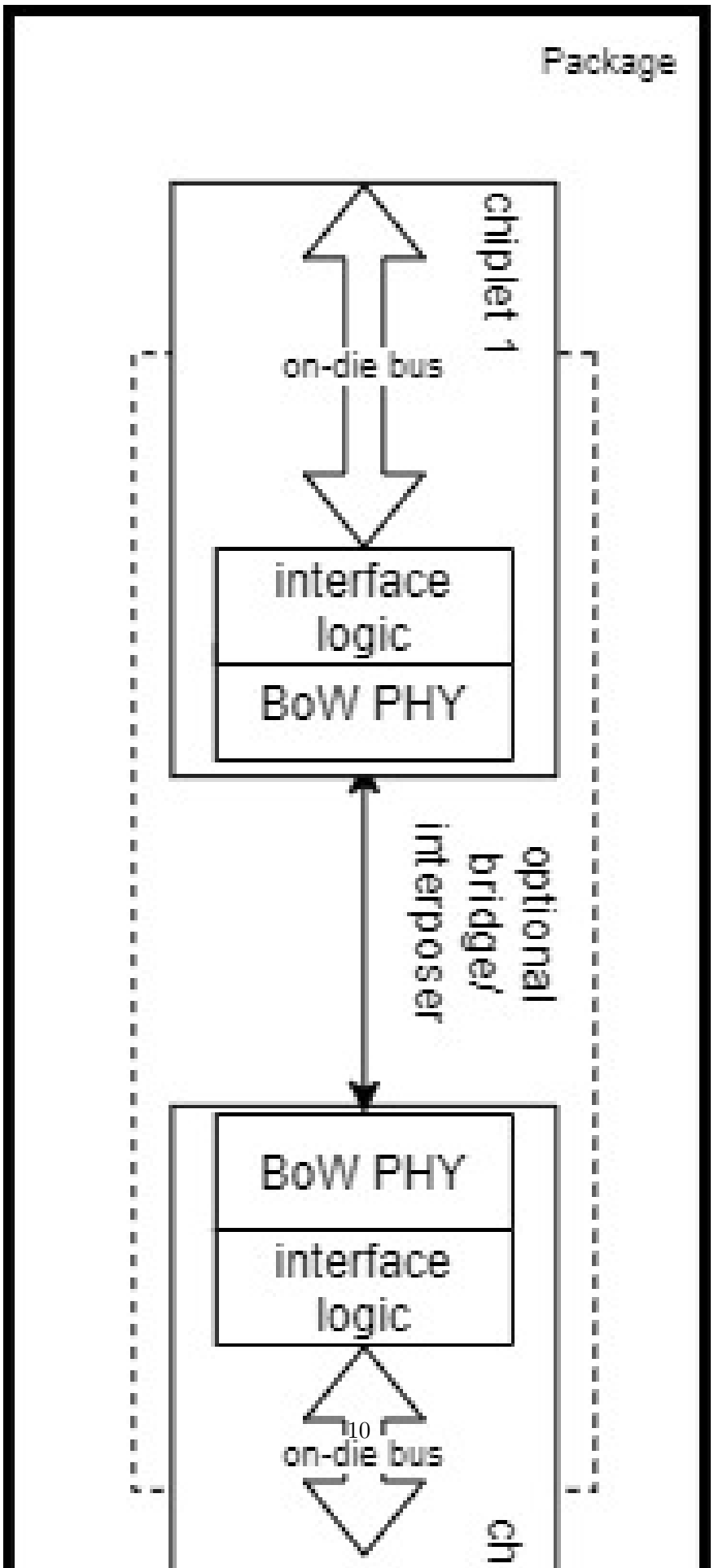
2.2. BoW in the Package

Within the package, the BoW datapath is transported on physical passive wires between the pair of connected die. The specifics of the wires, such as their density, maximum length, impedence characteristics and how they are realized vary with the packaging technology.

2.3. BoW for Common Transaction Protocols

Two connected die in a multi-chiplet device need to exchange logical information. The ODSA aims to define an open physical and logical interface for chiplets, as shown in Figure 4 to enable chiplets from multiple vendors to interoperate and be integrated in a multi-die package. The Bunch of Wires is an open D2D PHY option in the interface. The logical component of the ODSA interface aims to

support protocols used commonly for the two most common chiplet use cases, package aggregation and die disaggregation across a wide range of open and proprietary D2D PHYs such as PCIe, CXL, CCIX, AXI and proprietary streaming protocols.


Package
chiplet 1
on-die bus
interface logic
BoW PHY
optional bridge/ interposer
BoW PHY
interface logic
on-die bus
ch

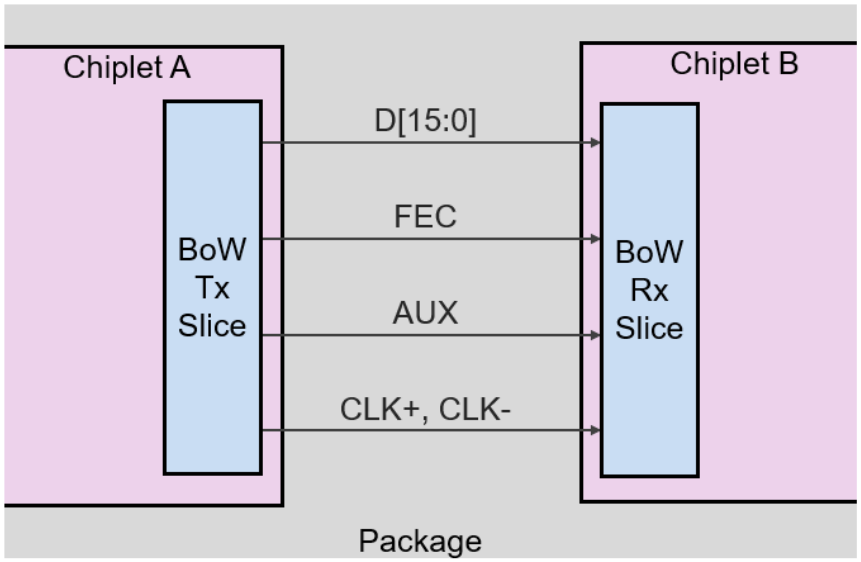


Figure 2. BoW Single Slice Signals

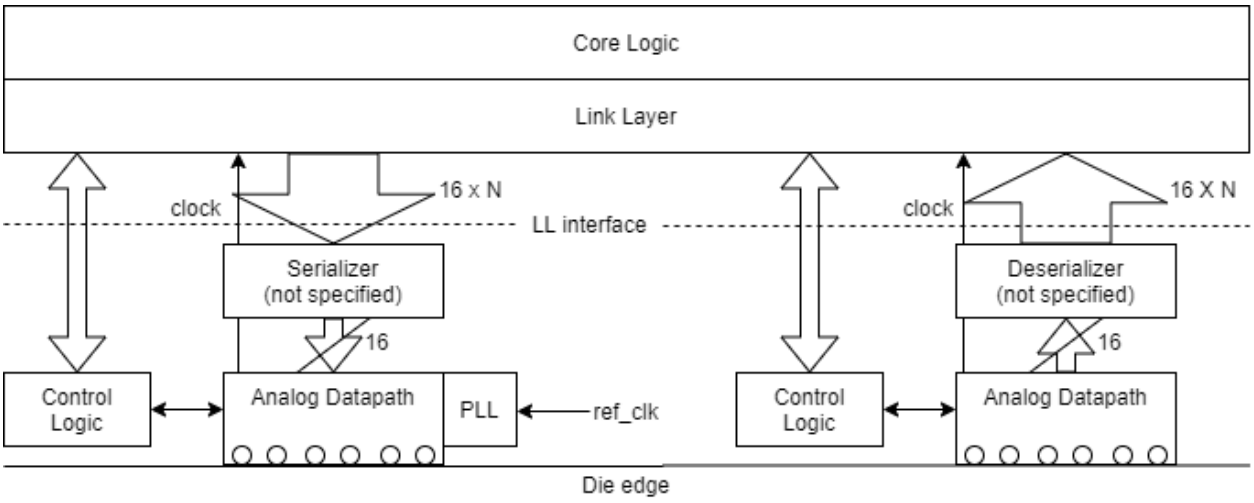


Figure 3. BoW connection to core logic

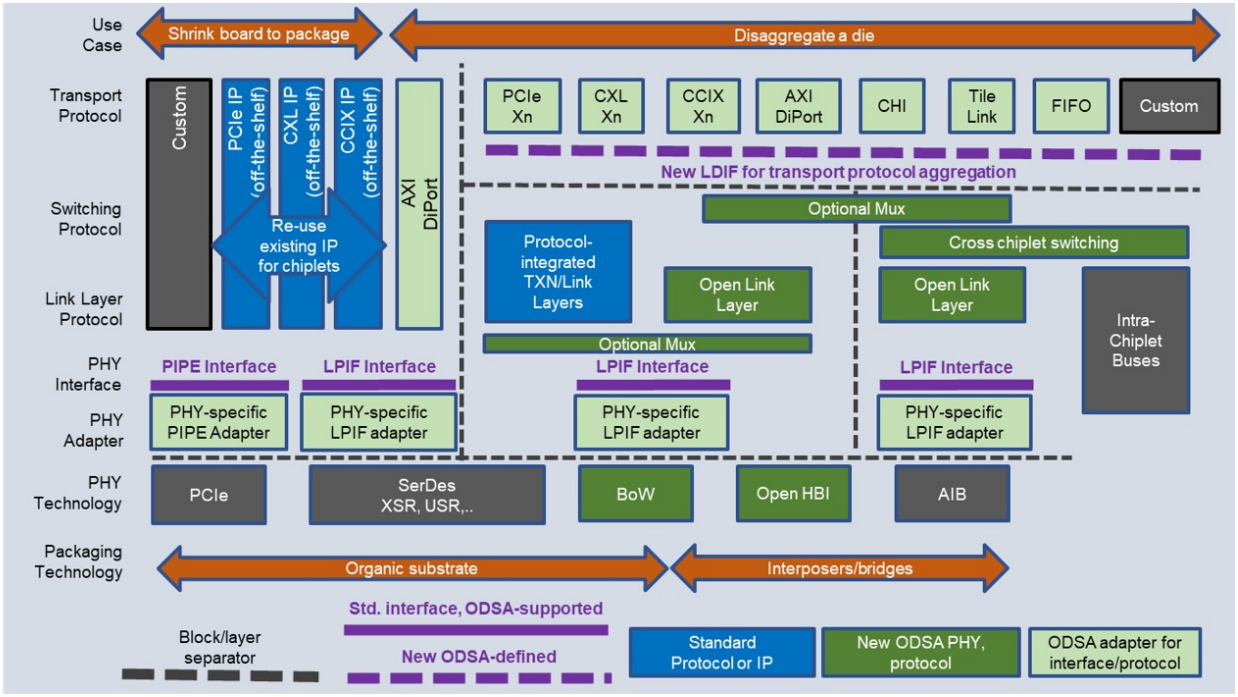


Figure 4. The BoW PHY in the ODSA Stack

The ODSA stack abstracts the PHY layer from the logical interface by using well-defined abstraction interfaces. Specifically, the ODSA will use two abstraction interfaces, the PIPE and LPIF interfaces. Any logic transaction controller, say a PCIe controller for example, that supports a PIPE (LPIF) interface can use any D2D PHY that also supports a PIPE (LPIF) interface as a physical layer to transport protocol bits between die. As shown in Figure 4, the BoW interface may need to receive data through either the PIPE or LPIF interfaces to support common transaction protocols. For this use case, the BoW interface will have to be augmented with an interface-specific adapter to support a protocol controller. The serializer/de-serializer functionality may also be integrated with the adapter. The specifications for each adapter are outside the scope of this document. Figure 5 shows how the BoW with an example PIPE adapter can be an underlay to transport PCIe transactions.

2.4. BoW Modes

BoW has been developed and specified such that multiple modes of operation can be defined and supported. All modes must be interoperable with the minimum definition. All modes are source synchronous parallel interfaces using a differential clock that may operate at multiple.

The range of operating frequencies for a mode must be documented in a data sheet. frequencies. Beyond the basic mode, adding termination provides higher performance per mm of beachfront bandwidth but is more complex to design. It may also be possible to get higher performance by using advanced packaging.

The BoW specification provides for optional technology to increase the data rate per wire, to enable a graceful cost-performance tradeoff. It is also possible for basic BoW configuration implemented with advanced packaging or between physically close chiplets to offer a higher data

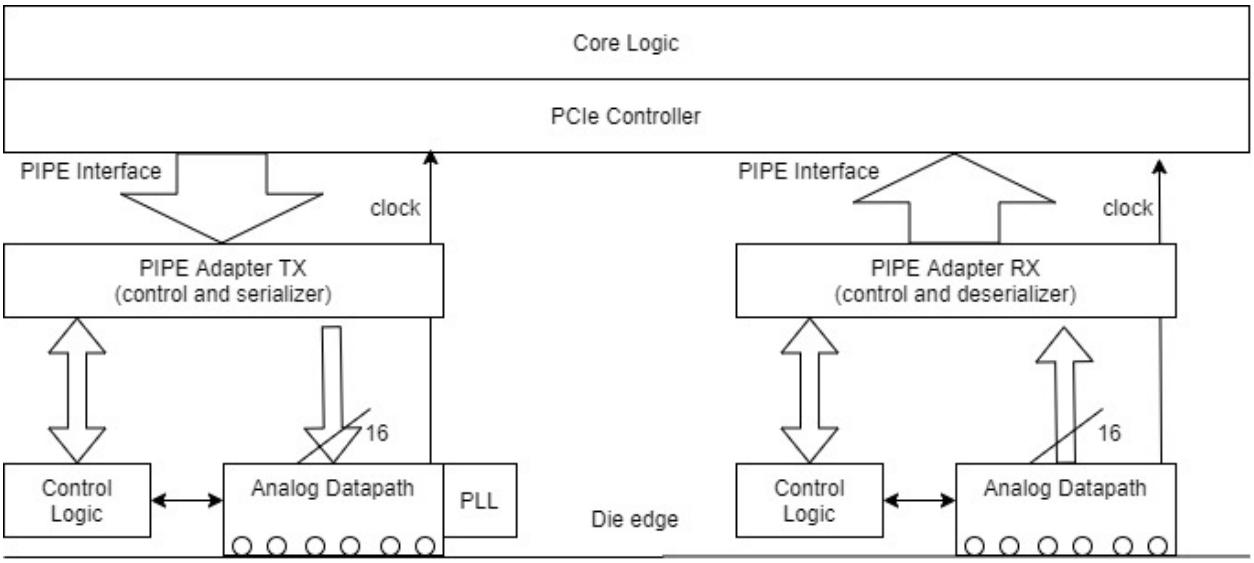


Figure 5. BoW with a PIPE adapter for PCIe transactions

rate per wire than a terminated BoW interface in some configurations. A high level view of the BoW Interface Data-rate ranges is shown below:

Bapi: Do we want all modes to be compatible with BoW Basic or BoW interop? I think we want every BoW implementation to support the BoW Basic Mode? How does BoW BiDi become compatible with that? It's a bit late, but configuration seems a more appropriate name than mode.

[bowspec_figtradeoff]

All modes must be interoperable with the minimum definition. The various BoW modes are specified such that they can be directly connected to one another. **When two interfaces are connected, data rate for the operating mode must be configured such that both ends support the same configuration at the same data rate.** All BoW implementations must support the minimum configuration of BoW (2Gbps datarate, 1 Ghz clock rate, un-terminated IO).

3. Signal Definitions

Figures. 6 - 7 provide a high level overview of the typical components that make up a Tx slice. Figures. 8 - 9 do the same for the Rx slice. The scope of this specification is limited to the inputs and outputs, as well as the components shown in the light blue box. In this section, we cover the signal definitions for the external 3.1 and link layer 3.2 facing interfaces. In Sections 8,9,10, the details pertaining to reset, initialization, configuration, and the calibration state machine are specified. In Section 11, a complete register definition for control and status signals is provided. Section 12.2 has details on testability. In Figures. 6 - 9, the datapath shown is only for illustrative purposes, specific implementations are left to the designer. The BoW PHY can accommodate FIFO implementations that are internal or external to the PHY. Examples for the Tx slice are shown in Figures 6 - 7 and for Rx slice in Figures 8 - 9. However, the Serializer/Deserializer fucntions with the appropriate gear ratios shall be implemented in the BoW PHY.

Function	# Signals	Signal Name	Notes
Clock	2	CLK+, CLK-	Differential
Data	16	D0-15	
Forward Error Correction	1	FEC	Optional
Auxilliary	1	AUX	Optional

Table 1. BoW Signals

3.1. External Facing Signals

As shown in Figures. 6 - 9, each BoW slice consists of a differential clock pair , 16 single-ended data wires, and optional FEC and AUX wires. Each BoW slice is unidirectional when in operation. A chiplet may be designed with with Rx-only and Tx-only slices, or each slice may have both Tx and Rx capability which is configured at runtime. A bidirectional link is composed of some number of slices configured for Rx and some for Tx.

FEC (Forward Error Correction) is an optional signal that allows using FEC to improve the bit error rate (BER). AUX is an optional signal that can be used for purposes such as DBI, flow control, redundancy, etc. Chiplets A and B will need to agree on the details on FEC and AUX usage.

Table 9 below summarizes these signals.

3.2. Link Layer Interface Signals

Each Tx or Rx slice shall interface to the link layer with a datapath width ranging from 32 bits to 512 bits. The corresponding pclk (tx, or rx) shall have a frequency (datawidth*pclk) that matches the overall throughput on the output wires of the slice (16* 0.5*line_rate). This allows for a wide range of core logic implementations and gear ratios by trading off latency. The serializer/deserializer functions shall be implemented in the PHY, as needed. However, the specification is flexible on the location of the FIFO (inside or outside the PHY) without compromising interoperability. Example datapath implementations of the FIFO inside are shown in Figs 6, 8. FIFO outside examples are shown in Figs 7, 9. More details on the Link Layer interface signals can be found in Section 3.4.

3.3. Bow Physical Interface

The BoW interface comprises either a transmit or receive slice with three signal types:

- Data signals
  - Inputs (RX): data input signals to the interface
  - Outputs (TX): data output signals from the interface
- Optional signals
  - AUX/Data bus inversion out (tx_AUX_DBI), sent from the interface
  - Forward Error Correction out (tx_FEC), sent from the interface
  - AUX/Data bus inversion in (rx_AUX_DBI), input to the interface
  - Forward Error Correction in (rx_FEC), input to the interface

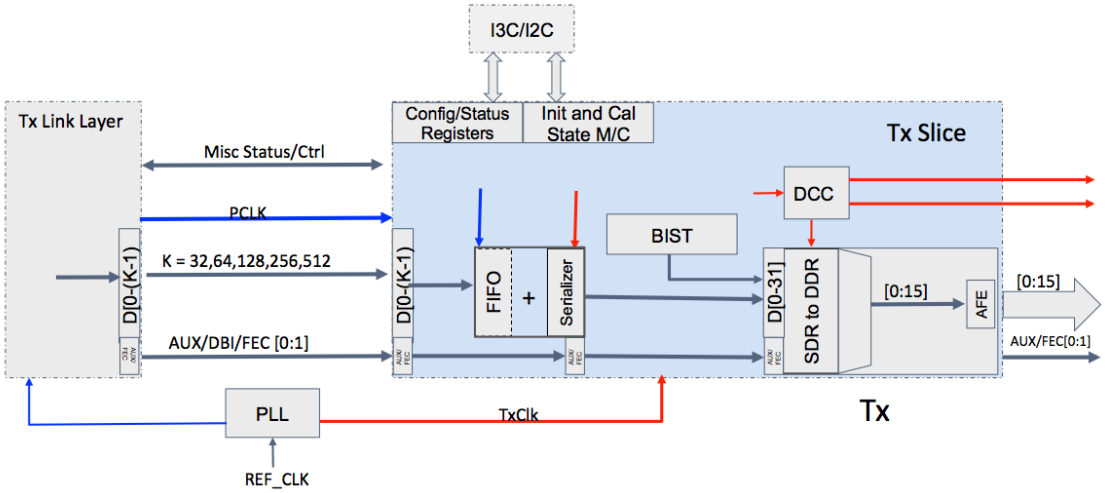


Figure 6. BoW Link Layer and Tx Slice with FIFO

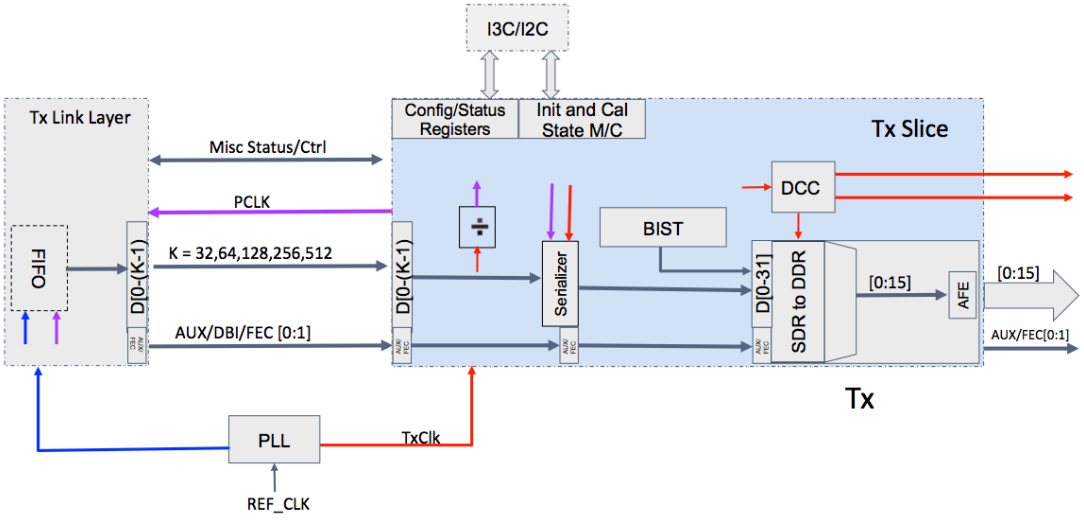


Figure 7. BoW Link Layer and Tx Slice with no FIFO

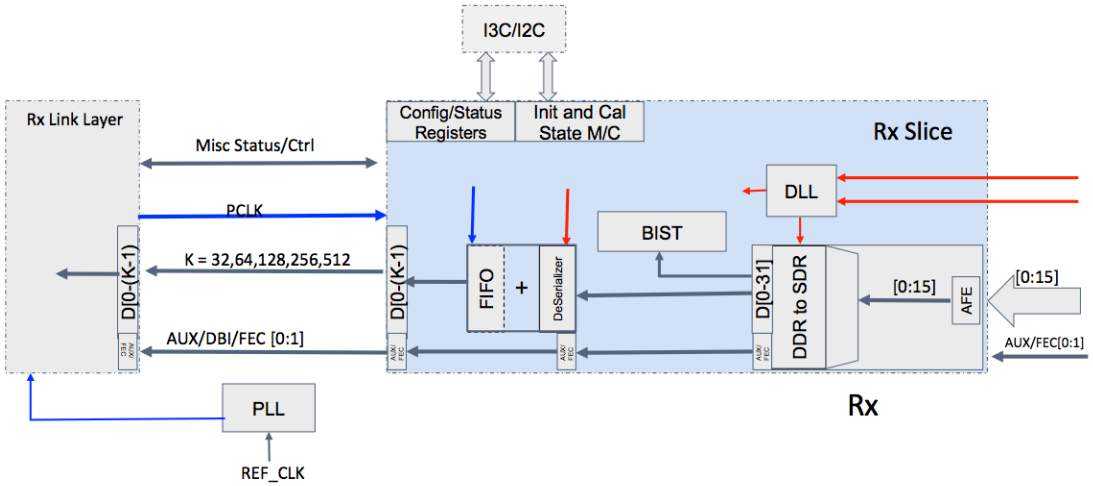


Figure 8. BoW Link Layer and Rx Slice with FIFO

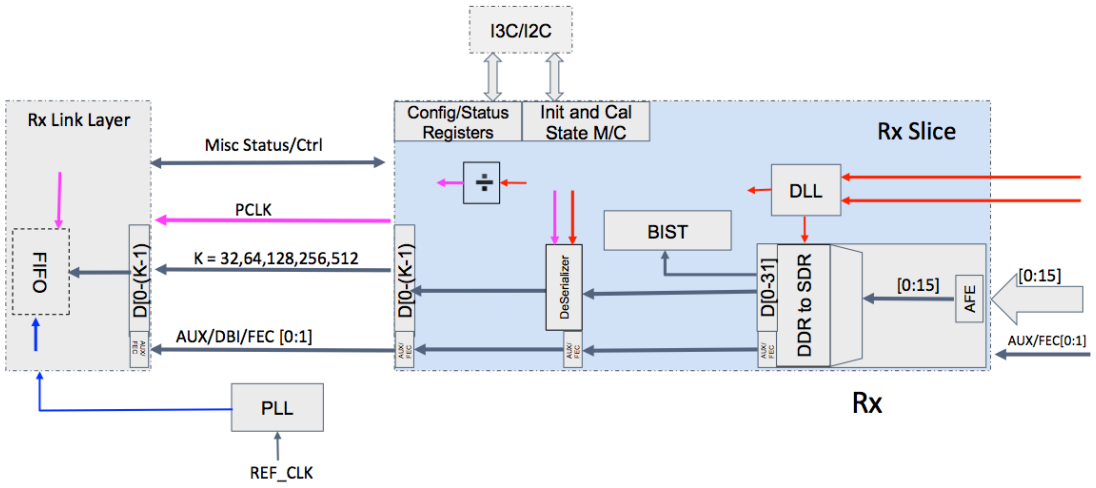


Figure 9. BoW Link Layer and Rx Slice with no FIFO

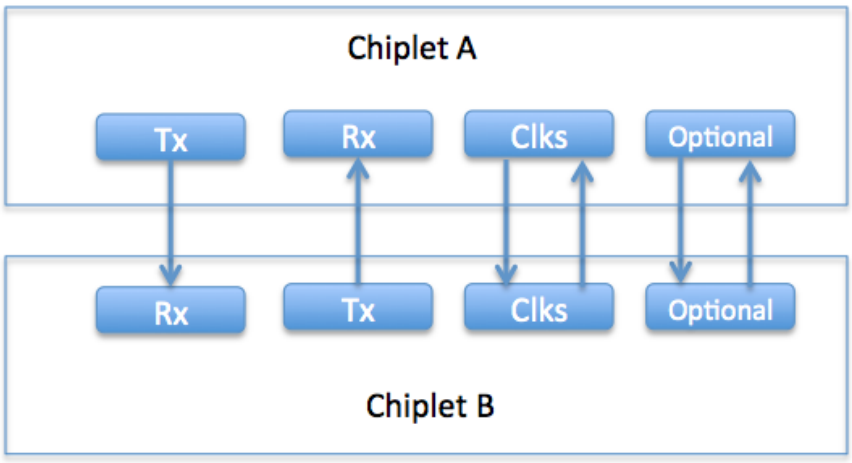


Figure 10. BoW Signal Types

- Clocks
  - Data clock out (tx_fwd_clk), sent from the interface
  - Data clock in (rx_fwd_clk), input to the interface

3.4. BoW-to-Link Layer(LL) Interface

The following signals shall constitute the interface to the MAC

3.5. BoW Modes

There are multiple BoW modes. All modes must be interoperable with the minimum definition. All modes are source synchronous parallel interfaces using a differential clock that may operate at multiple frequencies. The range of operating frequencies for a mode must be documented in a data sheet. frequencies. Beyond the basic mode, adding termination provides higher performance per mm of beachfront bandwidth but is more complex to design. It is also possible to get high performance by using

In each mode, a BoW interface can operate at multiple frequencies. The operating frequency and throughput per wire on a BoW interface mode will be affected by:

- The choice of packaging technology
- The physical distance between the chiplets being connected: Faster data rates may be easier to realize with chiplets that are physically closer
- Bump spacing: Coarse bumps may allow for circuitry to enable faster data rates.

3.5.1. BoW Basic Mode

In BoW Basic, data rates can go up to 5 or 8 Gbps, depending on the package technology and wire length. Wires are source-terminated (in the transmit circuit), but not terminated at the receiver.

Signal	Description
TX	Synchronous data transmitted from the interface.
RX	Synchronous data received from the interface.
tx_fwd_clk/tx_fwd_clkb	Transmit transfer clock, forwarded from the transmitter to its link partner for capturing received data.
rx_fwd_clk/rx_fwd_clkb	Receive transfer clock, forwarded to the receiver from its link partner for capturing received data.
tx_AUX_DBI	Optional signal from the transmitter to its link partner to indicate bus inversion status. If not used as DBI, this signal can also be used as Mode bit
rx_AUX_DBI	Optional input signal to receive bus inversion status. If not used as DBI, this signal can also be used as Mode bit
tx_FEC	Optional signal to carry FEC information from transmitter to its link partner.
rx_FEC	Optional input signal to receive FEC information.

Table 2. BoW Interface Signals

Signals	In(from LL) Out (to LL)	Description
data_in	In	For transmitting across the BoW link
data_out	Out	Received through the BoW link
tx_pclk	In/Out	Parallel Clock between LL and Tx slice In (if FIFO in Slice), Out(if FIFO in LL)
rx_pclk	In/Out	Parallel Clock between LL and Rx slice In (if FIFO in Slice), Out(if FIFO in LL)
tx_mac_rdy	In	For resetting Transmit data transfers and communicating LL readiness for calibration to the link partner
rx_mac_rdy	Out	Indicates that the Receive LL is ready to transmit data
tx_adapter_rstn	In	Resets the BoW Adapter
tx_dcc_dll_lock_req	In	Initiates calibration of transmit slice and its link partner (receive slice) for a BoW interface
rx_dcc_dll_lock_req	In	Initiates calibration of receive slice and its link partner (transmit slice) for a BoW interface
tx_transfer_en	Out	Indicate that calibration on the Transmit is complete for transmit and receive paths
rx_transfer_en	Out	Indicate that calibration on the Receive is complete for transmit and receive paths
Signals indicating any conditions that may cause de-assertion of data-transfer ready	Out	Sent to LL for possible data- transfer ready de-assertion by LL
Signals indicating any conditions that may cause BoW Adapter	Out	Sent to LL for possible adapter reset by LL

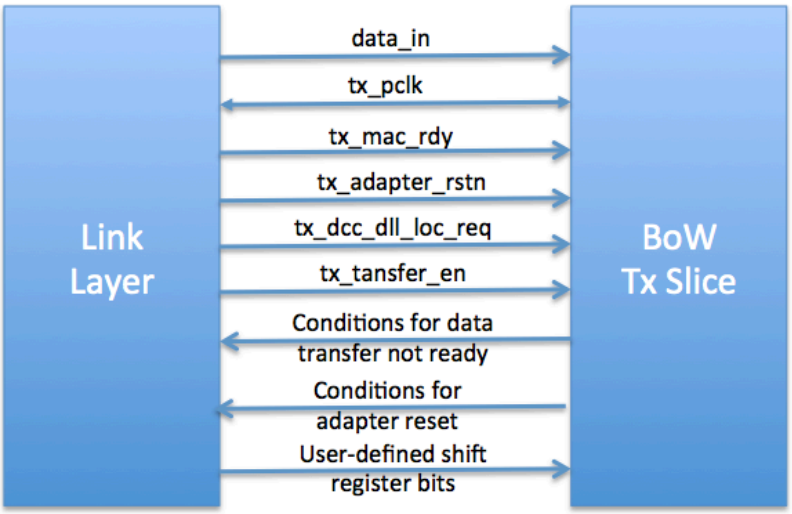


Figure 11. Link Layer Tx Slice Interface

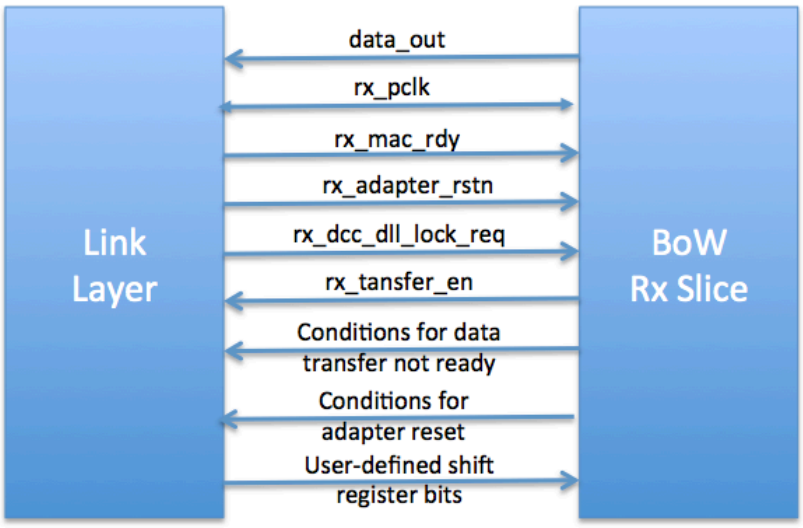


Figure 12. Link Layer Rx Slice Interface

Mode	Package	Recom Clock rate	Recom Data rate	Recommended Termination	Recom Reach
BoW Basic	All	4 GHz	8 Gbps	No	1 mm
BoW Basic	Advanced	4 Ghz	8 Gbps	Source	3 mm
BoW Basic	Laminate	4 GHz	8 Gbps	Source	10 mm
BoW Fast	Advanced	8 GHz	16 Gbps	No	1 mm
BoW Fast	Laminate	8 GHz	16 Gbps	Double	50 mm

Table 4. BoW Modes and Data Rates

3.5.2. BoW Fast Mode

In BoW Fast, data rates can go up to 16 Gbps. Wires are both source-terminated and receiver-terminated to minimize signal reflections to enable higher data rates and longer wires.

3.5.3. Modes and Data Rates

With proper attention to design,
All BoW links are DDR (double data rate). DDR is defined as one data bit transmitted for each rising and each falling edge of the clock.

Add an interop data rate columns of 2 Gbps/1 Ghz to the table.

"Laminate" is intended to include organic laminate packages (a.k.a. buildup) and similar technologies with approximately 25 um line and space rules. "Interposer" is intended to include silicon interposer and similar technologies with approximately 1 um line and space rules.

Ken sez: I copied this table from the HotI 2020 paper, but what is our supporting data for 8 Gbps and 50 mm cases?

All BoW implementations must support the minimum 1 GHz clock rate, 2 Gbps data rate using source-terminated I/O. For ease of interoperation, data rate multiples of N Gbps should be supported for integer N>=2 to the max rate for that implementation.

3.5.4. BoW Mode Extensions

System designers are allowed the flexbility to extend the two BoW modes defined above provided, at a minimum, the BoW Basic Mode (Section 2.3.1) is supported to guarantee interoperability with other BoW PHY implementations.

3.6. Control Signals

An BoW interface shall provide control and status bits for calibration and communication of control and status information between the transmitter(receiver) and receiver(transmitter) chiplets. These status bits shall be implemented as documented in Section XX.xxx.xx.

Transmit Slice Shift Register

A Transmit slice shall include two shift registers: one for holding Transmit control and status signals to its link partner (the Transmit shift register), and one for receiving control and status

Mode	Package	Clock rate	Data rate	Termination	Reach
BoW Basic	Laminate	1-2.5 GHz 1-4 GHz	2-5 Gbps 2-8 Gbps	Source	10 mm 5 mm
BoW Fast	Laminate	1-8 GHz	2-16 Gbps	Double	50 mm

Table 5. BoW Modes and Data Rates

Mode	Package	Clock rate	Data rate	Termination	Reach
BoW Basic	Advanced	1-2.5 GHz 1-4 GHz	2-5 Gbps 2-8 Gbps	Optional	2 mm 1 mm

Table 6. BoW Modes and Data Rates

signals from its interface partner (the Receive-copy shift register). The Transmit shift register shall contain 81 bits. The Receive-copy shift register shall contain 73 bits. All bits shall be implemented regardless of whether optional signals are implemented. Any signals not implemented shall permanently maintain their default values as defined in Table 50.

Receive Slice Shift Register

A Receive slice shall include two shift registers: one for holding Receive control and status signals to its interface partner (the Receive register), and one for receiving control and status signals from its link partner (the Transmit-copy shift register). The Receive shift register shall contain 73 bits. The Transmit-copy shift register shall contain 81 bits. All bits shall be implemented regardless of whether optional signals are implemented. Any signals not implemented shall permanently maintain their default values as defined in Table 51

Control Signals Control signals fall into one of the following categories:

- Calibration handshake
- Selective reset
- User-defined
- Reserved

The bits for any unused signals shall be maintained with default values for correct shift- register length. The control signals are summarized in Table 11 and are detailed in Table 51.

Calibration Status Signals

Calibration status signals shall be generated by internal state machines as described in Section 3.2.3.

User-Defined Signals

```
User-defined signals are available for application use.
Since both sides need to understand the function of user-defined signals,
using these signals may limit chiplet interoperability. If implemented in an application,
user-defined signals should be described in the chiplet data sheet.
```

Shift-Register Signals

The following datarates and reach targets are recommended for BoW implementation using advanced packaging:

All BoW links are DDR (double data rate). DDR is defined as one data bit transmitted for

Mode	Package	Clock rate	Data rate	Termination	Reach
BoW Basic	Laminate	1-2.5 GHz	2-5 Gbps	Source	10 mm
		1-4 GHz	2-8 Gbps		5 mm
BoW Fast	Laminate	1-8 GHz	2-16 Gbps	Double	50 mm

Table 7. BoW Modes and Data Rates

each rising and each falling edge of the clock.

"Laminate" is intended to include organic laminate packages (a.k.a. buildup) and similar technologies with approximately 25 um line and space rules. "Advanced" is intended to include silicon interposer and similar technologies which are often more resistive than organic laminate packages and must operate with reduced trace lengths. Termination (source or double) may not be necessary for implementations targeting Advanced packaging.

Table 10 defines the control signals for Transmit and Receive slices. The table is organized by signal type; in-order signal tables with default values are provided in Section 7.2. tx *prefixes refer to signals originating on the Transmit slice; rx* prefixes refer to signals originating on the Receive slice.

3.7. AUX_DBI

Data Bus Inversion (DBI) is intended to mitigate simultaneous switching output (SSO) noise of a BoW PHY by limiting the number of BoW data bits that can switch between immediate data transfer cycles.

DBI functionality is optional, but if implemented, DBI Mode shall be enabled by setting register bit AUX_DBI = HI

When DBI is on, DBI bits are part of the data bus [AUX_DBI + 16 bits] . The example below shows a 16 bit TX slice. RX wires have DBI at the same bit locations.

3.7.1. TX DBI with DBI Enabled

Within a group of 16 data signals, the TX DBI logic calculates the DBI bit based on the number of data signals changing from their previous state on the BoW slice data bus. The DBI logic below uses "+" to indicate arithmetic addition, "^" to indicate exclusive OR, and "?" as a ternary IF. "Current" refers to the new data word being prepared for sending on TX. "Prev" refers to the data immediately preceding the current data, that is the data issued onto the BoW bus before the current data.

DBIcurrent = ((data[15]current ^ data[15]prev) + (data[14]current ^ data[14]prev) … + (data[1]current ^ data[1]prev) + (data[0]current ^ data[0]prev)) > 8 ? 1 : 0);

Within a group of 16 data signals, if the DBI bit=1 then the TX DBI logic inverts the data signals. Each calculated DBI bit replaces one data bit in TX as described previously.

3.7.2. RX DBI with DBI Enabled

Within a group of 16 RX signals, the RX DBI logic extracts the DBI bit. If DBI = 1 then the RX DBI logic inverts the other RX bits.

			Signal Origin	**Bit Number**	
Signal Name	Signal Function	Bits	(Slice or LL)	Transmit	Rece
Calibration					
tx_dcc_cal_done	TX DCC calibration complete	1	LINK PARTNER	68	31
rx_transfer_en	RX calibration complete	1	LINK PARTNER	75	70
rx_dcc_dll_lock_req	Start RX calibration	1	LL	NA	69
rx_dll_lock	RX DLL locked	1	LINK PARTNER	74	68
tx_transfer_en	TX calibration complete	1	LINK PARTNER	78	64
tx_dcc_dll_lock_req	Start TX calibration	1	LL	NA	63
Device_ID	Chiplet ID	4	Transmit	8-11	32-35
Link_ID	Chiplet Link ID	4	Transmit	12-15	36-39
Slice_ID	Chiplet Slice ID	5	Transmit	16-20	40-43
Power_on_reset	PoR complete	1	Receive	21	44
tx_mac_rdy	For resetting Transmit data transfers and communicating LL readiness for calibration to the link partner	1	LL	22	45
rx_mac_rdy	For resetting Receive data transfers and communicating LL readiness for calibration to the link partner	1	LL	23	46
tx_adapter_rstn	Reset BoW adapter signal from Transmit slice to Link Partner	1	LL	24	47
rx_adapter_rstn	Reset BoW adapter signal from Transmit slice to Link Partner	1	Receive	25	48
AUX_DBI	Select DBI mode (1) or AUX mode (0)	1	LL	26	49

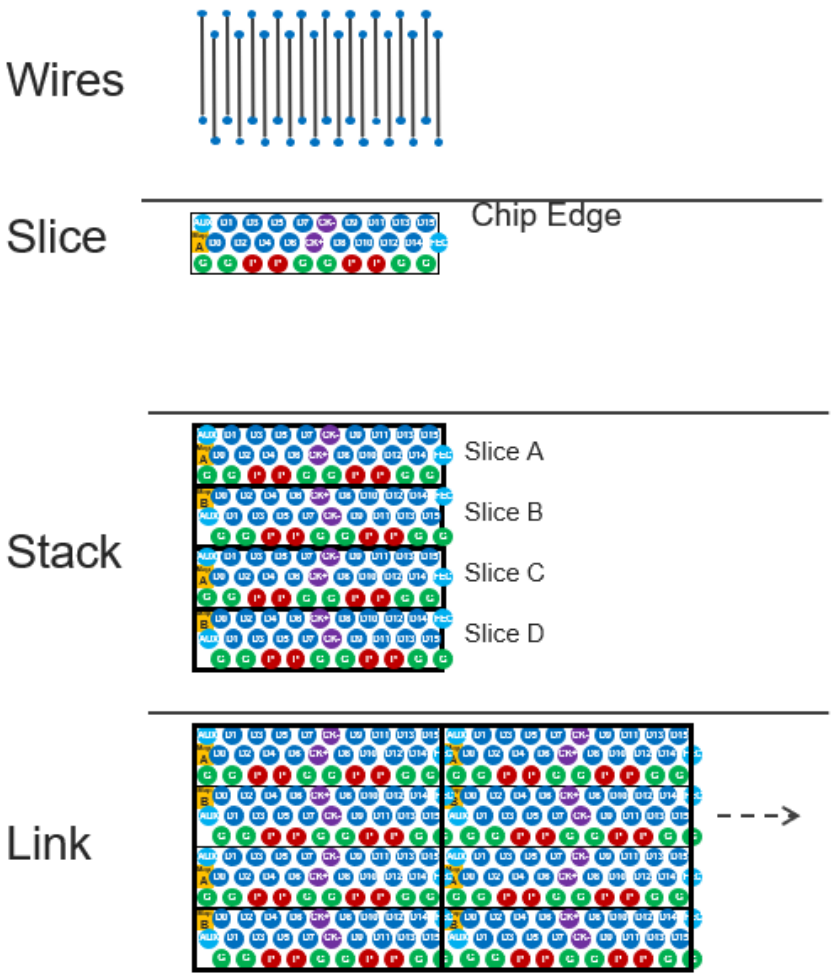


Figure 13. BoW Link Components

4. BoW Physicals

4.1. BoW Components

A BoW link between two chiplets is made up of wires, slices, and stacks.

- The basic unit is a slice with 18 or 20 signal bumps. It must have 2 signals for the differential clock and 16 single-ended data. It may also have the optional single-ended signals AUX and FEC. In the reference bump map, the long edge of a slice is parallel to the chip edge.
- Multiple slices may be placed in a stack. The slice positions are designated A, B, C, etc, starting with the slice closest to the edge of the chip.
- A link from one chiplet to another must be composed of one or more stacks placed along the chip edge. A link may be configured with equal numbers of Rx and Tx slices, or it may be asymmetric or one-way.

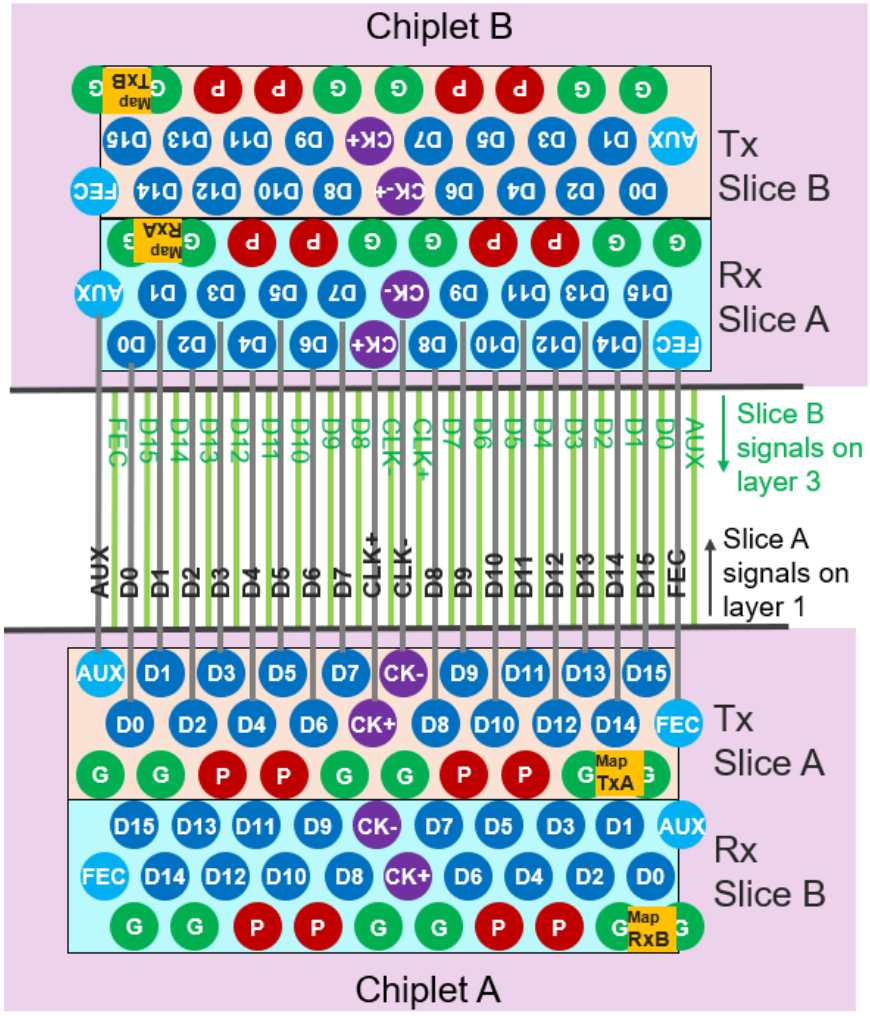


Figure 14. BoW Minimal Bidirectional Reference Link

4.2. Example Link

The minimal bidirectional reference link is shown in Figure 14.

In this example, each chiplet has one Tx slice and one Rx slice, arranged in a single stack on each chiplet. The position-A slices (at the chips' edges) are connected together on the topmost routing layer used for signals and the position-B slices are connected together on the next layer used for signals.

Each BoW slice consists of a differential clock pair, 16 single-ended data wires, and optional wires FEC and AUX. Each BoW slice is unidirectional when in operation. A chiplet may be designed with with Rx-only and Tx-only slices, or each slice may have both Tx and Rx capability, one of which is selected at runtime. A bidirectional link is composed of some number of slices configured for Rx and some for Tx.

FEC (Forward Error Correction) is an optional signal that allows using FEC to improve the bit error rate (BER). AUX is an optional signal that can be used for purposes such as DBI, flow control, redundancy, etc. Chiplets A and B will need to agree on the details on FEC and AUX

usage, which is defined in the Link layer.

This reference example uses hexagonal closest packing for the bumps: two rows for signal bumps and one row for power and ground bumps. In this pattern, the wire pitch is half the bump pitch. In order to maintain the closest bump packing, slices in rows B and D must have a different bump pattern than slices A and C. But bump patterns are not specified by BoW; only the signal *ordering* at the chip edge is specified for interoperability.

Alternate bump arrangements may include: 90-degree rotation of the hexagonal packing direction to decrease the wire pitch 14%; square bump arrays instead of hexagonal for regularity of layout; more than two rows of signal bumps to decrease the wire pitch; different ordering of power and ground bumps; multiple power and ground rows.

In an organic laminate package, signal layers should be alternated with ground layers in order to maintain a controlled impedance of 50 ohms. In interposer or other high-density packaging, the use of layers may be different.

In any technology, the position-A slice on chiplet A must be connected to the position-A slice on chiplet B (one must be configured for Tx and one for Rx). The position-B slices are connected together, and so on.

There is no specified limit to the number of slices in a stack. In organic laminate, the practical limit in 2020 is an 8-2-8 laminate which supports 4 slices. Layers on the bottom side of the package typically cannot be used for BoW signals due to low via density passing through the thick central core layer.

Function	# Signals	Signal Name	Notes
Clock	2	CLK+, CLK-	Differential
Data	16	D[15:0]	
Forward Error Correction	1	FEC	Optional
Auxiliary	1	AUX	Optional

An alternate slice arrangement may be to place the Tx and Rx slices side by side at the chip edges. This would take up more chip edge, but allow all the signals to run on the same package layer. Somewhat different wire and bump pitches between two chiplets can be accomodated with fan-out in the chip-to-chip wires. This is limited in BoW Basic by the 10 mm max wire length and the practical advantages of routing from chiplet to chiplet in just one package layer. (Note that even straight wires in slice position D are likely 5-6 mm long using 130-um bump pitch with the reference bump map in an organic package.)

4.3. Cross Section

A cross section for an organic laminate (a.k.a. “buildup”) package is shown in Figure [#fig-pkg_cross_section].

~ Figure { #fig-pkg_cross_section; caption: “Cross section of a BoW Link in an Organic Laminate Package” }

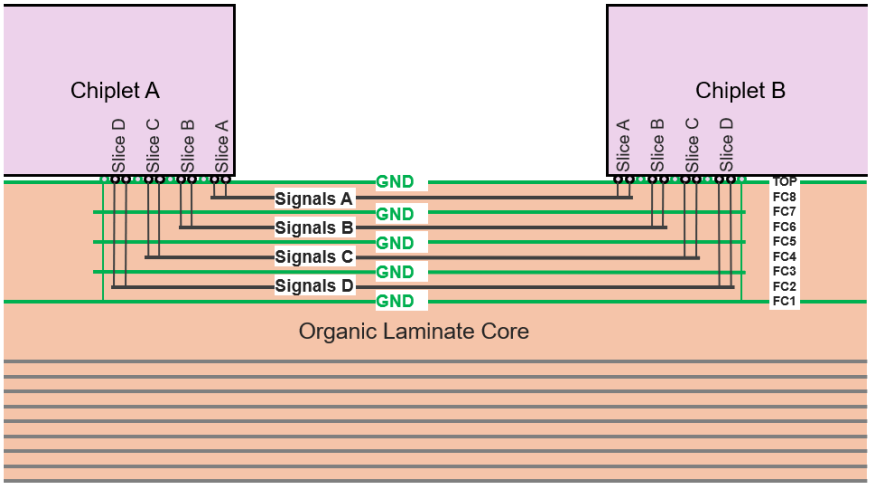


Table 9. BoW Signals

4.4. Signal Ordering

Ken sez: I copied this table from the HotI 2020 paper, but what is our supporting data for 8 Gbps cases?

A BoW interface must conform to these wire and slice order rules at the edge of the chip:

- The signals for a Tx slice are in the following order at the chip edge, going clockwise around the chiplet: AUX, D0, D1, D2, D3, D4, D5, D6, D7, CLK+, CLK-, D8, D9, D10, D11, D12, D13, D14, D15, FEC
- The signals for an Rx slice are in the reversed order (ascending goes counter-clockwise)
- The Tx slices in a link are numbered from 0 at the upper left edge of the link (facing from the chip center to the edge) and ascending through the Tx slices in a stack, then from stack to stack clockwise.
- The Rx slices in a link are numbered from 0 at the upper right, through the Rx slices in a stack, then stack to stack counterclockwise.

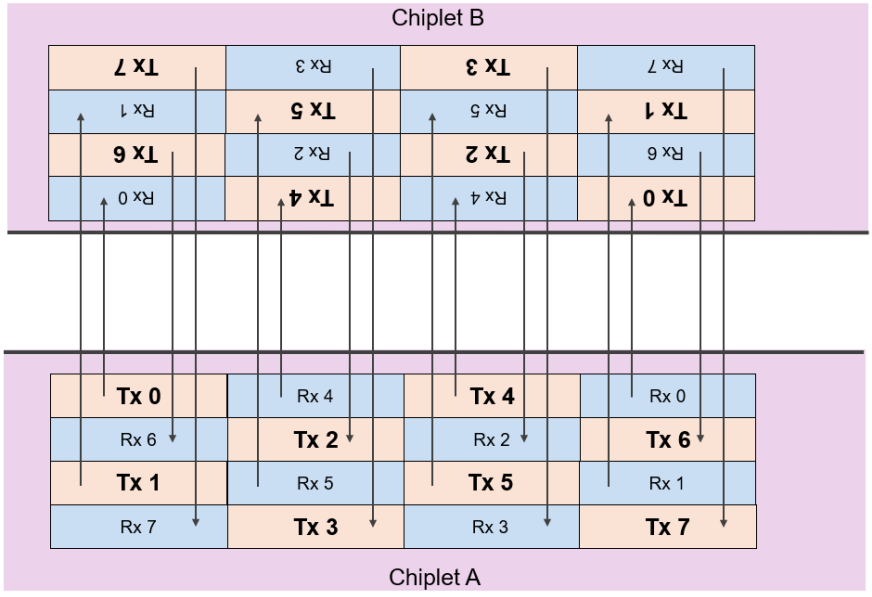
These rules allow BoW chiplets to be connected without signal reordering regardless of chiplet rotations.

In organic packages, each slice position should be associated with one signal layer and there is no mixing of signals from multiple slices. In interposer or other dense routing technologies, signals from multiple slices in a stack may be interleaved in a signal layer, but chiplet A's slice A is still connected to chiplet B's slice A and similarly for position B, C, etc.

4.5. Checkerboard Slice Pattern

For bidirectional links with more than one stack on each side, a checkerboard pattern of Tx and Rx slices should be used (Figure [#fig-checkerboard]). This allows connection of chiplets with differing stack depths and numbers of stacks to be as efficient as possible. Figure [#fig-checkerboard] shows a bidirectional link with 4 stacks or 4 slices each, for 8 Tx and 8 Rx slices on each chiplet.

~ Figure { #fig-checkerboard; caption: "Checkerboard Pattern of Tx and Rx Slices in a 1.25 Tb/s Link" }



An alternate approach may be used: provision every slice to operate as either Rx or Tx. This allows

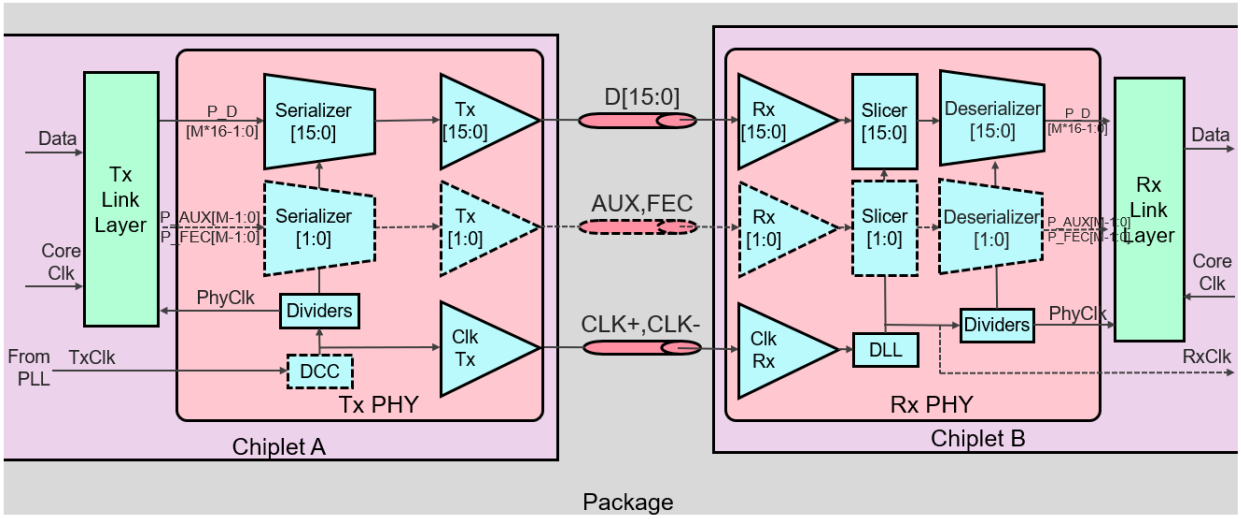


Figure 15. BoW Clock and Data Block Diagram - One Tx Slice, One Rx Slice

complete flexibility in interoperability and also provides an opportunity for wafer-test loopback testing.

In BoW Base at 5 Gb/s, link in Figure [#fig-checkerboard] provides a total of 1.25 Tb/s. In an organic substrate using the hexagonal bump pattern of Figure 14, if the bump pitch is 130 um, the total edge width is 4.16 mm without AUX and FEC, or 5.2 mm with; the depth from the edge is 1.35 mm. In an interposer, if the bump pitch is 40 um, the edge width is 1.28 or 1.60 mm and the depth is 0.42 mm.

5. Timing

5.1. Clocking

Figure 15 shows the clock and data flow for a single Tx slice and a single Rx slice. On the Tx side, data bits (and optional FEC and AUX bits) come in a wide word from the link layer, and are serialized to the line rate. At the Rx side, they are sampled with a common slicer clock in BoW Basic. BoW Fast may optionally implement per-bit delay adjust or per-bit slicer clock adjust.

All BoW interfaces shall be DDR (Double Data Rate) at the chip-to-chip interface: the data bit rate is twice the clock frequency, so data is clocked in on both edges of the clock in the Rx slice.

The following table provides clock and data rates for an example with 4 Gbps wire data rate and M=4 to support a 1 Gbps data rate at the Link-PHY interface.

The ratio M should be limited to integers, preferrably powers of two, and other ratios implemented in a gearbox in the Link layer.

The DDR clock TxClk is provided to the Tx PHY from elsewhere on Chiplet-A. This may come from an on-chip PLL (typically shared across multiple slices) or it can be routed from the RxClk of an Rx slice on Chiplet-A. In order to meet duty cycle requirements, a Duty Cycle Corrector (DCC) may be needed in the Tx slice. TxClk is used to drive the serializers and provide the output CLK+,CLK- to Chiplet-B.

On the Rx side, the PHY must align the slicer clock to sample the data correctly. This may be

Signal	Rate	SDR/DDR
TxClk	2 GHz	
CLK+,CLK- D[15:0],AUX,FEC	2 GHz 4 Gbps	DDR
PhyClk P_D[63:0],P_AUX[3:0],P_FEC[3:0]	1 GHz 1 Gbps	SDR

Table 10. Example Clock and Data Rates for Figure 15 with 4 Gbps, M=4

done with a DLL or adjustable delays or other methods. The PHY shall include control logic to self-align the slicer clock for correct sampling of the data. Alignment is started by signal AlignDll from the Rx Link Layer; the PHY provides a signal DllAligned to the Link Layer when it is complete.

All BoW interfaces shall be source synchronous within a slice. BoW Basic interfaces do not require per-wire alignment - the signals within a slice are aligned sufficiently well by matching their paths. BoW Fast interfaces may need per-wire delay adjustment or per-slicer clock adjustment.

Clock skew between the slices in each direction of a link depends on the implementation of the TxClk distribution to all the Tx slices. That is, for the data flow from Chiplet A to Chiplet B, the TxClk distribution on Chiplet A sets the the clock skew of the Tx slices on Chiplet A and the clock skew of the Rx slices on Chiplet B, and vice versa for flow from B to A. This skew must be no more than 3 ns. There is no specification of the skew between TxClk on Chiplet A vs TxClk on Chiplet B.

Ed: 3 ns is sufficient to cover Tx clock distribution skew across a 30 mm chip edge. Specifying a skew limit seems useful to clarify the requirements that will be imposed on the bonding FIFO in the Rx Link layer. Do we want to weaken this to a Should?

On both the Tx and Rx sides, the Link layer must include a Clock Domain Crossing (CDC) to align the data between CoreClk and PhyClk. These CDCs must also be able to absorb the slice-to-slice clock skew and core clock distribution skew across the whole link.

If DCCs are included in the PHY and they need an alignment cycle, they shall include control logic to perform self-alignment on a signal AlignDcc from the Link layer to the PHY and send a completion signal DccAligned back to the Link layer.

Ed: Need to use the same names for AlignDll, DllAligned, AlignDcc and DccAligned as in Logic Interface section.

5.2. Clock and Data Specifications

5.2.1. Timing Requirements

Figure 16 shows the definition of the eye diagram parameters.

The CLK and data signals at the receiving slice bumps must be able to meet these conditions:

V_{hi} of 0.75 V must be supported by all BoW implementations, but other values may be supported.

t_{eye} must be evaluated for each of the bits in a slice relative to the differential CLK+ - CLK- signal for that slice. t_{eye} must be evaluated for CLK edges up to 3 UI earlier than the eye center. This is because even though jitter on the data edges is correlated with the CLK jitter at the Tx side, the slicer in the Rx side is likely to use a different CLK edge due to delays in the Rx-side

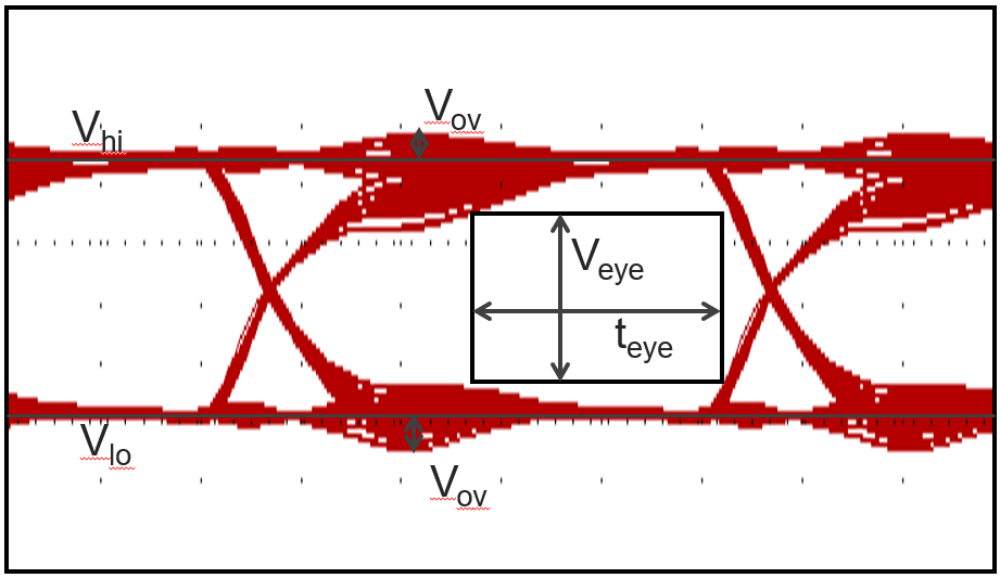


Figure 16. Eye Diagram Definitions

Symbol	Spec	BoW Basic	BoW Fast
V_{hi}	High signal voltage	750 mV	562 mV
V_{lo}	Low signal voltage	0 mV	188 mV
V_{tol}	Tolerance of Vhi, Vlo (5%)	+- 37 mV	+- 19 mV
t_{eye}	Data eye width	50% UI	50% UI
V_{eye}	Data eye height	40%(V_{hi}-V_{lo}) (300 mV)	20%(V_{hi}-V_{lo}) (75 mV)
V_{ov}	Data and CLK overshoot	25%(V_{hi}-V_{lo}) (188 mV)	50%(V_{hi}-V_{lo}) (188 mV)
t_{skew}	Slice to slice CLK skew	3 ns	3 ns

Table 11. BoW Basic Timing Requirements

Spec	Value	Normalized Value	Condition
Bit to bit skew: fanout	0 ps	0.00 UI	fanout for mismatched bump pitches
Bit to bit skew: mismatch	3.6 ps	0.014 UI	0.6 ps rms device mismatch (6 sigma)
Bit to bit skew: clock distn	3.9 ps	0.016 UI	0.65 mm from CLK to bits D0, D15
CLK to mean data skew	12 ps	0.05 UI	error of DLL alignment
CLK duty cycle	50%	0.50 UI	target
CLK duty cycle error	2%	0.02 UI	after DCC settled
CLK+ vs. CLK- skew	3.6 ps	0.014 UI	
CLK cyc-to-cyc random jitter	TBD		thermal jitter of the whole Tx clock path
CLK cyc-to-cyc bounded jitter	TBD		bounded jitter of the whole Tx clock path
CLK rise/fall time	43 ps	0.17 UI	
Data rise/fall time	43 ps	0.17 UI	
Data Eye width	125 ps	0.50 UI	not including jitter
Data Eye height	450 mV	60% Vswing	Vswing = nominal signal voltage swing
Data overshoot/undershoot	75 mV	10% Vswing	

Table 12. BoW Clock Reference Values

clock alignment circuit (usually a DLL). The evaluation of jitter must include all possible jitter contributors, including reference clock, clock distribution networks, any DCC, PLL and DLL jitter, power-supply noise and switching noise.

The slice to slice clock skew t_{skew} across the width of a BoW link must be less than 3 ns. This is dominated by the TxClk distribution network.

Since these signals do not leave the package, these values must be verified with simulation.

If the slice implementation allows programmatic control of the DLL alignment values, varying those values after locking the DLL may provide timing margin information. If the slice implementation allows programmatic control of the receiver voltage thresholds, varying those values may provide vertical margin information.

Should we require programmatic DLL control to enable margin testing?
Should we require threshold voltage control to enable margin testing?

5.2.2. Reference Slice Timing

Is this section worth keeping?

The following table shows clock and data timing for a 4 Gbps BoW Basic reference design in 65 nm CMOS on laminate packaging. These values are for CLK and data bits at the bumps of the BoW receiver.

The jitter in a link should meet:

DataEyeWidth - BitToBitSkewFanout - BitToBitSkewMismatch
- BitToBitClockDistn - CLKtoMeanDataSkew - CLKdutyCycleError
- CLKrandomJitter - DataToDataSkew $<$ 0.10 UI

6. Electrical Specifications

6.1. Voltages

To support a wide range of technologies BoW does not enforce a common VDD power rail voltage. In order to ensure that there is at least one common interoperable mode BoW implementations

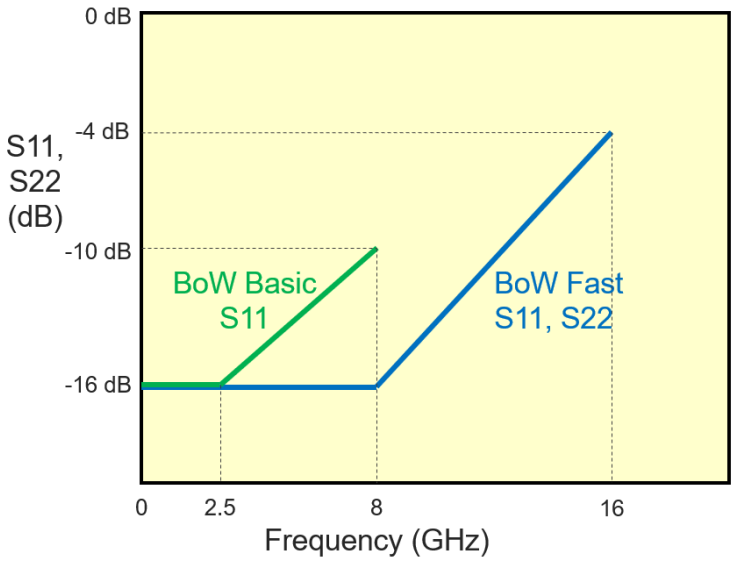


Figure 17. BoW Termination Return Loss S11 and S22

must support electrical specifications targeting rail to rail signaling based on a 0.75 V +/- 5% power supply (see electrical specifications). BoW interfaces can support higher or lower signaling voltages without limit but must support 0.75 V based signaling at 2 Gbps Double Data Rate (DDR) to ensure interoperability. BoW does not enforce a common VDD power rail voltage.

6.2. ESD

BOW I/O shall be designed to withstand 50 V CDM (Charged Device Model) and 250 V HBM (Human Body Model). This requirement is similar to other die-to-die interface standards.

6.3. Termination and Return Loss

On laminate or similar packages, BoW Basic and BoW Fast transmitters shall be source-terminated to 50 +/-8 ohms.

BoW Basic receivers shall be unterminated. BoW Fast receivers shall be terminated in 50 +/-8 ohms to $V_{swing}/2$ +/- 5% where V_{swing} is the signal voltage swing without a receiver termination.

A BoW Basic transmitter (both data and clock) shall meet the return loss specified in figure 17 when outputting a logic 0, 1 or at the midscale voltage. A BoW Fast transmitter and receiver shall meet the return loss specified in the same figure, when outputting a logic 0, 1 or at the midscale voltage.

A Source-Series-Terminated (SST) transmitter can meet the the BoW Basic specification.

7. Chip-to-Chip Channel Specifications

The channel (wires) between chips is not a required part of the specification of the PHY, but implementations should meet the following specs to ensure signal integrity.

Should the channel between chips be Shall or Should?

Parameter	Value	Comment
Length	10 mm @ 5 Gbps 5 mm @ 8 Gbps	
Length mismatch within a slice	1 mm	= ~6 ps <= 0.05 UI
Impedance	50+-5 ohms	
C_{cross}/C_{total} ratio	< 40%	
R_{series}	< 4 ohms	

Table 13. BoW Clock Specifications

7.1. BoW Basic Channel Specifications

BoW Basic channel length on laminate is limited by the round trip reflection delay to 10 mm for 5 Gbps.

BoW Basic channels longer than 2 mm (e.g., on laminate) should meet these specs:

C_{cross} is the total capacitance of a wire to all its neighboring wires. C_{total} is the total capacitance of a wire including to grounds. Stripline (buried) wires with 25 um line, 40 um space and 33 um dielectric thickness have C_{cross}/C_{total} of 28% and an impedance of 48 ohms.

Channels shorter than 2 mm (e.g., on interposer) will have different requirements, not yet specified.

Interposers have only 4 layers, very thin dielectrics, cannot reach 50 ohm impedances without huge series R. So they need to be designed differently.

Kens quick study found wire lengths from 570 to 1380 um (for 55 um bump pitch, 4-slice stack, 0.1 mm chip to chip spacing). The best config looked like 3 um line and space, alternating signal and ground layers. This gives a wire pitch of 1/9 the bump pitch, so we could even support 4 slices in just one signal layer. The line C for 1400 um is 0.76 pF, the R is 14 ohms, the RC is 11 ps, which is okay for 8 Gb/s. Source termination is probably not wanted in this case since there is so much Rline, the lines are short and we cant build a reasonable controlled impedance line.

Is this enough data to spec a BoW Basic on interposer?

Does anyone have more data?

7.2. BoW Fast Channel Specifications

BoW Fast channels should meet the following characteristics.

7.2.1. Channel Loss

To minimize the need for equalization, the channel should meet the limit in Figure 18.

7.2.2. Crosstalk

The crosstalk in the channel should meet the limit in Figure 19.

Power-sum crosstalk is the sum of crosstalk power of all aggressors on a target trace. The limit is

$$\text{XtalkLimit} = -37 * e^{-f/8GHz} - 10 \text{ dB}$$

Proposed text that needs numbers:

An example geometry which meets this limit is microstrip (surface) lines with X um line and Y um space, on Z um dielectric over a ground plane.

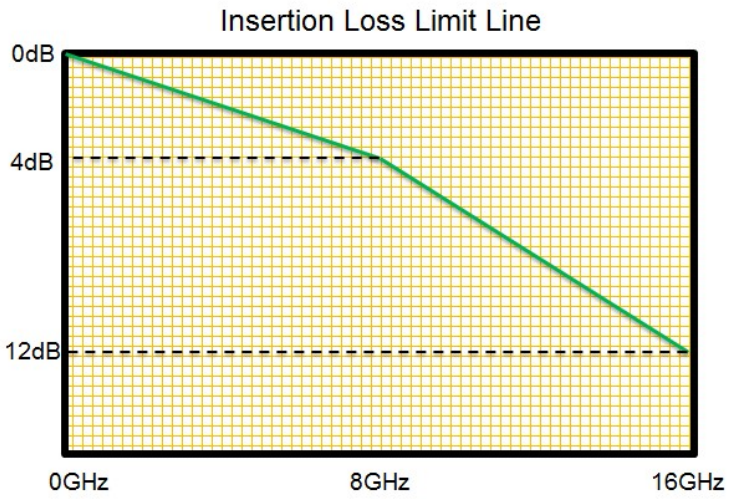


Figure 18. BoW Fast Wire Channel Loss Limit

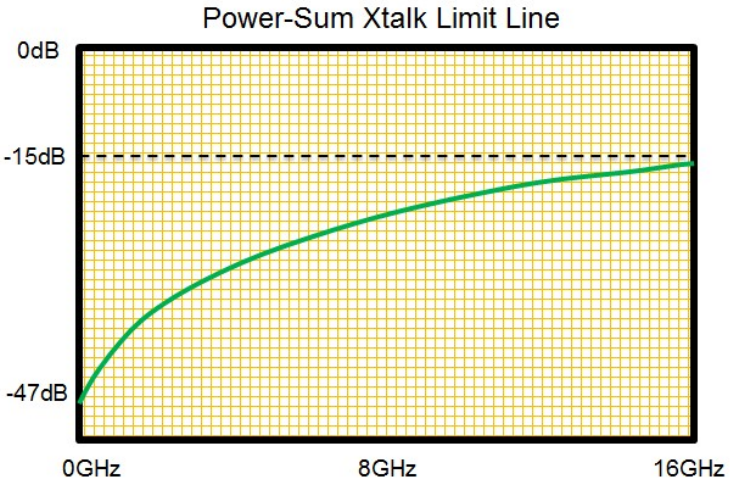


Figure 19. BoW Fast Wire Crosstalk Limit

What is the origin of the Xtalk Limit equation? Is there a paper reference?

8. Reset and Initialization

8.1. Control Options

In a system with one or more BoW interfaces, each interface pair (defined as one Tx slice on a first chiplet and one Rx slice on a second chiplet) in the system shall achieve interface ready status in each of its component slices. Once done, the interface shall signal readiness to the rest of the system. If any BoW link or slice is down (either at the MAC or the PHY level), it shall communicate this information to the appropriate interface partner as well as to the rest of the system.

Calibration and training will require the two endpoints of an interface to exchange status and control information. There is no dedicated sideband control interface defined. Instead, this exchange shall be facilitated using an independent I2C (I3C) interface, assumed to exist outside of the BoW interface, on each chiplet. I2C(I3C) was chosen as the preferred interface for the following reasons:

- It is a two wire interface
- It is multi-point
- It supports multi-leader and multi-follower topology
- The protocol for data reads and writes is standardized
- It is widely used as a configuration/status monitoring interface
- It does not require additional control signals such as chip select (SPI, one per follower) or BSDL files (JTAG) to access register information

However, the system designer is free to choose any suitable method for their application.

An example BoW system configuration is shown in Figure 3. Any of the chiplets shown can act as an I2C (I3C) leader or follower. Alternatively, a central system controller (shown in dotted lines in the figure) shall behave as the leader and the BoW chiplets shall be followers.

To facilitate device identification and target communications at the proper device, link, and slice, each BoW Interface shall have a unique Device_ID, Link_ID, and Slice_ID. A BoW interface mapping table [connection topology] should also be
provided by the system designer to facilitate proper assignment of link and slice states on each of the interface partners during initialization, calibration, or other sideband activity. The specifics of how this is topology information is propagated to each chiplet is left to the system designer.

8.2. Data-Transfer Ready

A data-transfer ready signal shall be made available for control by the MAC layer. The data-transfer ready signal may be de-asserted due to application-driven changes, including but not limited to:

- An intentional change in clock frequency
- Receipt of bad data

De-asserting the data-transfer ready signal may also be necessary due to conditions within the BoW interface, which may include but are not limited to:

- Completion of configuration during power-up

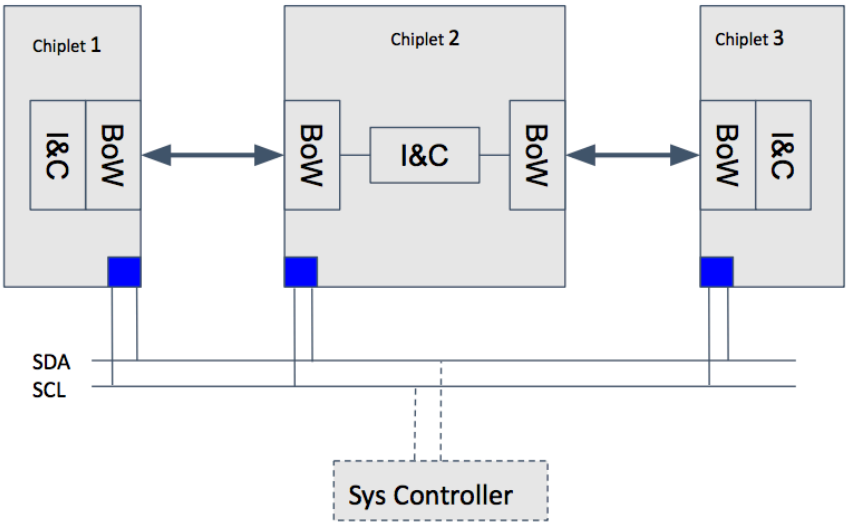


Figure 20. Example BoW System Configuration

- Initiation of reset by the link partner
- Loss of DLL lock

Internal BoW conditions indicating the need for de-assertion of the data-transfer ready signal shall be sent to the MAC so that the MAC can de-assert the data-transfer ready signal. Once data-transfer ready has been re-asserted after having been de-asserted, the BoW Adapter shall be re-calibrated (Section 11). The reverse is not true: calibration may be initiated without de-asserting data-transfer ready first.

8.3. Standby Mode

A signal shall be placed into standby mode by one of the following means:

- Driving the signal LO
- Putting the signal into tristate and enabling the weak pull-down. During initialization, data outputs shall be placed into standby mode.

8.4. Data-Transfer Ready Signals

Each slice shall have a tx_mac_rdy signal that is controlled by the Transmit MAC. When the tx_mac_rdy signal is asserted HI by the MAC, it shall indicate that the transmit slice is ready for calibration and data transfer. De-assertion of tx_mac_rdy shall affect only its own slice; other slices may continue transmitting data.

```
The tx_mac_rdy signal shall be forwarded to the link partner, and appropriate status
and control register shall be updated, in order to inform the Receive MAC
that the Transmit MAC is or is not ready for calibration.
```

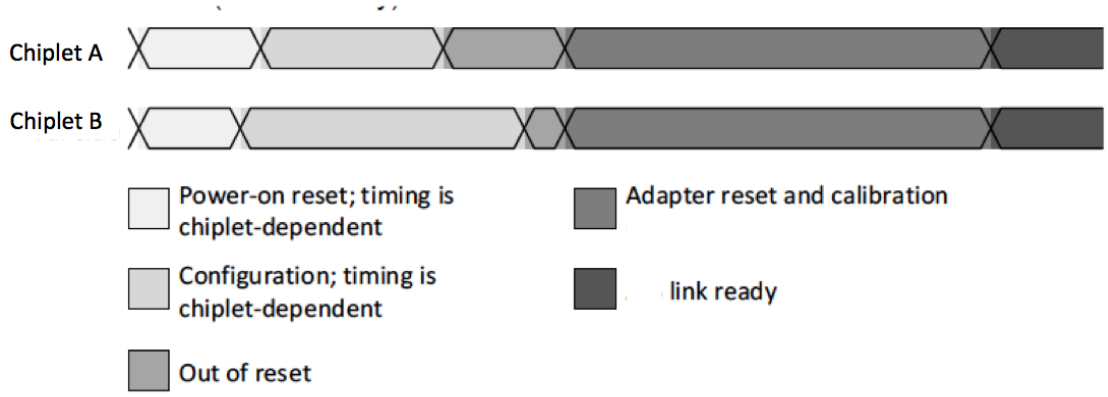


Figure 21. BoW Initialization

8.5. The Effects of De-asserting Data-Transfer Ready

While the tx_mac_rdy signal is de-asserted:

[Note: because this is not a hardwired signal, there is a latency in response depending on the polling frequency of the I2C sideband interface. Need to investigate whether there is a mechanism where each chiplet can become a Leader dynamically and broadcast changes in status to its link partner]

- Data transmission shall halt
- Data outputs shall be placed into standby mode (Section 9.3)
- The clock output tx_fwd_clk shall go into standby mode.
- The reset signal tx_mac_rdy shall be sent to the interface partner in order to communicate that data transmission has halted and to allow for the interface partner to be reset.

The contents of any retiming registers in the data path shall be undefined following de- assertion of data-transfer ready. De-assertion of the data-transfer ready signal shall not affect the free-running clock signals or the -control signals.

8.6. Initialization

Initialization will consist of three steps in sequence:

- Power-on reset synchronization
- Configuration
- Calibration

If there are multiple BoW interfaces on a single chiplet, they shall all come out of configuration at the same time, but they may complete adapter reset and calibration at different times depending on implementation.

8.7. Power-on Reset Synchronization

Power-on reset, being the first step in initialization, shall not require any features enabled by configuration, since configuration will not occur until after power-on reset.

8.8. Power-on Reset Signals

One signal (register bit) shall participate in power-on reset: power_on_reset.

8.9. Power-on Reset Sequence

During power-on reset, all input and output signals shall be placed into standby mode (Section 9.3). The power-on reset sequence shall proceed as follows:

1. If no paired device_ID signal is detected by the Receive, then the Receive may act both to ensure that it and its chiplet are in a safe state and to alert the MAC.
2. If no paired device_ID signal is detected by the Transmit, then the Transmit may act both to ensure that it and its chiplet are in a safe state and to alert the MAC
3. Each chiplet shall implement its own power-on reset routine. At the beginning of the routine, Receive interfaces shall assert their power_on_reset signals HI.
4. When a chiplet completes its power-on reset sequence:
    a. Transmit interfaces shall begin the configuration stage.
    b. Receive interfaces shall de-assert their power_on_reset signals LO and begin the configuration stage.

8.10. Unused Interfaces

In order to ensure correct operation for chiplets with unused Transmit interfaces, the power_on_reset register bits for those unused interfaces shall be set HI. In order to ensure correct operation for chiplets with unused Receive interfaces, the Device_ID register bits for those unused interfaces shall be set to 0x0000.

8.11. Test Provision

In order to test the power-on reset sequence at the wafer level, two signals shall be provided for use by automated test equipment to override the power_on_reset and device_detect signals when there is no Transmit/Receive pair available. por_ovrd overrides the power_on_reset signal, and device_detect_ovrd overrides the device_detect signal.

9. Configuration

Configuration may include:

- Host chiplet configuration (in the case of an FPGA or similar chiplet)
- BoW interface configuration
- BoW redundancy activation

The tx_mac_rdy signal shall be de-asserted LO during configuration and shall be asserted HI when configuration completes and the chiplet is ready for calibration and data transfer. The clock input from the MAC shall be stable prior to assertion of tx_mac_rdy.

9.1. Output State During Configuration

All outputs, including data outputs, shall be in standby mode (Section 9.3) during configuration.

9.2. Chiplet Configuration

Configuration of any non-BoW aspects of the chiplet is outside the scope of this specification.

9.3. BoW Interface Configuration

All intended BoW features shall be configured at power-up.

9.4. JTAG Configuration

The chiplet data sheet should document the configuration requirements that allow for successfully implementation of JTAG EXTEST and INTEST operations.

Control Shift Register Readiness

The control shift register shall be operational once configuration is complete.

9.5. Configuration Completion Signals

[Instead of conf_done, can we use this mechanism as a global interrupt to allow the side-band to respond with more immediacy or retain its conf_done status prior to calibration. Post link initialization, reuse it as a global interrupt signal. I think this can be done cleanly based on the status of a number of other signals participating in the initialization process]

Each chiplet shall have a conf_done signal. conf_done shall be an open-drain output. It shall be asserted LO when configuring, and it shall be released when configuration of all interfaces on the chiplet is complete, the analog circuits are stable, and the free-running clock is stable. conf_done shall indicate only that BoW configuration is complete. No other configuration completion (MAC, FPGA, etc.) shall be included in the generation of the conf_done signal.

All conf_done signals from all chiplets of a module should be connected in a wired-AND configuration to generate a module-level CONF_DONE signal that shall be HI when all chiplets on the module have completed BoW configuration. The pull-up resistor used to implement the wired-AND function may reside on the module containing the chiplets with BoW interfaces, or it may reside off the module. The CONF_DONE signal should be provided as an output of the module regardless of the resistor placement.

Data outputs shall remain in standby mode (Section 9.3) until CONF_DONE is asserted, including in the case where CONF_DONE is pulled low some time after being asserted high.

The resistance and VDD values should comply with

10. Calibration

The calibration sequence shall proceed as follows:

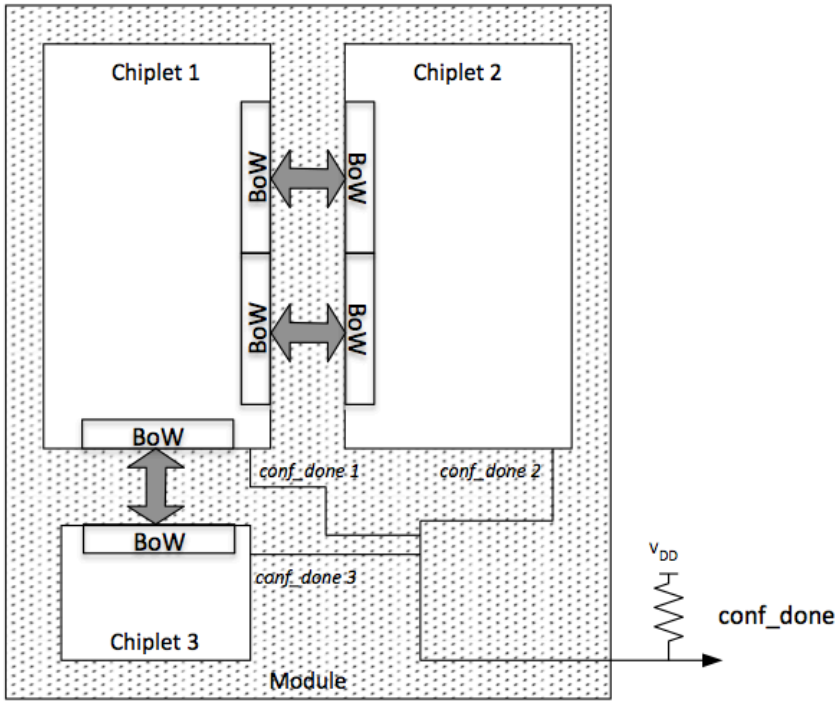


Figure 22. Configuration Complete Signals

Parameter	Value
Pull-up resistance	1kOhm
Pull-up Vdd	0.9V

Table 14. Wired-AND Pull-up Specifications

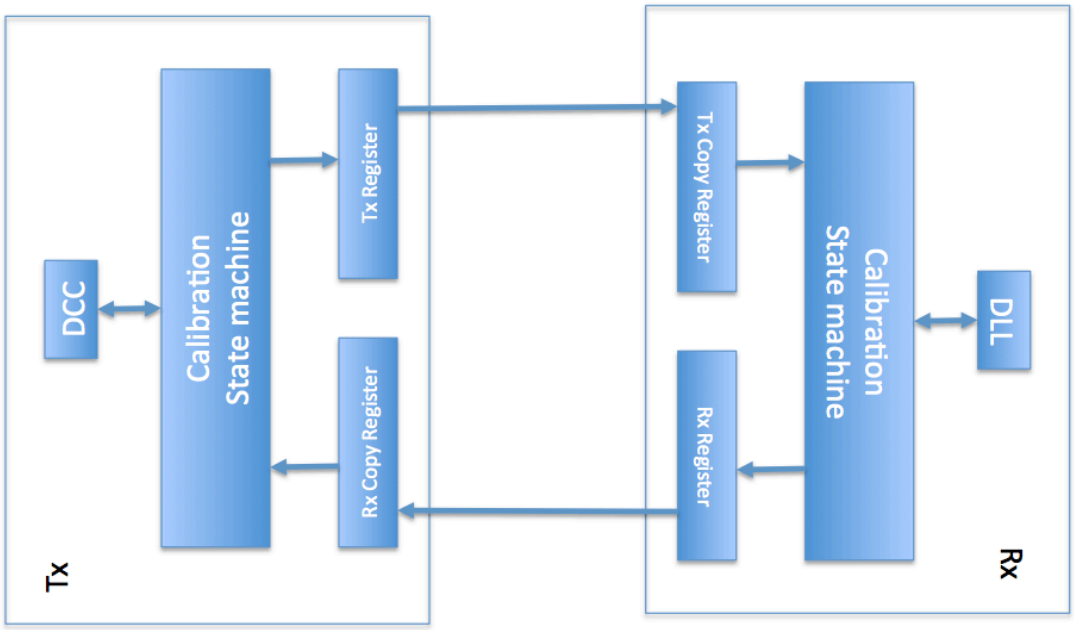


Figure 23. Data-Path Calibration Architecture

- Adapter reset
- Data path calibration

A BoW interface shall have an tx_adapter_rstn signal that is asserted by the MAC. It shall be forwarded to the link partner of the interface through the register interface.

When either the Transmit (tx_adapter_rstn) or Receive (rx_adapter_rstn) adapter reset signal is asserted LO, the adapter shall reset the calibration state machines. If adapter reset follows de-assertion of data- transfer ready, tx(rx)_mac_rdy must be asserted HI before tx(rx)_adapter_rstn is asserted HI.

10.1. Data-Path Calibration

Data-path calibration shall be implemented via state machines on the Transmit and Receive sides of the interface that intercommunicate via the control signals.
Following the de-assertion of the adapter-reset signal(s), a calibration request shall be made by asserting a calibration request signal. Either the transmit or receive slice can initiate data path calibration.

Datapath calibration shall comply with Figure 24. The numbers in black indicate the sequence of steps
Signals used in the datapath calibration sequence are listed in Table 15.

10.2. Adapter Reset

Data-path calibration shall be initiated when the MAC layer asserts the tx_adapter_rstn signal LO. If the data-transfer ready signal was de-asserted prior to the start of calibration, then the

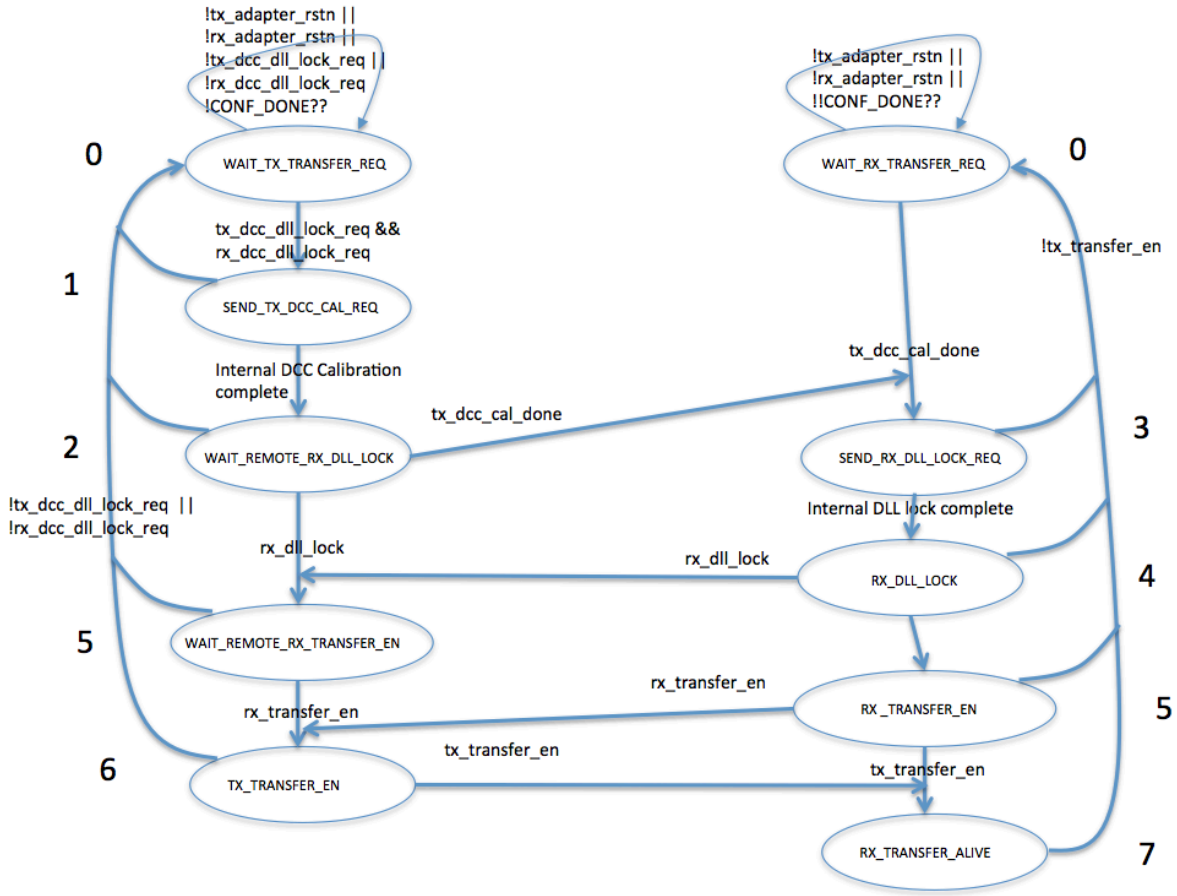


Figure 24. Datapath Calibration State Machine

Signals	Description
rx_dcc_dll_lock_req	Request from Receive to start calibration. Once asserted,shall remain asserted until a new calibration is requested.
tx_dcc_dll_lock_req	Request from Transmit to start calibration. Once asserted, shall remain asserted until a new calibration is requested.
tx_dcc_cal_done	Indicates that Transmit has completed its DCC calibration. Once asserted, shall remain asserted until a new calibration is requested.
rx_dll_lock	Indicates that Receive has completed its DLL lock procedure. Once asserted, shall remain asserted until a new calibration is requested.
rx_transfer_en	Indicates that Receive has completed its RX path calibration and is ready to receive data. Once asserted, shall remain asserted until calibration is complete.
tx_transfer_en	Indicates that Transmit has completed its TX path calibration and is ready to receive data.

Table 15. Datapath Calibration Signals

Calibration Initiator	Dataflow direction	Initiation signal
Transmit	Transmit to Receive	tx_dcc_dll_lock_req
Receive	Receive to Transmit	rx_dcc_dll_lock_req

Table 16. Calibration Initiation Signals

tx_mac_rdy signal must be asserted HI prior to asserting the adapter-reset signals HI The MAC must de-assert the adapter-reset signal prior to requesting calibration start.

10.3. Calibration Request

Calibration can be requested by either the Transmit or the Receive slice using the tx_dcc_dll_lock_req signal or the rx_dcc_dll_lock_req signal, respectively.

Calibration for a dataflow direction shall commence when either Transmit and Receive side has asserted its calibration request signal for that dataflow direction. Calibration request signals shall remain asserted until a new calibration is requested.

10.4. DCC Calibration

Upon receipt of an xx_dcc_dll_lock_req signal, the DCC shall be calibrated. The means of calibration is not specified and is left to the designer. If the optional DCC is not present, then the state machine in Section 11.1 shall remain the same, with the DCC calibration state serving only to provide a signal indicating DCC calibration completion.

10.5. DLL Calibration

Following DCC calibration, the receiving DLL shall be calibrated. The means of calibrating the DLL is not specified and is left to the designer. If the optional DLL is not present, then the state machine in Section 11.1 shall remain the same, with the DLL lock state serving only to provide a signal indicating DLL lock completion.

10.6. Calibration Completion

Calibration completion shall be indicated by the following signals. Full completion shall be indicated when all four signals are asserted HI. All four signals, once asserted, shall remain asserted until a new calibration sequence is requested.

10.7. BoW Link Ready

When both tx_transfer_en and rx_transfer_en are true, then the link shall be ready to transmit data.

10.8. Link Training

Link training will be addressed in a future revision of the spec

Calibration Completion Signal	Meaning
tx_transfer_en	Transmit transmit block has completed calibration.
rx_transfer_en	Receive receive block has completed calibration

Table 17. Calibration Completion Signals

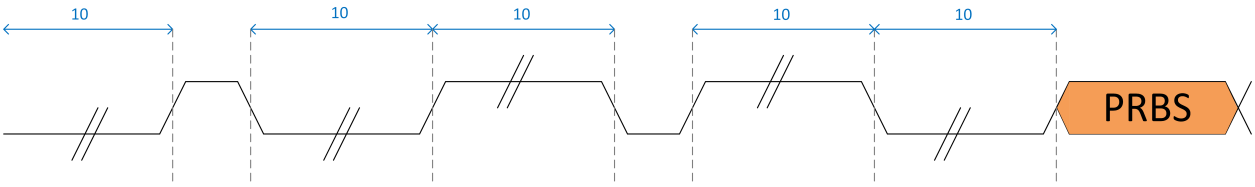


Figure 25. Stress Test Pattern

11. Control-Signal Shift Register Mapping

11.1. Transmit Control Register Definition

11.2. Receive Control Register Definition

12. Testability

12.1. Test Pattern

Suggested test patterns are:

- PRBS-9 Pattern, defined by polynomial of $X^9 + X^5 + 1$
- PRBS-31 Pattern, defined by polynomial of $X^{31} + X^{28} + 1$

Furthermore, to cover the DC wandering and single bit response, the following suggested pattern should be added to the beginning of the preferred PRBS pattern.

['0'] X 10 + '1' + ['0'] X 10 + ['1'] X 10 + '0' + ['1'] X 10 + ['0'] X 10

12.2. Loopback Test

A BoW interface will be used for loopback testing in two use cases: at wafer-sort time for chiplet test for full-system bring-up, and debug validation.

Wafer sort tests are currently only practical for the BoW interface with regular bump pitches

Bit Order	Control Signals from Transmit to Receive	Bit Width	Default Value	Description
[79]	Reserved	1	1	Reserved
[78]	tx_transfer_en	1	1	Transmit output to Receive to indicate that Transmit transfer has been enabled.
[77]	Reserved	1	1	Reserved
[76]	Reserved	1	1	Reserved
[75]	rx_transfer_en	1	1	Receive output to Transmit to indicate that Receive is ready for data transfer.
[74]	rx_dll_lock	1	1	Receive output to Transmit to indicate that RX DLL achieves lock.
[73:71]	Reserved	1	1	Reserved
[70:69]	Reserved	1	1	Reserved
[68]	tx_dcc_cal_done	1	1	Transmit output to Receive to indicate that DCC calibration is done
[67]	Reserved	1	0	Reserved
[66]	Reserved	1	1	Reserved
[65:27]	User defined	1	0	For application use
[26]	AUX_DBI	1	0	AUX-0, DBI -1
[25]	rx_adapter_rstn	1	0	Adapter reset signal from Transmit to Receive
[24]	tx_adapter_rstn	1	0	Adapter reset signal from Receive to Transmit
[23]	rx_mac_rdy	1	0	Data transfer ready signal from the Receive to the Transmit
[22]	tx_mac_rdy	1	0	Data Transfer ready signal from

Bit Order	Control Signals from Receive to Transmit	Bit Width	Default Value	Descripition
[71]	Reserved	1	0	Reserved
[70]	rx_transfer_en	1	1	Receive output to Transmit to indicate Receive Rx is ready to receive data
[69]	rx_dcc_dll_lock_req	1	1	DLL/DCC calibration request from Receive RX to Transmit TX BoW to start full DLL/DCC calibration.
[68]	rx_dll_lock	1	1	Receive output to Transmit (adapterand PHY) to indicate Receive DLL has achieved lock.
[67:65]	Reserved	1	0	Reserved
[64]	tx_transfer_en	1	1	Transmit sends to Receive (adapter and PHY) that it is ready for Trasnmit data transfer.
[63]	tx_dcc_dll_lock_req	1	1	PHY DLL/DCC calibration request from Receive TX to Transmit RX BoW to start full DLL/DCC calibration.
[62]	Reserved	1	0	Reserved
[61]	Reserved	1	0	Reserved
[60]	Reserved	1	1	Reserved
[59]	Reserved	1	0	Reserved
[58]	Reserved	1	1	Reserved
[57:50]	User defined	1	0	For application use
[49]	AUX_DBI	1	0	AUX-0, DBI -1
[48]	rx_adapter_rstn	1	0	Adapter reset signal from the Transmit to Receive
[47]	tx_adapter_rstn	1	049	Asynchronous adapter reset signal from the Receive to Transmit
[46]	rx_mac_rdy	1	0	Data transfer ready signal

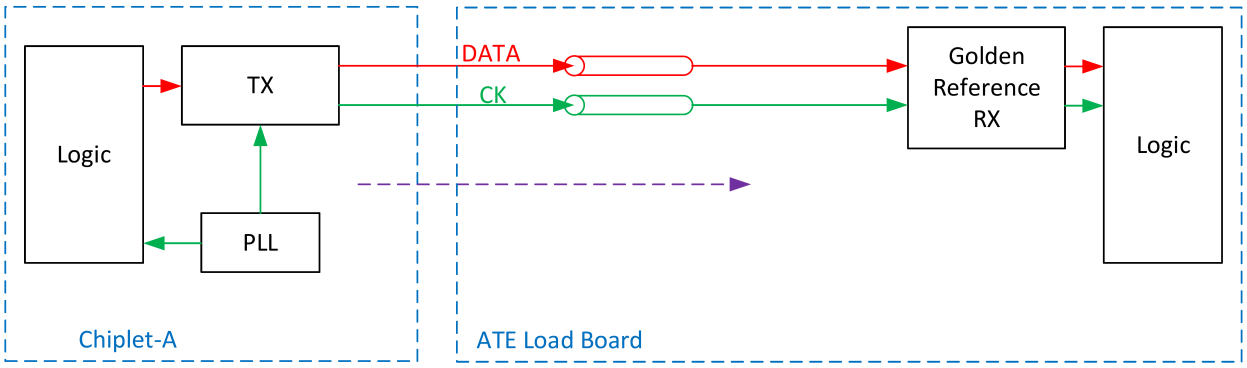


Figure 26. Open loop Tx chiplet testing

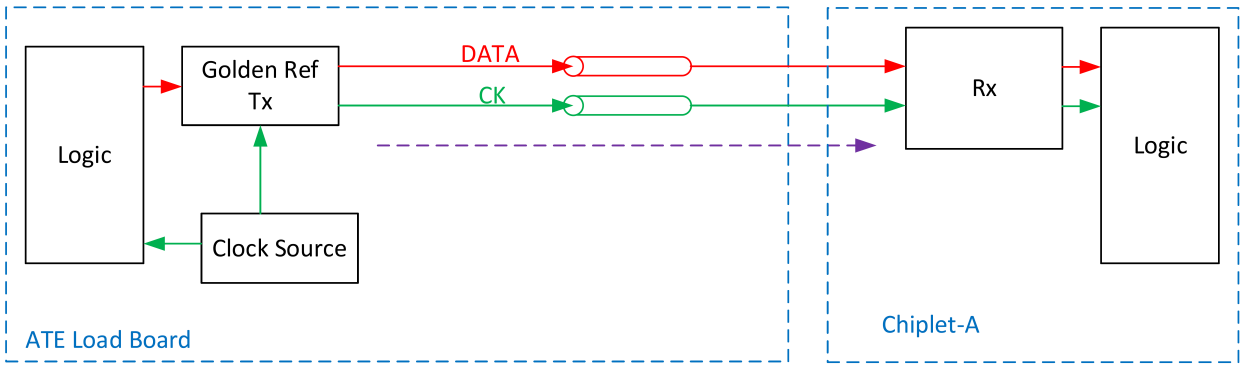


Figure 27. Open loop Rx chiplet testing

(~130um), where ATE (automatic testing equipment) probe boards with matching pin pitches are available. Microbump probes will require additional effort.

Unidirectional links will need open-loop testing. In Tx-Open-Loop testing, shown in Figure 26, Chiplet-A transmits a known test pattern (PRBS9 or PRBS31) to a golden reference receiver through the ATE load board. The received pattern is verified in the ATE load board.

Rx-Open-Loop testing, shown in Figure 27, is used for a link where the DUT is only a receiver. A golden reference Tx transmits a known pattern (PRBS9 or PRBS31) through the channel to the chiplet. The received pattern will be analyzed for quality and functional tests.

In bidirectional links, loopback tests can be implemented in two modes:

- short loopback mode: Data is looped back within the chip (shown in Figure 28). The short loopback can be triggered by the ATE.
- long loopback mode , the PRBS pattern is generated by chiplet-A, sent over the replica channel on the ATE load board which loops it back (shown in Figure 29). The received pattern will be passed to a bit error rate tester (BERT) to analyze the performance of the link with off-chip data and clock wires.

Both loopback modes can potentially be used for in-field validation bring-up and test. Co-operation across chiplets will be required to execute these tests in the field. Open-loop testing

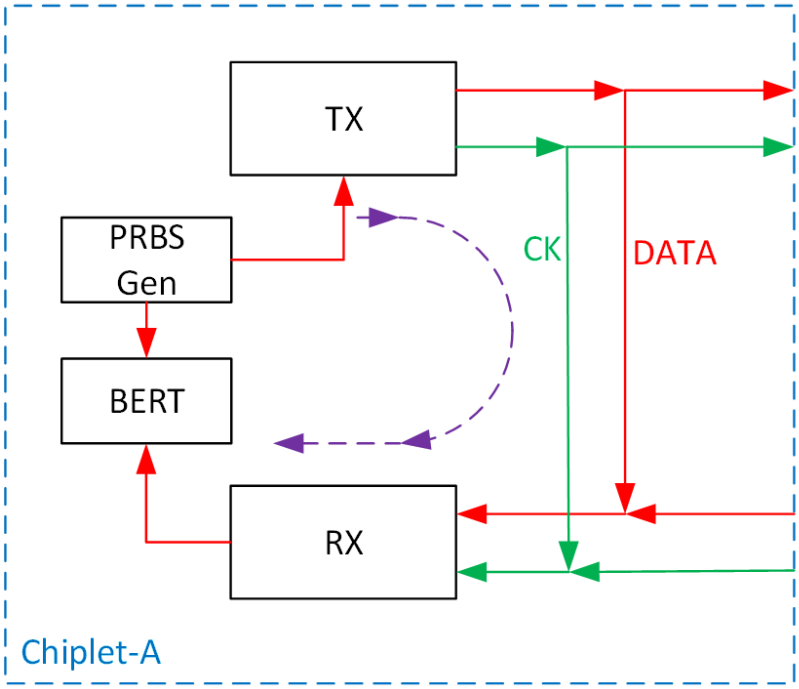


Figure 28. Short loopback testing

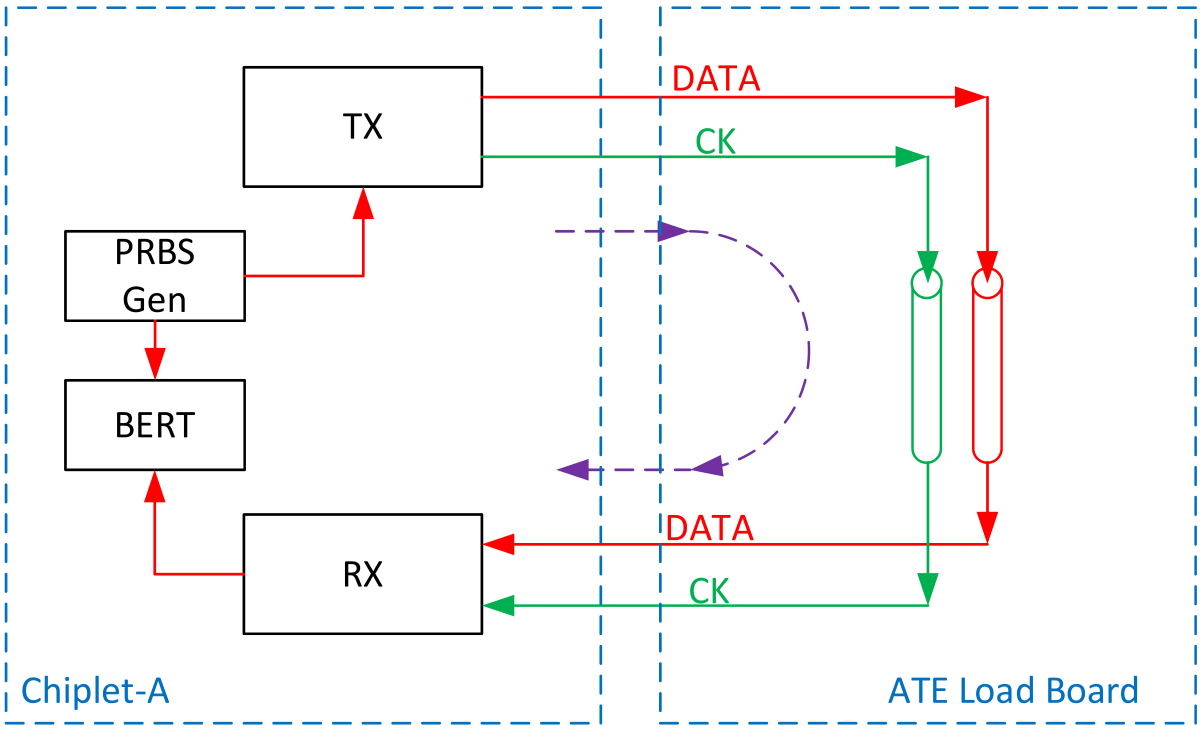


Figure 29. Long loopback testing

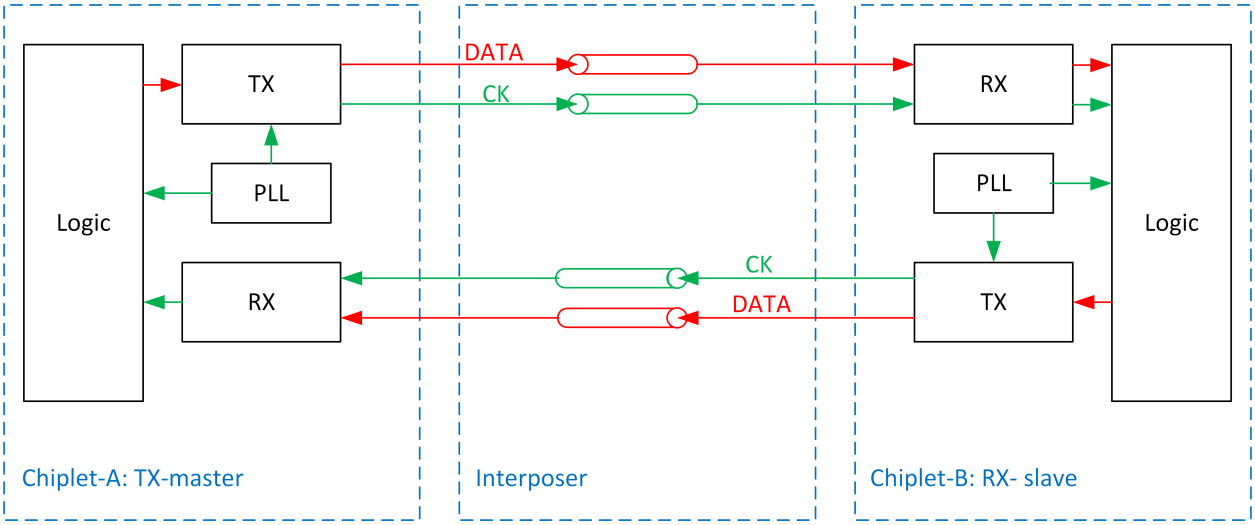


Figure 30. Chiplet-to-chiplet long loopback

requires the use of a fixed test pattern recognized by both ends and is the only option for unidirectional links. Long loopback mode can be implemented on interposer or organic laminate for validation/verification purposes.

Figure 30 shows how a long loopback mode is executed across two chiplets for in-field validation and test where Tx and Rx are in different chiplets. Furthermore, this configuration can be expanded to loop back the data from the transmitter of chiplet-A to the receiver of chiplet-A.