



OPEN

Compute Project

OpenHBI Specification Version 1.0

[Release Candidate - WG Approved 9/29/2021](#)

Authors:

Kenneth Ma, Principal Architect, Xilinx

Andy He, IP Lead, Google

Tom Wilson, Sr. Design Engineering Group Director, Cadence

Moo Sung Chae, Sr. Principal Design Engineer, Cadence

Jeffrey Bostak, Design Engineering Architect, Cadence

Peter Nyasulu, Principal R&D Engineer, Synopsys

Brian Worobey, Sr. Staff R&D Engineer, Synopsys

Bao Anh Nguyen, Sr. Manager, Synopsys

Millind Mittal, Fellow, Xilinx

Xiaobao Wang, Principal Engineer, Xilinx

Table of Contents

1.	License (OCP OpenHBI Specification)	7
1.1	Open Web Foundation (OWF) CLA	7
1.2	Acknowledgements	8
2.	OCP Tenets Compliance	9
2.1	Openness	9
2.2	Efficiency	9
2.3	Impact	9
2.4	Scale	9
3.	Revision Table	10
4.	Scope	11
4.1	In Scope	11
4.2	Out of Scope	11
5.	Overview	12
5.1	Key Features	13
5.2	Physical Orientation Convention	13
5.3	OpenHBI Interface Unit (DWORD or DW)	14
5.4	Scalable Bandwidth	14
5.5	Multi-layered Interface	14
5.6	OpenHBI V1.0 IO Electricals	16
6.	OpenHBI PHY Layer and Bump Map Description	17
6.1	OpenHBI DWORD and Virtual Bump Map	17
6.2	Signal Descriptions	21
6.3	OpenHBI PHY Layer	21
6.4	OpenHBI Scalability	24
6.5	Chiplet Configuration and Test (CCT) interface	27
7.	OpenHBI Logical PHY Layer	36
7.1	Bit Reordering for Rotated Die	37
7.2	Data Bus Inversion – Enhanced 9-bit DBI mode	37
7.3	Framing and Alignment	38
7.4	Parity	40
7.5	Options and Modes	41
7.6	Interface Pin Assignments per Mode	41
8.	Orientation and Routing	42
8.1	Orientation-Independent Routing	42
8.2	Same Orientation Routing – Bit Reordering Mode 0 (bypass, no reordering)	43
8.3	Rotated Orientation Routing – Bit Reordering Mode 1	44
8.4	WDQS (TX_Clk) and RDQS (RX_Clk) Clock Lane Repair	49

9.	OpenHBI IO Electricals	51
9.1	Transmitter Electrical Parameters	51
9.2	Receiver Electrical Parameters	52
9.3	REFERENCE for Transceiver and Channel	53
9.4	Channel Parameters	53
9.5	REFCLK Specifications	54
9.6	BER spec.....	54
10.	Clocking, Initialization and Configurations.....	55
10.1	Clocking.....	55
10.2	Initialization Flow.....	58
10.3	Capability, Control and Status (CCS) Registers	59
10.4	MISR/LFSR Test Mode.....	65
10.5	Training and Calibration	65
10.6	Continuous Training during operation.....	68
11.	Power Management.....	70
12.	References	71
Appendix A - Requirements for IC Approval (to be completed Contributor(s) of this Spec)		72
Appendix B-_____ - OCP Supplier Information (to be provided by each Supplier of Product)		73

Glossary of Terms

This section provides glossary used in this specification. Note that it is not organized in alphabetical order but in sequential order to best understand the terms and definitions of OpenHBI.

Abbreviation	Term	Description
OpenHBI	Open High Bandwidth Interface	An optimized chiplet interconnect which is compatible with DWORD channel and interoperable with HBM3 (Reference[1]) electricals with highly optimized Figures of Merit (performance, bandwidth density, energy/bit and cost), a standard developed via ODSA OpenHBI Workstream.
OpenHBI V1.0	OpenHBI Version 1.0	Name of this specification. OpenHBI V1.0 provides the specification of the features in-scope of OpenHBI for this version.
DWORD or DW	Data Word	Each DWORD consists of up to 42 data-carrying signals, RX clock pair, TX clock pair and two lane repair RD0/RD1. DWORD and DW will be used interchangeably in this specification.
Beachfront or BF	Beachfront	The length along the die edge that is occupied by the OpenHBI C2C interface
C2C, D2D, Chiplet interface	Chip-to-Chip Die-to-Die Chiplet Interface	The C2C, D2D and Chiplet interface terms will be used interchangeably in the OpenHBI specification depending on the contexts.
ubump	Microbump	Microbump
Y-pitch	Y-pitch	OpenHBI ubump-to-ubump pitch along the beachfront
X-pitch	X-pitch	OpenHBI ubump-to-ubump pitch towards the center of the anchor or chiplet die (perpendicular to the beachfront)
Anchor	Anchor die	Anchor die refers to a chiplet that can provide configuration, management and test services and/or can support multiple chiplets on one or more of its die edges. The notion of Anchor is optional if it uses an external controller to provide the configuration and management services.
PHY	Physical Layer	The low-level hardware interface of a high-speed input/output protocol
Logical PHY	Logical PHY layer	A thin layer above PHY layer to provide optional functions and services including Parity, Framing, DBI and Bit Reordering.
Link Layer	Link Layer	Link layer is a layer above Logical PHY layer. It supports optional DWORD aggregation function, flow control and data integrity and reliability services.
TPA interface	TransPort Abstraction (TPA) interface	TPA abstracts the C2C transport. C2C transport includes the PHY layer, Logical PHY and Link Layer.
TPA adaptation	TPA adaptation layer	TPA adaptation layer adapts the OpenHBI C2C transport to the TPA interface for mapping to the protocol layer(s).

layer		
DBI	Data Bus Inversion	Data Bus Inversion is an optional normative service to minimize signal toggling on the physical wire. If the DBI service is enabled, it counts the number of data transitions to ensure no more than 50% of the data wires (including DBI signal) will toggle to reduce power and improve signal integrity.
CCS Registers	Capability, Control & Status Registers	A set of reference Capability, Control and Status Registers.
JEDEC HBM	JEDEC High Bandwidth Memory specification	JEDEC HBM standard is a wide, interposer-based chiplet interface between an SoC and HBM memory which is a 3D-stacked DRAM device.
JEDEC HBM3	HBM3 Specification	JEDEC HBM3 standard specification. Reference [1]
Gearbox		Speed/width conversion function in a SERDES
SERDES		SERializer - DESerializer
TX	Transmit	Prefix for Transmit-side; e.g., TX_Clk, TX DWORD
RX	Receive	Prefix for Receive-side; e.g., RX_Clk, RX DWORD
UI	Unit Interval	The time period of a single data bit on the interface bus (= $\frac{1}{2}$ of TX_Clk / RX_Clk period)
Gbps or Gb/s		Gigabits per second – unit of data rate
Tbps or Tb/s		Terabits per second – unit of data rate
Gbps/mm		Gigabits per second per mm – unit of bandwidth density
Tbps/mm		Terabits per second per mm – unit of bandwidth density
IO or IOs	Input / Outputs	Input receiver and Output driver
pJ/bit or pJ/b		Picojoule / bit – unit of energy per bit transferred
PDN	Power Delivery Network	The power delivery network is the current path from the power supply (or power supplies) through packaging substrate to the silicon die.
PPA		Power, Performance and Area. This term describes some key metrics in silicon or IP design considerations.

Language

OpenHBI specification adopts the use of the words “shall”, “must”, “should”, “may” and “can” as defined below:

Term	Description
“shall” or “must”	It indicates a normative requirement strictly to be followed. Failure to meet the requirement results in non-conformance.
“should”	It indicates a strong suggestion or recommendation, but not a requirement. Failure to implement the suggestion does not result in non-conformance.
“may”	It indicates “it is permitted to” implement or take a course of action.
“can”	It indicates “it is able to” or used as a statement of possibility and capability.

1. License (OCP OpenHBI Specification)

1.1 Open Web Foundation (OWF) CLA

Contributions to this Specification are made under the terms and conditions set forth in the **modified** Open Web Foundation Contributor License Agreement (“OWF CLA 1.0”) (“Contribution License”) by:

ARM, CADENCE, GOOGLE, SIFIVE, SYNOPSYS, TERADYNE, XILINX

You can review the signed copies of the applicable Contributor License(s) for this Specification on the OCP website at <http://www.opencompute.org/products/specsanddesign>

Usage of this Specification is governed by the terms and conditions set forth in the **modified Open Web Foundation Final Specification Agreement (“OWFa 1.0”) (“Specification License”)**.

Notes:

1) The following clarifications, which distinguish technology licensed in the Contribution License and/or Specification License from those technologies merely referenced (but not licensed), were accepted by the Incubation Committee of the OCP:

“None”

2) The above license does not apply to the Appendix or Appendices. The information in the Appendix or Appendices is for reference only and non-normative in nature.

NOTWITHSTANDING THE FOREGOING LICENSES, THIS SPECIFICATION IS PROVIDED BY OCP "AS IS" AND OCP EXPRESSLY DISCLAIMS ANY WARRANTIES (EXPRESS, IMPLIED, OR OTHERWISE), INCLUDING IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR A PARTICULAR PURPOSE, OR TITLE, RELATED TO THE SPECIFICATION. NOTICE IS HEREBY GIVEN, THAT OTHER RIGHTS NOT GRANTED AS SET FORTH ABOVE, INCLUDING WITHOUT LIMITATION, RIGHTS OF THIRD PARTIES WHO DID NOT EXECUTE THE ABOVE LICENSES, MAY BE IMPLICATED BY THE IMPLEMENTATION OF OR COMPLIANCE WITH THIS SPECIFICATION. OCP IS NOT RESPONSIBLE FOR IDENTIFYING RIGHTS FOR WHICH A LICENSE MAY BE REQUIRED IN ORDER TO IMPLEMENT THIS SPECIFICATION. THE ENTIRE RISK AS TO IMPLEMENTING OR OTHERWISE USING THE SPECIFICATION IS ASSUMED BY YOU. IN NO EVENT WILL OCP BE LIABLE TO YOU FOR ANY MONETARY DAMAGES WITH RESPECT TO ANY CLAIMS RELATED TO, OR ARISING OUT OF YOUR USE OF THIS SPECIFICATION, INCLUDING BUT NOT LIMITED TO ANY LIABILITY FOR LOST PROFITS OR ANY CONSEQUENTIAL, INCIDENTAL, INDIRECT, SPECIAL OR PUNITIVE DAMAGES OF ANY CHARACTER FROM ANY CAUSES OF ACTION OF ANY KIND WITH RESPECT TO THIS SPECIFICATION, WHETHER BASED ON BREACH OF CONTRACT, TORT (INCLUDING NEGLIGENCE), OR OTHERWISE, AND EVEN IF OCP HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.2 Acknowledgements

The Contributors of this Specification would like to acknowledge the following companies and members for their feedback (list in alphabetical order of the member companies) :

Javier DeLaCruz, Sr. Director, Arm

Marc Greenberg, Product Marketing Group Director, Cadence

Sue Hung Fung, Lead Technical Marketing Engineer, Cadence

Juan-Carlos Calderon, Chief SoC IP Architect, SiFive

Khanh Pham, Manager, Synopsys

Octavian Beldiman, Sr. Manager, Synopsys

Loc Huynh, Sr. II. – AMD Circuit Design Engineer, Synopsys

Marc Hutner, Sr. System Architect, Teradyne

Chadi Barakat, Sr. Director. Silicon IP Design Engineering, Xilinx

Arash Izadi, Staff IC Design Engineer, Xilinx

2. OCP Tenets Compliance

2.1 Openness

The OpenHBI V1.0 specification is the first version of OpenHBI developed to build an open, public and interoperable OpenHBI IP and Chiplet ecosystem. All members agreed to grant the Necessary Claims by executing the modified OWFa 1.0 FSA to enable user implementation of the OpenHBI spec.

With chiplet trend still in its infancy, the spec allows choice of fine-pitch packaging technologies to match the use cases and cost target. The spec has provisions to facilitate cross-vendor interop with Virtual Bump Map concept, a well-defined CCT (Chiplet Config & Test) interface to enable wafer level tests for KGD (known good die) and post-packaged final test utilizing standard JTAG and MIPI I3C as config and management interfaces.

2.2 Efficiency

An OpenHBI V1.0 Full Instance could deliver up to > 10Tbps bandwidth in as little as 4.48mm beachfront and up to 2.29Tbps/mm bandwidth density. OpenHBI adopts highly efficient 0.4V low swing IO. A typical “N-over-N” IO structure and Phy layer design would achieve state-of-the-art < 0.4pJ/bit transferred and a nominal latency of < 4ns which is important to enable chiplet-based design to mimic a monolithic design.

2.3 Impact

OpenHBI V1.0 is the first OCP Chiplet interface specification. It is the first such standard with provisions for pre-packaged wafer-level tests for KGD and post-packaged anchor-chiplet final tests. It sets a new bar for best-in-class bandwidth density, energy efficiency and latency. The spec combines the virtual bump map concept, flexibility in packaging tech and bump pitches, and orientation-independent beachfront maximizes cross-vendor interoperability.

2.4 Scale

OpenHBI V1.0 is scalable. Each Full Instance supports up to > 10Tbps bandwidth. The beachfront bandwidth scales linearly with multiple OpenHBI Instances which is only limited by size of the die edge.

The spec supports heterogenous Anchor-to-Chiplet and homogenous peer-to-peer Anchor-to-Anchor and Chiplet-to-Chiplet use cases to create high performance compute or accelerator clusters, for example.

3. Revision Table

Date	Revision #	Author	Description
12/18/2020	0.5	Kenneth Ma	Complete Chapter 1-6 (previous template) Changed to TX_Clk clock edges are edge-aligned (phase aligned) relative to the TX data Added “virtual” Bump Map target Added Framing and Alignment details Added Bit Reordering w/ Lane Repair modes and details Added IO Electrical template Added informative Clocking architecture options First draft of Initialization, training and calibration flow
6/30/2021	0.7	OHBI WG	DWORD Virtual Bump Map (VBM) <ul style="list-style-type: none"> - Add True & R180 VBM example diagrams - Add example of potential GDS reuse OpenHBI Scalability <ul style="list-style-type: none"> - Add notions of “Full”, “Half”, “Quarter” instances - 4DW-deep Anchor shall support 2-deep Chiplets - Interop example diagrams - Y-pitch, DWORD Height Min-Max range CCT <ul style="list-style-type: none"> - Updated CCT function, probe pad sharing - MIPI I3C Basic as (normative) Config Interface - CCT signals + VS (Vendor-Specific signals) - CCT signals redundancy and lane repair support - CCT Association on edge with “multi-instances” - CCT size guidelines (Informative) - Optional Dummy CCT Enhanced 9-bit DBI Add “DBI-enabled, No framing/parity” 38-data mode Updated IO Electrical parameters and RefClk BER (Bit Error Ratio) targets 1e-25 Initialization Flow, Loop Back Tests Training and Calibration Capability/Control/Status Registers (Informative)
9/15/2021	0.9	OHBI WG	Ported to new OCP’2021 template. General editorial edits. Change to I3C but allows the option to use I3C Basic CCT 1.2V IO, I3C VE-CCC, IBI with MDB CCT adds PwrGood. Dedicated JTAG_TRST and VS15 Remove VS[18:17], VS[16]. Updated Lane Repair table Section 6.5.7: CCT “Virtual Bump Map” target Chiplet PAR error shall generate I3C IBI to notify Anchor Section 10: Added I3C BCR/DCR Registers. Updated Initialization Flow, tINIT1, tINIT2, Training and Calibration, Broken Lane Detection flow and diagrams. Changed terms to Configuration Manager/Subordinate.
9/29/2021	Version 1.0	OHBI WG	WG unanimously approved Rev 0.9 draft to become OpenHBI Version 1.0 Release Candidate on 9/29/2021

4. Scope

4.1 In Scope

This document defines the technical specifications for the OpenHBI V1.0 under OCP Server project, ODSA subproject. Scope of OpenHBI V1.0 includes but not limited to the following:

- DWORD and Bump signals definitions
- PHY layer
- Logical PHY layer
- IO electricals
- Reference DWORD “Virtual Bump Map”
- Y-pitch and DWORD height (Min-Max range)
- Beachfront Scalability
- Routing and Orientation considerations
- Chiplet Configuration and Test (CCT) interface
- Chiplet Wafer sort/testing

Please also refer to Section 5.5, Figure 5-3 which illustrates the scope of the OpenHBI V1.0 specification with respect to the typical ODSA stack.

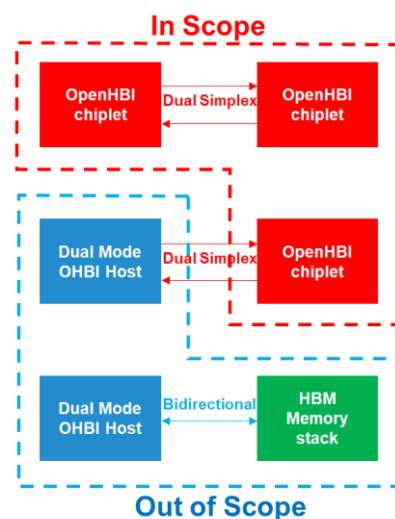


Figure 4-1: Scope of OpenHBI

4.2 Out of Scope

- Dual-mode Host that can support both OpenHBI-conformant chiplets and JEDEC HBM3 memory.
- Physical media used for OpenHBI V1.0
- Data rates great than 8Gbps
- Upper layer definition
- Use of OpenHBI V1.0 outside of a package
- Security features such as Authentication, Integrity Check, Encryption

5. Overview

The ODSA OpenHBI or Open High Bandwidth Interface specification defines a high-performance die-to-die or chiplet-to-chiplet interconnect on a medium such as a silicon interposer or equivalent technology in a common package.

(The terms “chip-to-chip”, “die-to-die” and “chiplet-to-chiplet”, C2C and D2D will be used interchangeably in the rest of this specification. Chiplet or chiplet, Anchor or anchor will also be used interchangeably.)

The OpenHBI defines an optimized C2C interface specification which is compatible with DWORD channel and interoperable with HBM3 (Reference[1]) electricals with highly optimized Figures of Merit. It is intended for datacenter, HPC, AI, networking and other applications that demands high performance, high bandwidth density, low latency and energy efficient (low pJ/bit) C2C interconnect.

OpenHBI V1.0 IO electrical is defined to deliver reliable signal integrity with very low BER (Bit Error Ratio) interconnect, target < 1e-25. It has provision for interoperability, manufacturability and testability between heterogenous chiplets, especially for chiplets from different vendors, at co-packaging level.

OpenHBI V1.0 supports data rates up to 8Gbps. Future versions of OpenHBI specification will further increase data rates, bandwidth density and functional features to meet increasing performance demands, with backward compatibility.

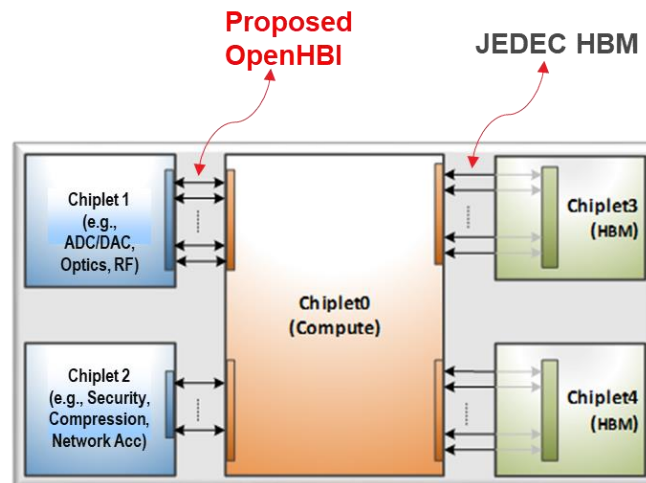


Figure 5-1: Example OpenHBI chiplet-based device

OpenHBI defines a symmetric, multi-channel die-to-die parallel interface between OpenHBI chiplets on an interposer which can be a silicon interposer or equivalent technologies on a common package. Below describes its key features and attributes.

5.1 Key Features

- Optimized IO electricals defined to deliver a reliable link (< 1 FIT) with raw target BER < 1e-25
- Supports silicon interposer, wafer-level integrated fan-out or equivalent technologies
 - For example, TSMC CoWoS™, InFO™ or fine pitch organic substrate
- Symmetric, multi-channel chip-to-chip interface
- Speed target of 8Gbps per pin
- Up to 3mm trace reach at max data rate
- Power target $\leq 0.4\text{pJ/bit}$
- Linear (beachfront) bandwidth density > 1.5Tbps (up to 2.28Tbps) per mm (TX+RX bandwidth)
- Orientation-Independent routing;
 - Support interface with normal and rotated die orientation
- Scalable;
 - Bandwidth and beachfront (number of DWORDS) to match various use cases.
 - Provision for multiple chiplets per edge on Anchor chiplet.
- Chiplet Configuration and Test interface
- Lane Repair support to enhance manufacturing yield in either orientation

5.2 Physical Orientation Convention

This specification allows OpenHBI interfaces along any one of the 4 die edges of an Anchor, referred to as Top, Right, Bottom and Left edges as illustrated in the example in Figure 5-2.

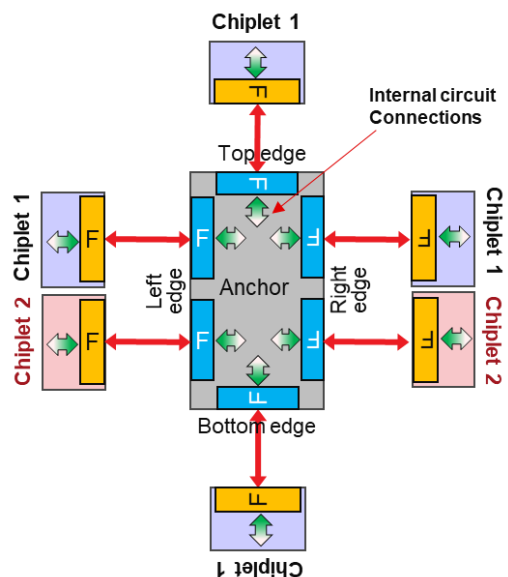


Figure 5-2: Example Anchor with OpenHBI chiplets on Top, Right, Bottom and Left edges

OpenHBI specification defines the beachfront bump pattern shall (Normative) rotate 90° clockwise when traversing the Top → Right → Bottom → Left edges of an Anchor.

Implementation Note: Anchor may optionally implement mirrored instead of rotated 180° beachfront bump pattern between Top and Bottom edges and between Right and Left edges. Such implementation must provide additional implementation-specific Bit-reordering functions similar to the concept described in the Section 7 and 8 to enable seamless routing with chiplets that are compliant to this specification, on any of the Anchor die edge and orientation (either rotated 180° or mirrored beachfront) .

For clarity of the specification, the orientation descriptions and examples refer to a side-by-side chiplet orientation with OpenHBI signals flowing horizontally between chiplets. Unless otherwise noted (and without loss of generality), the descriptions and examples will apply to any one of the 4 die edges.

5.3 OpenHBI Interface Unit (DWORD or DW)

The basic OpenHBI communication unit is a 48-signal group, referred to as a 'data word', **DWORD** or **DW** for short. A DWORD consists of 42 data wires, 4 clock/strobe wires, and 2 redundant wires for lane repair.

A DWORD may support either configurable (bidirectional) or fixed (TX or RX) directionality. A typical bidirectional DWORD is configurable at initialization time to operate as TX or RX DWORD. An Anchor shall implement configurable bidirectional DWORDs unless it is designed to only support chiplets of known and fixed configurations. A chiplet may implement fixed direction PHY layer and DWORD interface units only if it has a known configuration and/or it will be connected to Anchor with configurable bidirectional DWORDs.

The DWORD structure is symmetric in the sense that there is no difference in its design between different chiplet roles (i.e., no notion of 'Manager' or 'Subordinate'). Each DWORD can be configured at boot time as output (TX) or input (RX), and the choice applies to all 42 DWORD data signals and the 2 redundant wires.

In operation, each DWORD is unidirectional. A chiplet TX DWORD transmitting data will drive the data signals and the differential transmit clock pair TX_Clk to the chiplet on the receiving side. A chiplet receiving data will receive the data signals and will accept the clock pair via the receive clock inputs RX_Clk. Clock edges shall be edge-aligned with data signals (D[41:0] and RD[1:0]) relative to the data eye.

In this specification, the TX_Clk and WDQS_t/_c will be used interchangeably, also the RX_Clk and RDQS_t/_c will be used interchangeably.

5.4 Scalable Bandwidth

OpenHBI interface bandwidth can be scaled up by using multiple DWORD units. Please refer to Section 6.4, OpenHBI Scalability for more details.

5.5 Multi-layered Interface

To enable flexible and modular architecture, OpenHBI adopts a reference multi-layered interface model compatible with ODSA stack concept as shown in Figure 5-3.

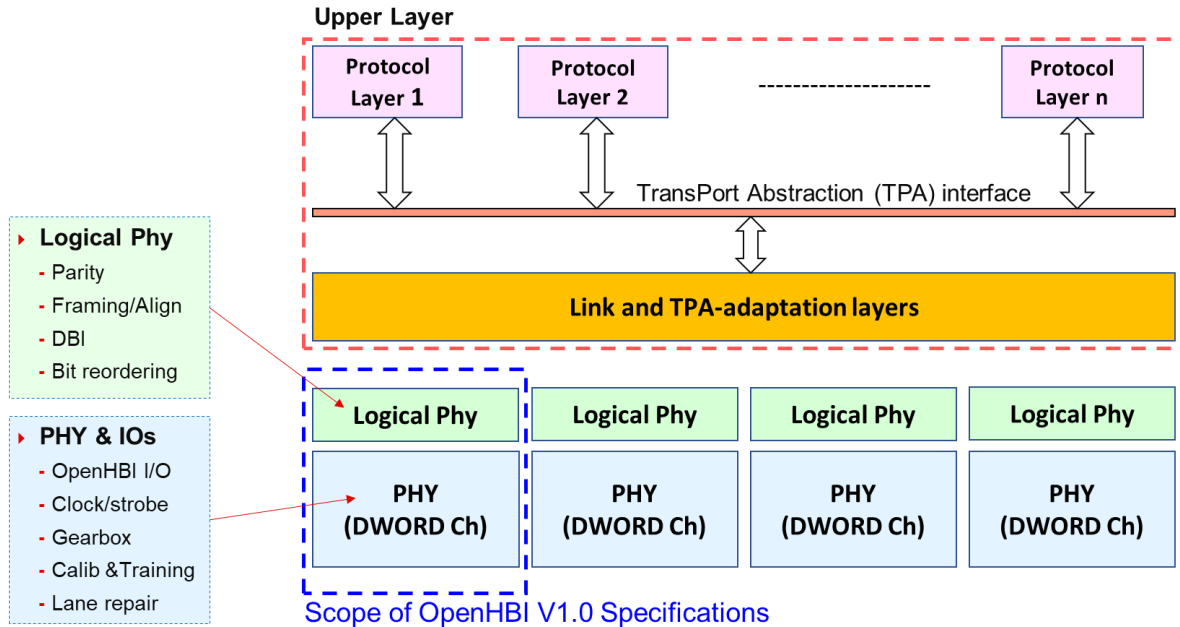


Figure 5-3: OpenHBI Reference Multi-Layered Model

The OpenHBI V1.0 defines two layers, PHY and Logical PHY layers:

5.5.1 PHY Layer: This layer defines OpenHBI PHY which performs the transmit, receive, clocking, serializing/deserializing (aka Gearbox) functions during operations, and the clock-to-data training functions during initialization. PHY layer also performs Lane Repair for defective signal connection within a DWORD.

5.5.2 Logical PHY Layer: The OpenHBI Logical PHY layer supports optional-normative functions such as bit reordering, DBI (data bus inversion), framing and alignment, parity generation/check. The bit reordering function is to ease the routing of two OpenHBI chiplets in 180° rotated ("R180") orientation.

The OpenHBI C2C transport can support various high-level protocols (e.g., PCIe, CCIX/CXL, AXI) via the "Upper Layer" which may consist of Link, TPA-adaptation and Protocol adapter layers.

The Link layer supports optional DWORD aggregation function, flow control and data integrity and reliability services.

The TPA adaptation layer adapts the OpenHBI C2C transport to the TransPort Abstraction (TPA) interface for mapping to the protocol layer(s).

Figure 5-4 illustrates how OpenHBI Anchor and chiplets operations in layered architecture.

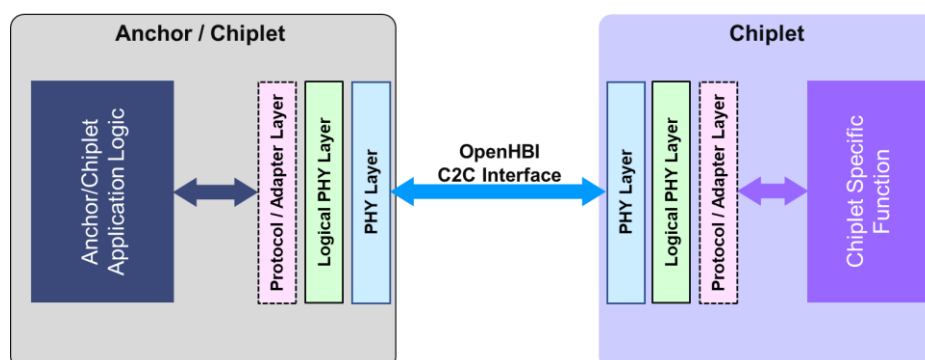


Figure 5-4: OpenHBI Anchor and chiplet layered architecture

Different function chiplets may implement the layers only required by their functions. The minimum set of layers required for normal operation are PHY and Logical PHY layers. It is therefore desirable to optimize the design, complexity, latency, and PPA (Power, Performance and Area) of these 2 layers.

5.6 OpenHBI V1.0 IO Electricals

OpenHBI V1.0 IO Electrical section defines the IO swing, Transmitter, Receiver and Channel electrical and jitter parameters, Pin cap and other key attributes to achieve the target BER at the target data rate.

It supports DDR signaling with source synchronous clocking. It adopts single-ended data signals and differential clocks (TX_Clk and RX_Clk). The IO is electrically interoperable with HBM3 IO at the physical interface level. Refer to Section 9 for detailed IO Electrical definitions.

6. OpenHBI PHY Layer and Bump Map Description

6.1 OpenHBI DWORD and Virtual Bump Map

OpenHBI specification defines a (normative) reference DWORD “virtual” Bump Map (aka Bump map or Bump pattern). An OpenHBI Anchor or chiplet does not need to follow an exact physical microbump pattern so long as it is designed in a way that it can route efficiently treating the reference OpenHBI Bump Map as a “virtual target” and meet the timing and SI on the target interposer or substrate media.

Each DWORD “virtual” bump map has 14 rows by 12 columns of ubumps, called Standard Matrix, in checkerboard arrangement containing 48 signals and ubumps for one or more power rails and ground rail as shown in the Figure 6-1:

- 42-bit Data (D[41:0]) – “yellow” cells
- RD0 and RD1 (Redundant signals for Lane Repair) – “pink” cells
- Differential WDQS_t/c output clock pair (also referred to as “TX_Clk”) – “teal” cells
- Differential RDQS_t/c input clock pair (also referred to as “RX_Clk”) – “dark green” cells

The unlabeled “light green” cells are typically used for power delivery network (PDN) including GND and one or more power rails. Since the power rails between Anchor and Chiplets may have different voltage levels and are not connected via interposer traces, designers may use these ubumps for implementation-specific PDN as long as it meets the SI/PI (signal integrity/power integrity) requirements at the target data rate.

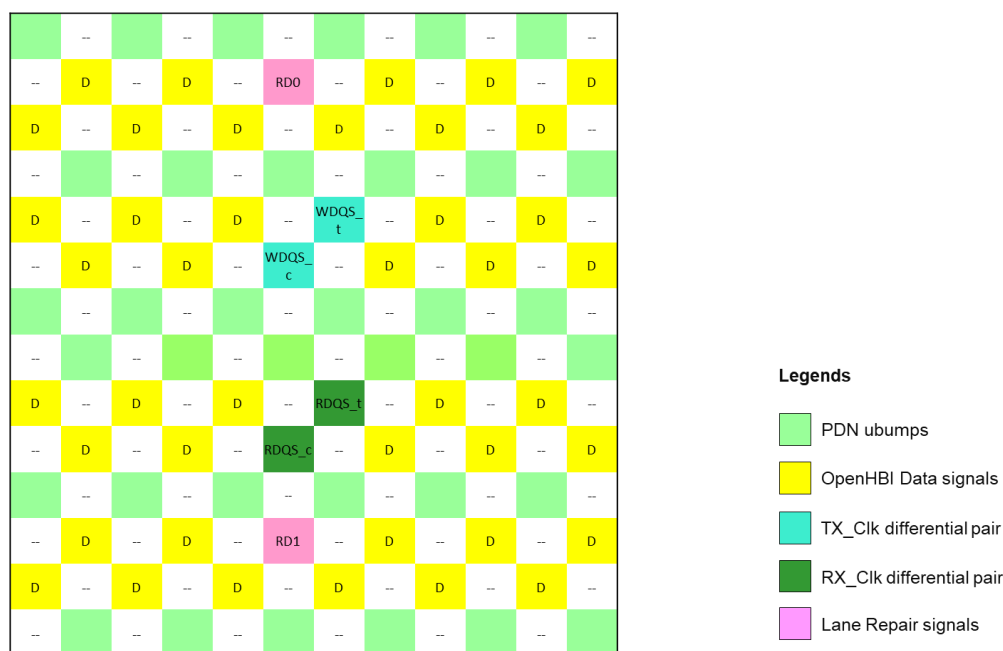


Figure 6-1: OpenHBI DWORD Virtual Bump Map concept – Standard

There is an alternative (normative) “Compact Matrix” virtual bump map. Please refer to Section 6.4 for more details.

In subsequent sections, the terms “True” and “R180” used are defined as below:

- Chiplet “True” orientation DWORD is the native bump map of Chiplet beachfront facing Right
- Chiplet “R180” orientation DWORD is the native bump map of Chiplet beachfront facing Left
- Anchor “True” orientation DWORD is the native bump map of Anchor beachfront facing Left
- Anchor “R180” orientation DWORD is the native bump map of Anchor beachfront facing Right

Figure 6-2 defines OpenHBI Chiplet “True” orientation DWORD virtual bump map facing the right edge. The light orange, pink and purple colored cells illustrate those are OpenHBI Data[x] or D[x] that are non-data signals in Reference [1].

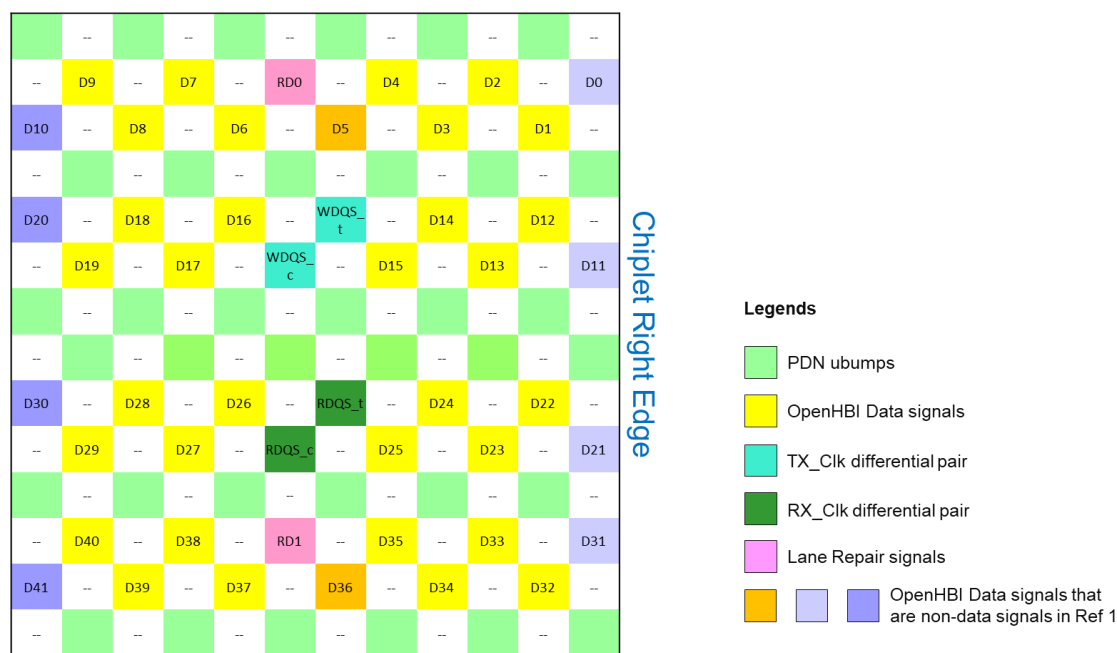


Figure 6-2: Chiplet "True" DWORD Virtual Bump Map (facing Chiplet Right Edge)

Refer to Section 8, the bump patterns have the notions of “True” and “R180” orientations. Note the ubump locations relative to the Chiplet and Anchor die edges below.

- Chiplet “True” and Anchor “True” DWORD bump patterns – see Figure 6-3
- Chiplet “R180” and Anchor “R180” DWORD bump pattern – see Figure 6-4

In the example shown in Figure 6-3, when both Chiplet and Anchor are in “True” orientation facing each other, this is referred to as “SAME ORIENTATION”. Similarly, in example shown in Figure 6-4, if both Chiplet and Anchor are in “R180” orientation facing each other, they are also in “SAME ORIENTATION”.

As the ubumps for PDN are implementation-specific, in Figure 6-3 and Figure 6-4, the “light green” PDN ubumps between on Chiplet and Anchor sides do not need to correspond to each other.

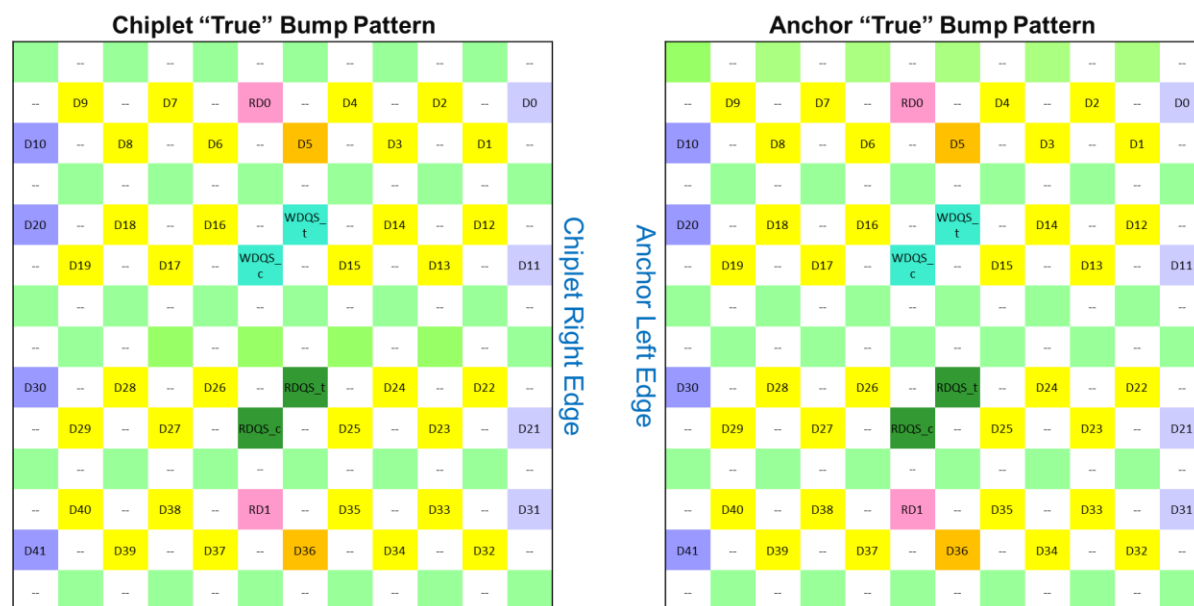


Figure 6-3: Chiplet "True" and Anchor "True" Bump Pattern: SAME ORIENTATION

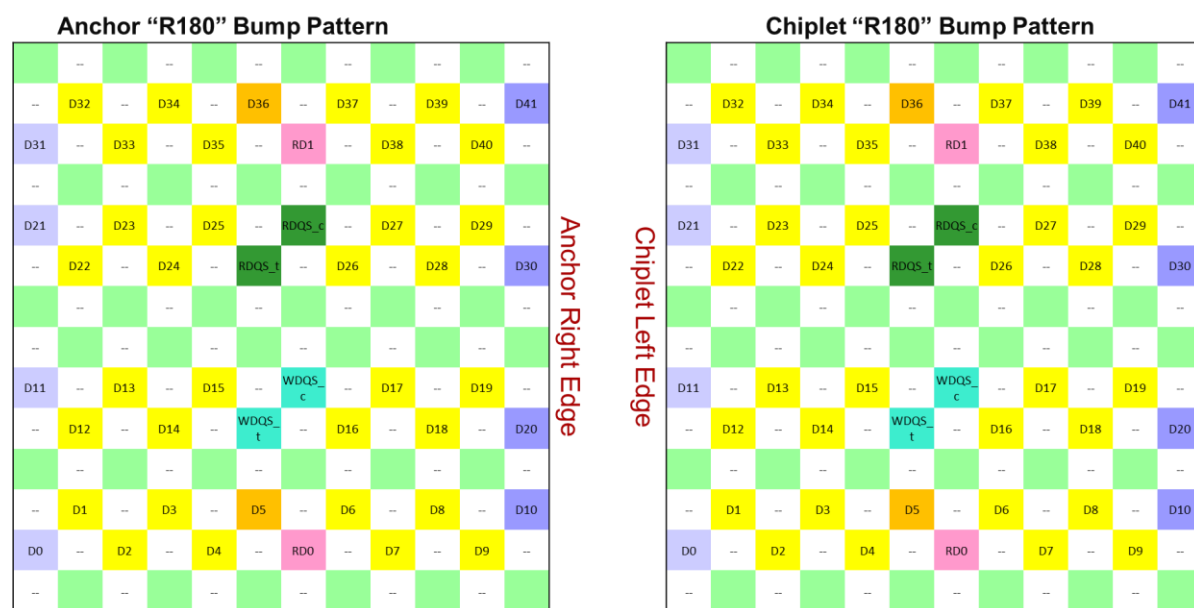


Figure 6-4: Anchor "R180" and Chiplet "R180" Bump Pattern: SAME ORIENTATION

If a Chiplet is in “R180” and an Anchor is in “True” orientation, or vice versa, they are referred to as “ROTATED ORIENTATION”, as shown in the example in Figure 6-5.

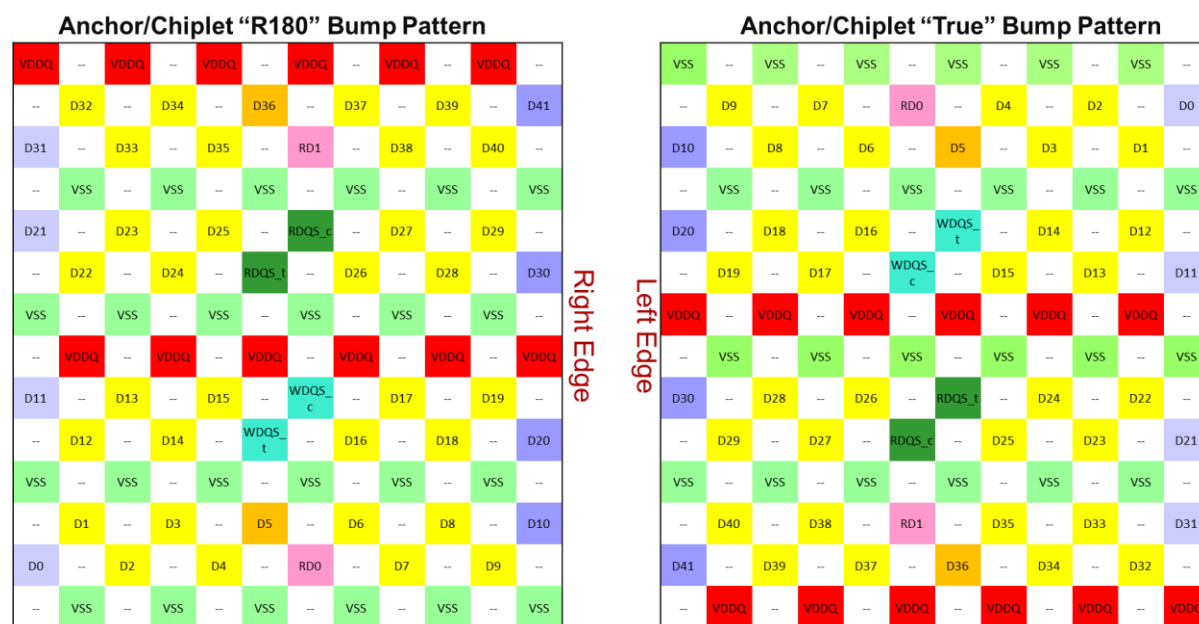


Figure 6-5: Example "ROTATED ORIENTATION" between Anchor and Chiplet DWORD bump pattern

Implementation Note: The example in Figure 6-5 illustrates the possibility to use a single GDS on both Anchor and Chiplet or in Anchor-to-Anchor (homogeneous peer-to-peer) use case where the DWORD bump patterns on two sides, including signals and PDN ubumps, are rotated 180° and in ROTATED ORIENTATION. Please refer to Section 8 on OpenHBI Bit-reordering function that enables seamless and straight routes in both SAME and ROTATED orientations.

6.2 Signal Descriptions

Table 6-1 describes OpenHBI DWORD signals and their high-level functions.

Table 6-1: OpenHBI DWORD Signals

Symbol	OpenHBI functions	TX-side	RX-side	Transport Data
D[41:0]	Data signals	Out	In	Yes
TX_Clk	TX_Clk / Differential TX clock pair	Out	Not used	No
RX_Clk	RX_Clk / Differential RX clock pair	Not used	In	No
RD0, RD1	Redundant 0 / 1 for lane repair	Out	In	Yes

D[41:0] are data signals that nominally carries data between two chiplets / dice.

TX_Clk (aka WDQS_t/_c) is differential output clock pair on TX side.

RX_Clk (aka RDQS_t/_c) is differential input clock pair on the RX side.

The DWORD direction is configurable to be TX or RX either pre-configured, during initialization, or optionally during run time, via configuration bus.

In normal configuration, TX_Clk differential output clock pair on TX side drives the RX_Clk differential input clock pair on the RX side.

The RX_Clk pair on TX side, and TX_Clk pair on RX side are not used typically and may be left as “No Connect” as long as the RX_Clk pair inputs on TX side are pulled to proper valid state and not floated.

6.3 OpenHBI PHY Layer

As shown in Figure 6-6, the PHY layer contains the TX_Clk and RX_Clk pairs, up to 42 data signals and two Lane Repair signals and it performs the following functions:

- Speed/width conversion function (aka Gearbox function)
- Calibration and training function
- Lane repair function (if broken lane is detected)
- Optional “Signal Swap” functions. (Refer to Section 8 for details.)
 - WDQS (TX_Clk) <> RDQS (RX_Clk) clock pairs swap
 - _t <> _c (true<>complement) differential polarity swap of WDQS and RDQS
 - RD0 <> D[5] swap and RD1 <> D[36] swap

The redundant signals RD[1:0] will be “consumed” by the PHY layer. If there are defective connections that are repairable, the PHY layer will perform lane remapping as described in Section 6.3.5. If there is no defective connection, the RD[1:0] will be left unused and will not be passed to the layer above PHY layer.

As the OpenHBI IO data rate (e.g., Phy_Bitrate = 8Gbps) can be substantially higher than the Logical PHY layer or core logic operating frequency (e.g., 1GHz), the PHY layer can perform R:1 serialize / deserialize (Gearbox) functions and convert the frequency between two domains, where the ratio $R = \text{Phy_Bitrate} / \text{LogPhy_Clk}$ and must be an 2^n integer value.

The number of signals between PHY layer and Logical PHY or upper layer will be at least $42 * R$ signals. For example, if Phy_Bitrate = 6.4Gbps and LogPhy_Clk = 1GHz, then $R = 8$.

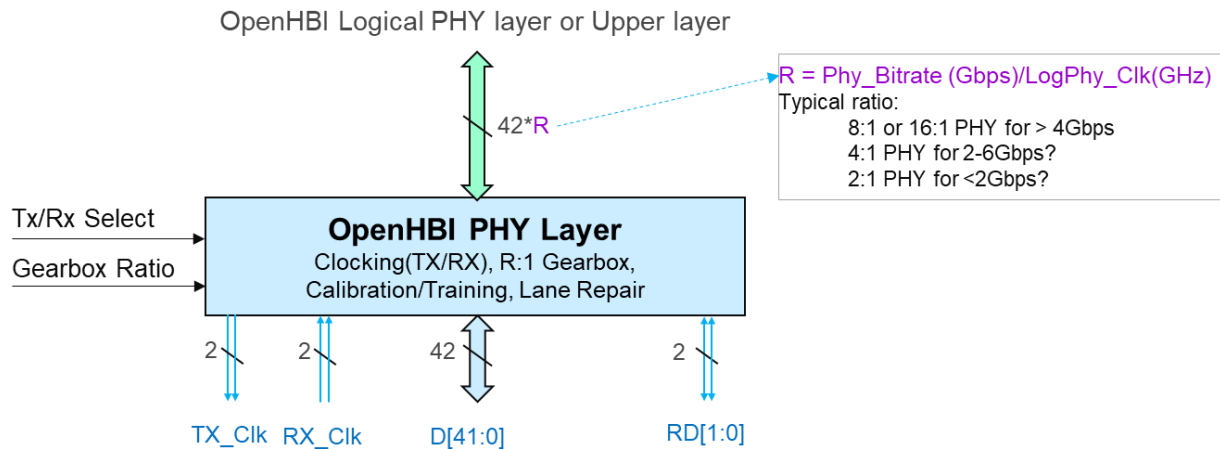


Figure 6-6: OpenHBI PHY layer functions

6.3.1 Data I/O and Clock/Strobe

The OpenHBI PHY layer generates TX_Clk (output) and uses RX_Clk (input) to sample the input D[41:0] and RD[1:0].

The TX_Clk shall be in phase-aligned relationship with DWORD D[41:0] and RD[1:0].

The RX_Clk shall be in phase-aligned relationship with DWORD D[41:0] and RD[1:0].

OpenHBI V1.0 defines a minimum of 200Mbps and a maximum of 8Gbps (unterminated) data rate running in DDR mode. The OpenHBI DWORD can operate at any data rate in between 200Mbps and 8Gbps. The TX_Clk / RX_Clk frequency have a range of minimum 100MHz to a maximum of 4GHz correspondingly.

6.3.2 Speed/Width Conversion

The PHY layer shall support 2:1, 4:1, 8:1, and 16:1 conversion ratios, programmable at initialization time. Therefore, on the Logical PHY or upper layer side, the resulting data bits width shall be 84, 168, 336, and 672 bits, respectively.

Table 6-2 illustrates a few example PHY Bitrates, Logical PHY or upper layer clock frequency and the resulting Gearbox ratio (R) and the pin count between PHY and the Logical PHY / upper layer.

Table 6-2: Example Gearbox Ratio and # of signals between PHY and Logical PHY / upper layer.

Gearbox Ratio (R)	OpenHBI PHY Bitrate [Gbps]	Logical PHY or upper layer clock	# of Data signals / DW	# of signals to Logical PHY or upper layer (= 42*R)
2	2.0	1GHz	42	84
4	4.0	1GHz	42	168
8	6.4	800MHz	42	336
16	8.0	500MHz	42	672

Implementation Note: Anchor or Chiplet on the link can employ different Gearbox ratios. The latency due to Gearbox function is (typically) one Logical PHY layer clock period, independent of the Gearbox ratio employed.

6.3.3 Serialization Order

The OpenHBI PHY shall serialize data in contiguous groups starting at the **LSB** (least significant bit). Table 6-3 below shows an example serialization order of 8:1 gearbox ratio with all 42 data signals D[41:0] are transporting data. The LSB of each data beat shall be transmitted on OpenHBI D[0].

Table 6-3: Example Serialization Order for 8:1 gearbox ratio

Serialized data beat	Logical PHY layer bits transmitted
1 (first)	Bit [41:0]
2	Bit [83:42]
3	Bit [125:84]
4	Bit [167:126]
5	Bit [209:168]
6	Bit [251:210]
7	Bit [293:252]
8 (last)	Bit [335:294]

6.3.4 Calibration and Training

Typically, the IO calibration and Clock-to-Data training and deskewing are performed in the PHY layer. Refer to Section 10.5 for more details.

6.3.5 Lane Repair

OpenHBI supports lane repair feature to enhance yield and improve manufacturability. It supports remapping of one broken lane connection per double byte (20 lanes). Each double byte within a DWORD can be repaired independently.

The redundant signal RD0 is used for remapping OpenHBI D[20:6] and [D4:0].

The redundant signal RD1 is used for remapping OpenHBI D[41:37] and D[35:21].

Note that, the D[5] and D[36] are treated differently than other D[x] signals and that they are not repairable by the lane repair scheme. Please refer to Section 8.3 and Section 10.3.2.10 for more details.

6.3.6 PHY Operation

Once enabled and trained, the OpenHBI PHY layer shall provide continuous data flow per direction (TX_Clk will toggle continuously).

A Link layer may provide optional flow control and valid data / idle transfer indication as needed, but the PHY layer always sends/receives a word (up to 42 data bits) every clock cycle.

6.4 OpenHBI Scalability

6.4.1 DWORD Array and C2C Instance

OpenHBI defines DWORD as basic building block of the C2C interface. Each OpenHBI DWORD can transfer up to 336 Gbps (i.e., 42 x 8Gbps) of raw bandwidth. The OpenHBI C2C interface bandwidth can be scaled up by using multiple DWORD units in both TX and RX directions.

To facilitate interoperability, routability and configuration management, the specification defines the concept of an OpenHBI C2C “Instance”, as defined in Table 6-4 below.

Table 6-4: Definition of Full, Half, Quarter Instances and Anchor / Chiplet Beachfront Rules

Instance	DWORD Array	# of DW	Anchor BF rules	Chiplet BF rules
“Full”	8DW-high x 4DW-deep	32	Allowed	Allowed
“Half” (Case 1)	8DW-high x 2DW-deep	16	Allowed but not recommended	Allowed
“Half” (Case 2)	4DW-high x 4DW-deep	16	Disallowed	Allowed but not recommended
“Quarter” Instance	4DW-high x 2DW-deep	8	Disallowed	Allowed

Interoperability Rules:

- 8DW-High Anchor shall be interoperable with Chiplets of either 8DW- or 4DW-high
- 4DW-deep Anchor shall be interoperable with Chiplets of either 4DW- or 2DW-deep

An Anchor should support one or multiple “Full” instances of 8H x 4D (i.e., 8DW-high x 4DW-deep) array of configurable, bidirectional DWORDs to maximize interoperability.

Table 6-5: Interoperability Matrix based on the interoperability rules and guidelines

Chiplet		8-High		4-High	
Anchor		4DW-deep	2DW-deep	4DW-deep	2DW-deep
8-DW High	4DW-deep	Yes	Yes	Yes	Yes
	2DW-deep	Yes*	Yes	Yes*	Yes

* A Chiplet with 4DW-deep instance can optionally support 2DW-deep mode to interoperate with 2DW-deep Anchor.

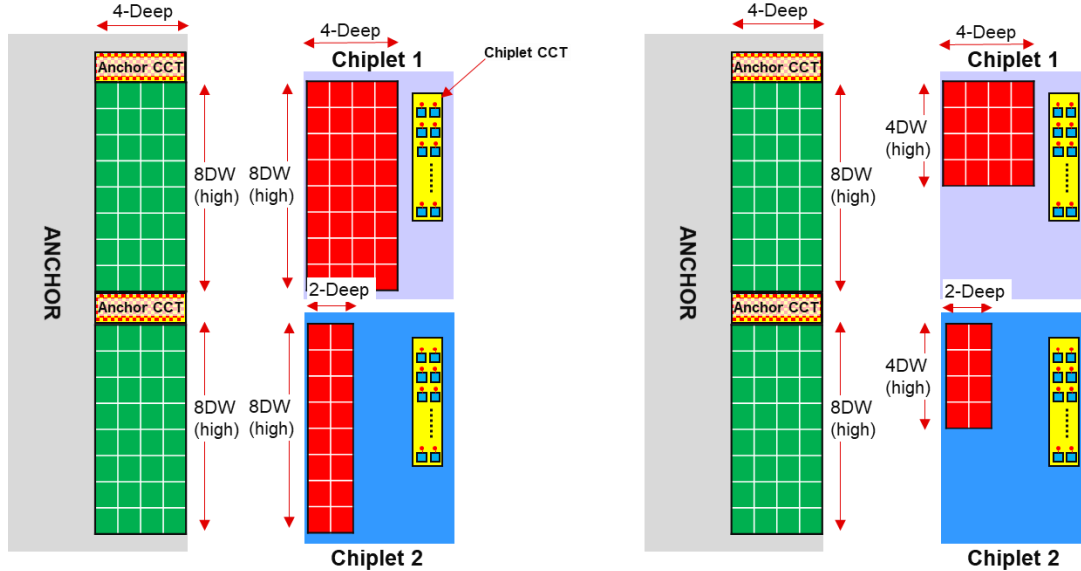


Figure 6-7: Anchor Full instance and Chiplet Full/Half/Quarter Instance interop examples

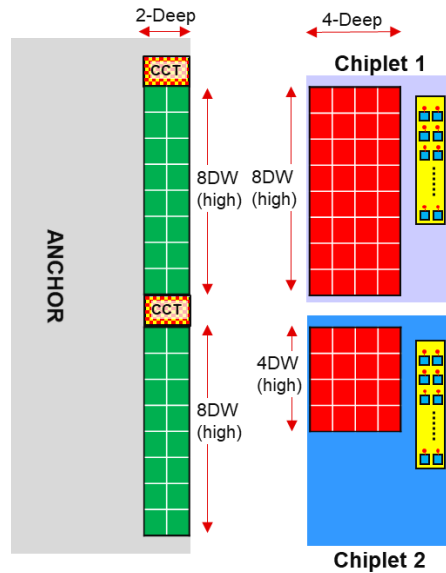


Figure 6-8: Anchor Half instance and Chiplet Full/Half Instance interop examples

6.4.2 Standard Matrix and Compact Matrix

Section 6.1 introduced the “virtual” bump map concept. It is referred to as “Standard Matrix” on the left of Figure 6-9 below.

OpenHBI C2C interface may also support an alternate “Compact Matrix” as shown on the right of Figure 6-9, which conceptually convert each row of 6 signals of “Standard Matrix” into two rows of 3 signals in a

checkerboard pattern of the same height. Effectively converting a DWORD with 14 rows x 12 columns (Standard Matrix) into 28 rows x 6 columns (Compact Matrix) of exact same height.

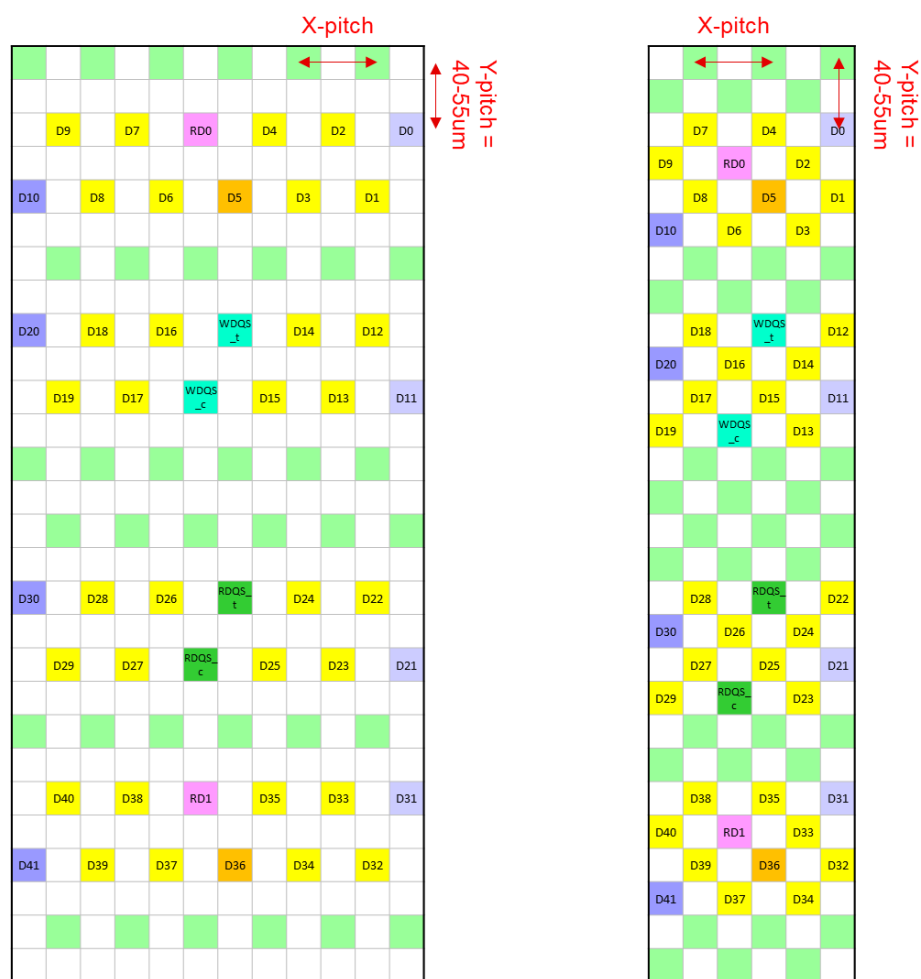


Figure 6-9: **Standard Matrix** (Left)

and

Compact Matrix (Right)

6.4.3 Y-pitch Range and DWORD Height Range

To avoid confusion, the Y-pitch definitions are different between the two matrix options:

- Standard Matrix: Y-pitch is ubump pitch, in the beachfront height direction, between the immediate next checkerboard row, as shown in the left of Figure 6-9.
- Compact Matrix: Y-pitch is ubump pitch, in the beachfront height direction, between every other checkerboard row, as shown in the right of Figure 6-9.

OpenHBI specifies the Y-pitch range from 40um to 55um.

Correspondingly, the DWORD height (= Y-pitch x 14) has a range of 560um to 770um.

X-pitch definition is the same for both matrix options, which is the ubump pitch, in the beachfront depth direction, between every other checkerboard column, as shown in Figure 6-9. X-pitch and DWORD depth are implementation-specific and do not affect interoperability and routability.

6.5 Chiplet Configuration and Test (CCT) interface

Anchor CCT (Chiplet Configuration and Test) interface provides configuration and test resource for chiplets on a quantized basis.

The Anchor shall provide one CCT every 8DW-high instance (i.e., either 8H x 4D Full instance or 8H x 2D Half instance). The Anchor CCT can perform up to 3 functions and Chiplet CCT can perform up to 2 functions summarized below.

Table 6-6: Summary of Anchor and Chiplet CCT functions

	Anchor CCT Functions	Chiplet CCT Functions
1	Chiplet Config (I3C Controller)	Chiplet Config (I3C Target)
2	Post-packaged tests	Post-packaged tests Wafer-level test probes
3	Approximate the Beachfront-to-Beachfront gap between two adjacent chiplets on same Anchor die edge to alleviate C2C beachfront misalignment	n/a

6.5.1 Anchor and Chiplet CCT

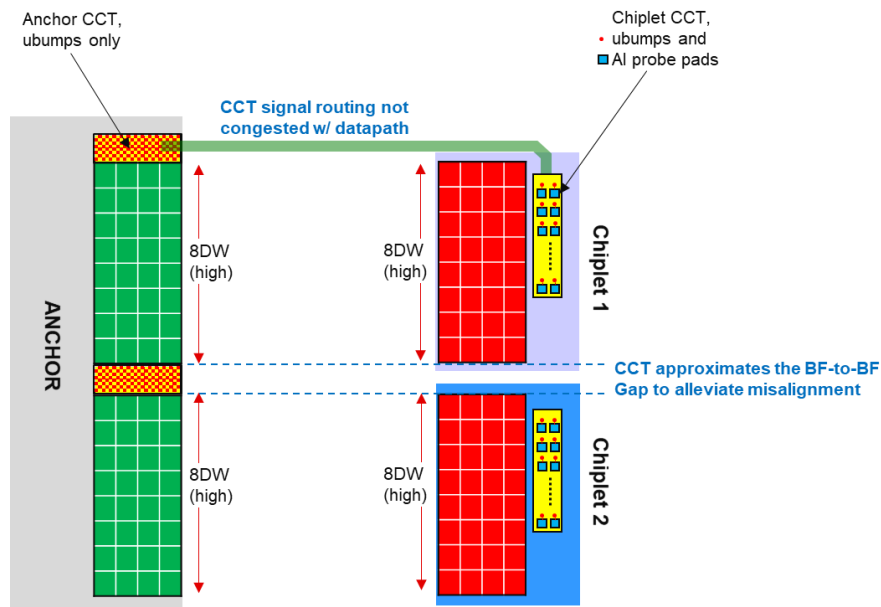


Figure 6-10: Anchor and Chiplet CCT concept

Figure 6-10 illustrates the Anchor and Chiplet CCT concept.

Anchor CCT is placed next to the Anchor C2C instance and contains ubumps only.

Chiplet CCT is **not** placed next to the Chiplet C2C instance. Its placement is chiplet-specific. It can be placed somewhere convenient for wafer probe (aka wafer sort), for example, behind the Chiplet C2C instance as shown in Figure 6-10.

Each Chiplet CCT signal ubump is connected to a co-located probe pad. For critical signal that support redundancy, two signal ubumps will share a probe pad as illustrated in Figure 6-11 below to minimize the loading.

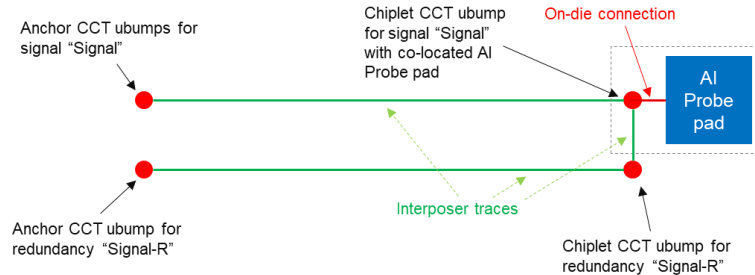


Figure 6-11: Signal with redundancy sharing a probe pad

6.5.2 MIPI I3C as Configuration Interface

OpenHBI specification adopts MIPI I3C v1.1.1 (or future backward-compatible version) as the default configuration interface and shall support HDR-TSP mode up to 33Mbps (effective).

The I3C interface for OpenHBI CCT shall support 1.2V operating voltage. It shall support I3C IBI (In-Band Interrupt) with a Mandatory Data Byte (MDB).

Implementation Note: Anchor or Chiplet is allowed to support I3C Basic, which is interoperable with I3C but at a lower data rate. If MIPI I3C Basic is used, it shall support SDR mode up to 12.5Mbps.

During initialization, the configuration firmware/software shall switch I2C FM+ mode (1Mbit/s) to the highest data rate mode commonly supported between the Anchor and Chiplet as soon as possible before performing the chiplet configuration.

OpenHBI V1.0 specification does not define a common I3C configuration command set. The Anchor and Chiplet I3C controllers shall support Direct Read/Write, Direct Write and Direct Read CCC.

6.5.2.1 I3C Vendor Extension Direct Common Command Codes (CCCs)

To enhance interoperability between vendors, OpenHBI devices shall support the following I3C Vendor Extension CCCs.

Vendor extension codes 0xE0 to 0xEF will be reserved for OpenHBI CCCs.

Vendor extension codes 0xF0 to 0xFE will be reserved for vendor-specific CCCs.

Table 6-7: I3C Vendor Extension Common Command Codes (CCCs)

Command Code	Command Name	Command Description
E0	SETLRR	Transfers the lane repair information from the anchor RX DWORDs to the chiplet TX DWORDs. The number of bytes transferred is equal to 2 times the number of anchor RX DWORDs.
E1	GETLRR	Transfers the lane repair information from the chiplet RX DWORDs to the anchor TX DWORDs. The number of bytes transferred is equal to 2 times the number of chiplet RX DWORDs.
E2 – EF	RFU	Reserved for OpenHBI CCCs.
F0 - FE	VSCCC	Reserved for vendor specific CCCs.

6.5.2.2 I3C In-Band Interrupts (IBIs)

To enhance interoperability between vendors, OpenHBI devices shall support I3C IBI with Mandatory Data Byte (MDB).

OpenHBI IBIs use the interrupt group identifier for “Non MIPI Reserved” (MDB[7:5] = 3'b011).

Vendor-specific IBIs use interrupt group identifier for “User Defined” (MDB[7:5] = 3'b000).

Table 6-8: I3C In-Band Interrupts (IBIs)

Interrupt Group Identifier (MDB[7:5])	Specific Interrupt ID (MDB[4:0])	Interrupt Name	Interrupt Description
3'b000	5'h00 – 5'h1F	RSVDV	Reserved for vendor specific.
3b001 – 3'b010	5'h00 – 5'h1F	MIPI	Reserved for MIPI.
3'b011	5'h00	INITDONE	Indicates that PHY initialization completed without errors.
	5'h01	INITDONERR	Indicates that PHY initialization completed with errors.
	5'h02	TRNDONE	Indicates that training completed without errors.
	5'h03	TRNDONERR	Indicates that training completed with errors.
	5'h04	TXRQST	Request TX to transmit data. The length of the transmitted data is controlled by burst length register (BLR).
	5'h05	RXDERR	Indicates there was a parity error on the received data.
	5'h06 – 5'h1F	RSVDO	Reserved for OpenHBI.
3'b100	5'h00 – 5'h1F	RSVDV	Reserved for vendor-specific timing information.
3'b101 – 3'b111	5'h00 – 5'h1F	MIPI	Reserved for MIPI.

6.5.3 CCT and Vendor-Specific Signals

Table 6-9 shows the OpenHBI-defined (normative) CCT signals. All CCT signals shall be nominal 1.2V IO. The signals Mode, PwrGood, Reset, RefClk, I3C_CLK, I3C_DATA, JTAG_TCK/TMS/TDI/TDO/TRST are critical signals and support redundant ubumps and interposer connections to ensure proper operation and for yield enhancement.

“-R” denotes the redundant signal of the named signal. E.g., Mode-R is the redundant signal of Mode.

Anchor can use “PwrGood = high” to indicate to the Chiplet that power is good and can also use it as power-on / cold reset signal. Both PwrGood and Reset can functionally reset the OpenHBI interface and the Chiplet-specific functions. One primary difference between the two signals is that certain sticky registers/bits or certain status (e.g., security or persistent memory status) etc., may persist through assertion of “Reset” but cannot persist when PwrGood is de-asserted.

Mode=0 (default) configures the chiplet into Operating mode. Mode=1 configures the chiplet into Vendor-Specific Test mode which enable the Anchor to exercise post-packaged testing of the Chiplet that is connected to that specific CCT.

OpenHBI adopts I3C for configuration and management as described in Section 6.5.2. JTAG and I3C interfaces can be used during wafer testing of chiplets and post-packaged boundary scan and additional vendor-specific test features and should include VS[15:0] and RSVD[0] broken lane detection to facilitate lane repair using LR[0], if needed. JTAG_TRST and JTAG_TRST-R are optional normative signals.

CCT defines 16 vendor-specific signals VS[15:0]. Note that, the VS[15:0] in Mode=0 and Mode=1 may perform same or completely different functions in Operating and Vendor-Specific Test modes. All other CCT signals have the same functions between Mode = 0 and Mode =1.

RSVD[0] is reserved for future OpenHBI version. LR[0] is the Lane Repair signal. It covers VS[15:0] + RSVD[0] in both Mode=0 and Mode=1.

Table 6-9: OpenHBI-defined CCT signals [Normative]

Mode = 0	Mode = 1	Direction		Probe Pad	Notes
Operating Mode (default)	Vendor-Specific Test Mode	Anchor	Chiplet	Chiplet CCT Probe Pad	
PwrGood	PwrGood	Out	In	1	High = PwrGood
PwrGood-R	PwrGood-R	Out	In		Redundancy for PwrGood
Reset	Reset	Out	In	1	High = Reset
Reset-R	Reset-R	Out	In		Redundancy for Reset
Mode	Mode	Out	In	1	
Mode-R	Mode-R	Out	In		
RefClk (default)	RefClk (default)	Out	In	1	
RefClk2	RefClk2	Out	In		Redundancy for RefClk RefClk & RefClk2 ubump/traces are not connected to reduce loading
I3C_CLK	I3C_CLK	Out	In	1	
I3C_CLK-R	I3C_CLK-R	Out	In		
I3C_DATA	I3C_DATA	I/O	I/O	1	
I3C_DATA-R	I3C_DATA-R	I/O	I/O		
JTAG_TCK	JTAG_TCK	Out	In	1	
JTAG_TCK-R	JTAG_TCK-R	Out	In		
JTAG_TMS	JTAG_TMS	Out	In	1	
JTAG_TMS-R	JTAG_TMS-R	Out	In		
JTAG_TDI	JTAG_TDI	Out	In	1	
JTAG_TDI-R	JTAG_TDI-R	Out	In		
JTAG_TDO	JTAG_TDO	In	Out	1	
JTAG_TDO-R	JTAG_TDO-R	In	Out		
JTAG_TRST	JTAG_TRST	Out	In	1	Optional signal
JTAG_TRST-R	JTAG_TRST-R	Out	In		Optional signal
VS[15:0]	VS[15:0]	I/O	I/O	16	Vendor-Specific signals
RSVD[0]	RSVD[0]	I/O	I/O	1	Reserved for future version
LR[0]	LR[0]	I/O	I/O		Lane Repair[0]
Total = 40	Total = 40			Total = 28	

The Anchor and Chiplet CCT block should provide an 8-bit register (informative) to provide the lane repair signal remapping. The lower 4 bits controls VS[7:0] remapping and the upper 4 bits controls VS[15:8] and RSVD[0] remapping. The lower-nibble and upper-nibble cannot be set to Non-1111b value at the same time since LR[0] can only repair one signal out of VS[15:0]+RSVD[0].

The 4-bit nibbles encoding and the lane repair shift chain in Table 6-10 and Table 6-11 are normative.

Table 6-10: 4-bit field encoding for VS[7:0] lane repair

Description	Encoding (Lower 4 bits)	VS0 (Lane0)	VS1 (Lane1)	VS2 (Lane2)	VS3 (Lane3)	VS4 (Lane4)	VS5 (Lane5)	VS6 (Lane6)	VS7 (Lane7)	LR[0]
Repair Lane 0	0000	XX	VS0	VS1	VS2	VS3	VS4	VS5	VS6	VS7
Repair Lane 1	0001	VS0	XX	VS1	VS2	VS3	VS4	VS5	VS6	VS7
Repair Lane 2	0010	VS0	VS1	XX	VS2	VS3	VS4	VS5	VS6	VS7
Repair Lane 3	0011	VS0	VS1	VS2	XX	VS3	VS4	VS5	VS6	VS7
Repair Lane 4	0100	VS0	VS1	VS2	VS3	XX	VS4	VS5	VS6	VS7
Repair Lane 5	0101	VS0	VS1	VS2	VS3	VS4	XX	VS5	VS6	VS7
Repair Lane 6	0110	VS0	VS1	VS2	VS3	VS4	VS5	XX	VS6	VS7
Repair Lane 7	0111	VS0	VS1	VS2	VS3	VS4	VS5	VS6	XX	VS7
Reserved	1000-1110	VS0	VS1	VS2	VS3	VS4	VS5	VS6	VS7	LR0
No Repair(default)	1111	VS0	VS1	VS2	VS3	VS4	VS5	VS6	VS7	LR0

Table 6-11: 4-bit field encoding for VS[15:8] and RSVD[0] lane repair

Description	Encoding (Upper 4 bits)	VS8 (Lane0)	VS9 (Lane1)	VS10 (Lane2)	VS11 (Lane3)	VS12 (Lane4)	VS13 (Lane5)	VS14 (Lane6)	VS15 (Lane7)	RSVD0 (Lane8)	LR[0]
Repair Lane 0	0000	XX	VS8	VS9	VS10	VS11	VS12	VS13	VS14	VS15	RSVD0
Repair Lane 1	0001	VS8	XX	VS9	VS10	VS11	VS12	VS13	VS14	VS15	RSVD0
Repair Lane 2	0010	VS8	VS9	XX	VS10	VS11	VS12	VS13	VS14	VS15	RSVD0
Repair Lane 3	0011	VS8	VS9	VS10	XX	VS11	VS12	VS13	VS14	VS15	RSVD0
Repair Lane 4	0100	VS8	VS9	VS10	VS11	XX	VS12	VS13	VS14	VS15	RSVD0
Repair Lane 5	0101	VS8	VS9	VS10	VS11	VS12	XX	VS13	VS14	VS15	RSVD0
Repair Lane 6	0110	VS8	VS9	VS10	VS11	VS12	VS13	XX	VS14	VS15	RSVD0
Repair Lane 7	0111	VS8	VS9	VS10	VS11	VS12	VS13	VS14	XX	VS15	RSVD0
Repair Lane 8	1000	VS8	VS9	VS10	VS11	VS12	VS13	VS14	VS15	XX	RSVD0
Reserved	1001-1110	VS8	VS9	VS10	VS11	VS12	VS13	VS14	VS15	RSVD0	LR0
No Repair(default)	1111	VS8	VS9	VS10	VS11	VS12	VS13	VS14	VS15	RSVD0	LR0

Implementation Note: In Anchor-to-Anchor use case, one of the Anchor dice will act as configuration manager and the other Anchor will act as configuration subordinate. Similarly, in Chiplet-to-Chiplet use case, one of the Chiplet dice will act as the manager and the other will act as the subordinate. The mechanism to configure the Manager and Subordinate roles is out of scope of the OpenHBI specification.

6.5.4 CCT Association Rules

A CCT shall associate with an 8DW-high (Full 8Hx4D or Half 8Hx2D) Instance as defined in the Table 6-12 below.

Table 6-12: CCT Association Rules (Normative)

Anchor Die Edge	CCT Association
Right	Each CCT associates with C2C Instance “below” it
Bottom	Each CCT associates with C2C Instance on its “left”
Left	Each CCT associates with C2C Instance “above” it
Top	Each CCT associates with C2C Instance on its “right”

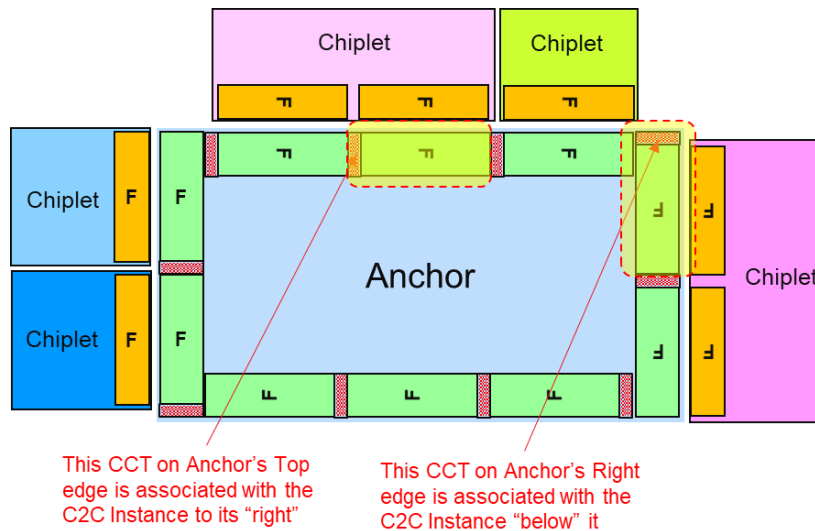


Figure 6-12: Example CCT association with C2C Instances on different Anchor die edges

6.5.5 CCT size guidelines (Informative)

One of the CCT functions is to approximate the Beachfront-to-Beachfront gap between two adjacent chiplets on same Anchor die edge to alleviate C2C beachfront misalignment.

However, due to different advanced packaging technologies have:

- Different Anchor-to-Chiplet and Chiplet-to-Chiplet spacing
- Different CSR (corner stress relief) and AM (alignment mark) placement and size requirements
- Different minimum ubump pitch

Together with a range of Y-pitch, OpenHBI defines Anchor CCT Height as informative guideline as below:

$$\text{CCT Height} \sim \text{Chiplet Spacing} + 2 \times (\text{S} + \text{DS} + \text{Offset})$$

where Chiplet Spacing: Packaging technology-specific chiplet-to-chiplet spacing requirements

S = Scribe of Chiplet die

DS = Die Seal of Chiplet die

Offset = additional offset by CSR/AM (design-specific and packaging technology-dependent)

Implementation Note: The CCT ubump pitch does not necessarily same as OpenHBI C2C beachfront. Anchor and Chiplet designers need to ensure the CCT has enough ubumps.

6.5.6 Optional Anchor “Dummy” CCT

Refer to Section 6.5.1, with the CCT placed next to the C2C beachfront, one would notice that the combined CCT and C2C beachfront are not rotation-symmetric mechanically.

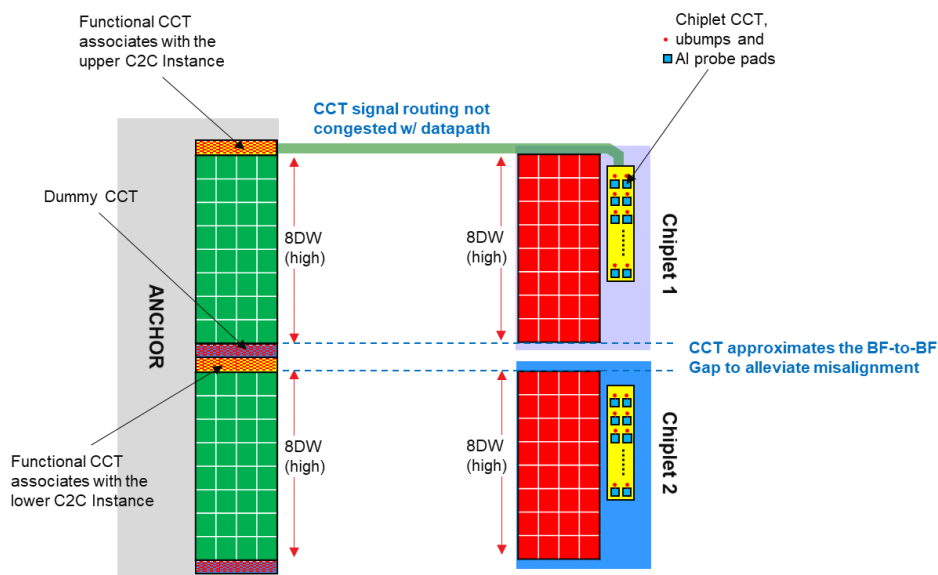


Figure 6-13: Optional Dummy CCT concept

Some interposer packaging technologies have very stringent mechanical spacing rules in order to avoid unnecessary dummy dice, especially between opposite edges of the Anchor die.

One solution is to introduce the optional “Dummy CCT” that is same size as the Functional CCT. Refer to Figure 6-13, the Functional and Dummy CCT will sandwich the C2C beachfront and will provide mechanical rotation symmetry.

Notice that, in the example, the new combined Functional CCT + Dummy CCT will approximate beachfront-to-beachfront gap so adding the optional Dummy CCT will reduce the Functional CCT approximately by half. Care must be exercised to ensure Dummy CCT do not create incorrectly sized Anchor CCT and introduce misalignment between Anchor and Chiplet beachfront.

6.5.7 CCT Virtual Bump Map (CCT VBM)

Similar to DWORD Virtual Bump Map target concept described in Section 6.1, OpenHBI specification defines a (normative) reference CCT “virtual” Bump Map, or CCT VBM, to ease Anchor and Chiplets CCT routability even if they have different dimensions, and rows / columns arrangement.

The CCT “virtual” bump map has 2 rows by 20 columns of signal ubumps.

Figure 6-14 defines Anchor “True” orientation CCT VBM facing left edge.

Figure 6-14 also applies to Chiplet “True” orientation CCT VBM facing its right edge. Note that, Chiplet-side CCT (with probe pads) is not placed inline with the DWORD array and is chiplet-specific but its physical bump map should be designed to route seamlessly to the CCT virtual bump map.

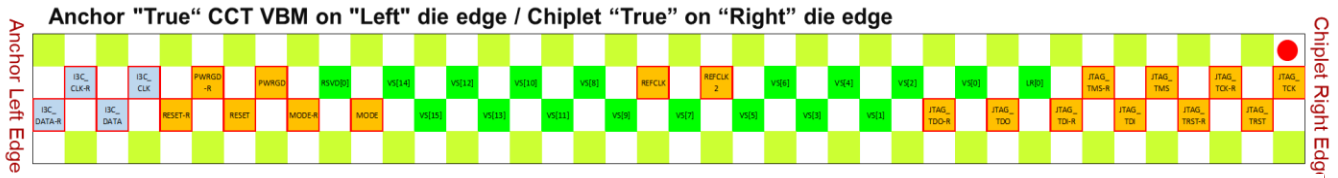


Figure 6-14: CCT "True" Virtual Bump Map concept

Figure 6-15 defines Anchor "Rotated" orientation CCT VBM facing right edge. Figure 6-15 also applies to Chiplet "Rotated" orientation CCT VBM facing its left edge.

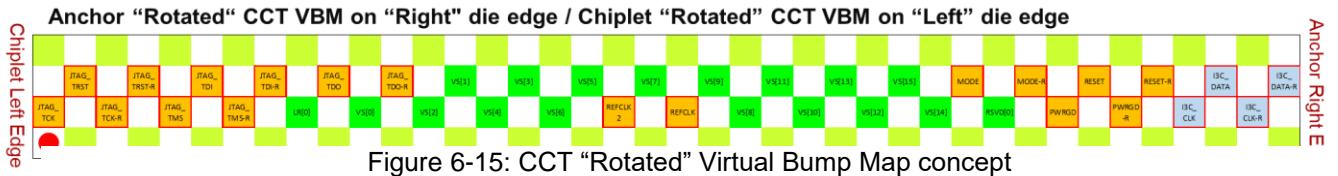


Figure 6-15: CCT "Rotated" Virtual Bump Map concept

Each cell shows the CCT signal as defined in Section 6.5.3.

- Signals in "orange" cells with "red border" are those signals with redundant counterpart.
- I3C signals in "light blue" cells with "red border" indicates these are I3C-compliant IOs with redundant counterpart.
- Signals in "green" cells are VS[15:0], RSVD[0] and the LR[0] signals grouped around RefClk and RefClk2 and is close to rotation-symmetric for optional bit-reordering support.

The unlabeled "light green" cells are typically used for power delivery network (PDN) including GND and one or more power rails for CCT IO. Since this is a virtual bump map, designs may use implementation-specific PDN as long as it meets the SI and reliability of the JTAG, I3C and the vendor specific VS[15:0] signal group.

For normal Anchor-Chiplet interconnection like the example in Figure 6-10, if both Anchor and Chiplet CCT and its orientation are compliant to the CCT VBM, then the CCT signal traces should route seamlessly on any of the Anchor die edges.

For homogenous peer-to-peer use case involving an Anchor 1 connecting to an identical Anchor 2 configured as a configuration subordinate as shown in Figure 6-16, the two CCTs will not be facing each other and will be in R180 orientation. Therefore, the CCT connection will have longer traces (typically in the order of 5-10mm) which is fine with JTAG typically running at low-100MHz data rate and I3C is design for long reach and multi-drop. Care should be taken, and additional SI analysis may be needed if the VS[15:0] group is running at higher than a low-100Mbps data rate.

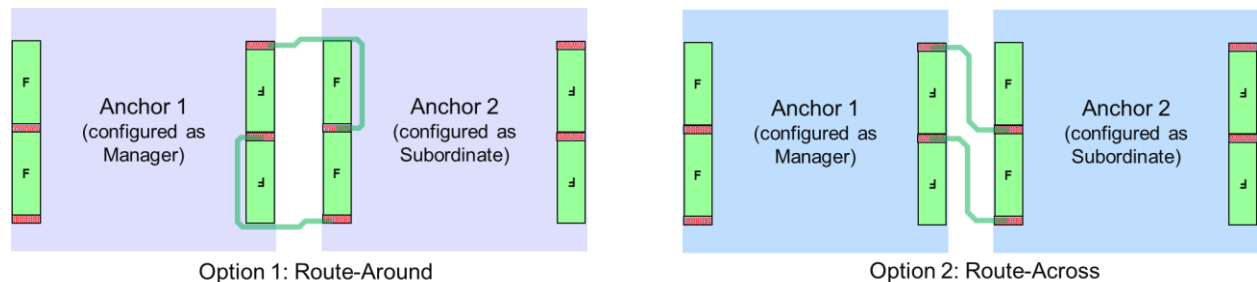


Figure 6-16: Anchor-to-Anchor CCT connection options

Typically, there are two routing options as shown:

- Option 1: Route-around the C2C beachfront
- Option 2: Route-across the C2C beachfront

Option 1 is recommended as it doesn't require additional routing layer(s). In addition, "Route-around" effectively "un-rotate" the connections between the two CCTs so they are effectively in the SAME orientation and easier to route.

Option 2 requires additional routing layer(s). The signals with redundant counterpart will have crisscrosses that need to be resolved by interposer routing. If VS[15:0] signals and RSVD[0] are used, it may support CCT bit-ordering (Optional Normative) to reorder the connection map below in Table 6-13, if enabled:

Table 6-13: Anchor-to-Anchor CCT VS[15:0] and RSVD[0] connection map

Anchor 1 (configured as Manager)	Anchor 2 (configured as Subordinate)
VS[0]	RSVD[0]
VS[1]	VS[15]
VS[2]	VS[14]
VS[3]	VS[13]
VS[4]	VS[12]
VS[5]	VS[11]
VS[6]	VS[10]
VS[7]	VS[9]
VS[8]	VS[8]
VS[9]	VS[7]
VS[10]	VS[6]
VS[11]	VS[5]
VS[12]	VS[4]
VS[13]	VS[3]
VS[14]	VS[2]
VS[15]	VS[1]
RSVD[0]	VS[0]

6.5.8 Compatibility with Wafer Probe environments for chiplet test

The CCT is primarily used to communicate between Anchor and Chiplet within the final multi-dice package. The Chiplet-side CCT also provides a test control interface for wafer probe to sort for the "Known Good Dice" or KGD before packaging.

The minimal set for control would be JTAG and I3C. The electrical specifications should be compatible with wafer probe environments (e.g. ESD, drive strength and probe interface). If VS[15:0], RSVD[0] and LR[0] signal group is used and is required at wafer probe, Chiplet designer or IP provider should ensure it is compatible with the wafer probe environment as well.

7. OpenHBI Logical PHY Layer

This section describes OpenHBI Logical PHY layer and related services to work with PHY layer to form an OpenHBI transport for chiplet applications.

OpenHBI Logical PHY layer provides optional-normative services. It involves TX and RX functionalities. A typical bidirectional Logical PHY layer function is programmable at initialization time to operate in either TX or RX mode. Fixed function chiplets or designs with known configurations may implement fixed direction Logical PHY interface units.

Refer to Figure 7-1, this layer defines functions (services) which include:

- Bit reordering for rotated die
- DBI (data bus inversion)
- Framing and alignment
- Parity generation/check

This Bit-reordering function enables straight line routing between two OpenHBI chiplets in 180° rotated (aka R180) orientation. Refer to Section 8 for details.

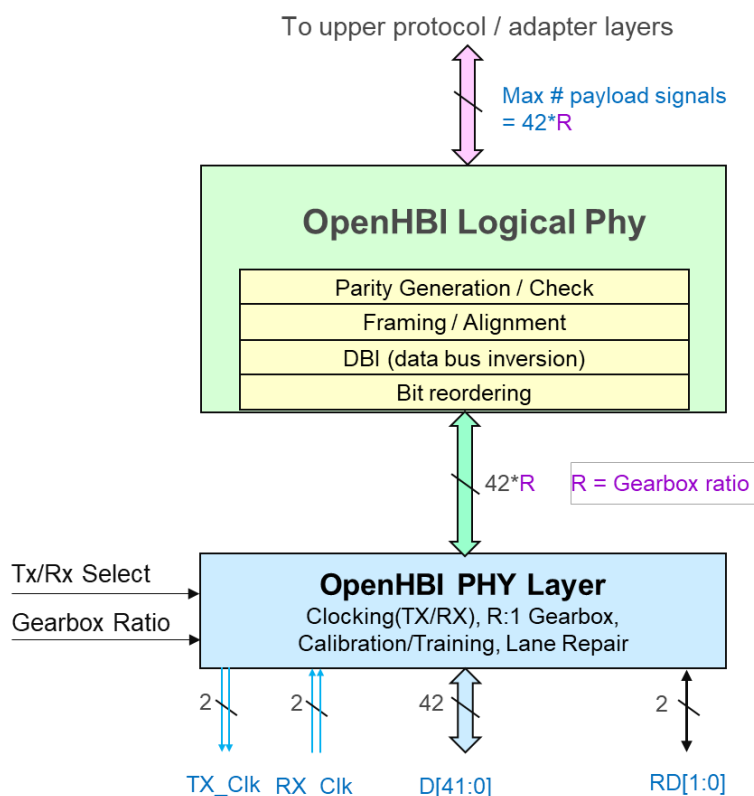


Figure 7-1: OpenHBI Logical PHY layer functions

The interface between PHY layer and Logical PHY layer consists of $42 \times R$ signals (R =gearbox ratio) plus local clock and optional implementation-specific signals.

Each Logical PHY layer service is optional-normative and can be disabled. When all features are enabled, 6 of the 42 data pins of a DWORD are reserved for Logical PHY layer functions, as follows:

- 1 pin for Framing/Alignment (D[41])
- 1 pin for Parity (D[40])
- 4 pins for DBI (D[39:36])
- 36 pins for data payload (D[35:0])

Disabling a service frees up the associated pin(s) and increases the number of signals available to the upper layer. For example, disabling DBI feature will provide $4 \times R$ more signals to upper layer, where R = Gearbox ratio. The maximum number of data payload signals between Logical PHY layer and upper layer is $42 \times R$.

In use case where Logical PHY layer is not used, but the chiplet interface are in 180° rotated die orientation, the upper layer that connect to the PHY layer directly shall include the equivalent Bit-reordering function as defined in this Section 8.

7.1 Bit Reordering for Rotated Die

When 2 chiplets connect with same-orientation, interposer routing is mostly straight-lines and shortest paths. When one die is rotated 180 degrees (in R180 orientation), it is not feasible to cross all the DWORD wires on the interposer.

The Bit Reordering feature enables the interposer wires remain mostly straight-lines and shortest paths, and the resulting out-of-order bits get reordered in the Logical PHY layer, on the RX side only. There are two Bit-reordering modes and sub-modes depending on whether lane repair is used. For more details see Section 8.

7.2 Data Bus Inversion – Enhanced 9-bit DBI mode

OpenHBI supports enhanced 9-bit DBI function. DBI is an optional normative service that ensures no more than 50% of the 9 data signals + DBI signal (total 10 signals per DBI signal) will toggle on the OpenHBI physical wires, which can reduce the energy/bit with the added benefits to reduce the SSO (Simultaneous Switching Output) noise and improve the signal integrity of the C2C interconnect.

If DBI is enabled, 4 pins are dedicated to DBI, each pin is associated with 9 data pins.

- D[36] = DBI0 covers D[8:0]
- D[37] = DBI1 covers D[17:9]
- D[38] = DBI2 covers D[26:18]
- D[39] = DBI3 covers D[35:27]

If DBI is not enabled, the DBI signals can be used as data-carrying signals D[39:36] (Logical PHY Mode 2, 3 or 4) or those signals can be disabled (or do not toggle) to save power.

Framing/D[41] and Parity/D[40] bits are not covered by DBI functions.

On the TX side, the DBI function counts the number of data transitions within the associated 9-bit word. This involves comparing current bit value to previous bit value. If 5 or more bits change value, then the DBI pin will be asserted (=1) and all 9 data bits will be inverted. Otherwise, DBI is not asserted (=0) and the data bits are unchanged. This ensure always ≤ 4 transitions (out of 9-bit word) in any transfer.

On the RX side, the incoming DBI bit, when asserted, is used to invert the 9 incoming data bits.

When DBI is enabled, the maximum number of transitions per DWORD is $4 \times (4 \text{ (data)} + 1 \text{ (dbi)}) + 2 \text{ (d41, d40)} = 22$, when compared with 42 transitions without DBI.

Note that on the TX side, DBI state shall be reset at initialization time. The first transfer may contain more than 4 transitions, but the subsequent signal toggling counts will generate the correct DBI state accordingly.

7.3 Framing and Alignment

Logical PHY layer may support optional framing and alignment services for Intra-DW Framing (within a DWORD) and Inter-DW Alignment (with optional alignment service between multiple DWORDs by the upper layer above the Logical PHY).

When configured, D[41] becomes a dedicated “Framing” signal sent along with the data D[40:0]. The Framing signal shall be active High, 1UI wide pulse concurrent with the data phase of D[40:0] that corresponds to the first beat (least significant word) in a serialized sequence generated by TX gearbox.

The RX Logical PHY is responsible to detect, achieve and maintain proper Framing boundary. Goal is to ensure first “beat” in a burst sequence carries the “Framing bit = 1” signaling.

Alignment Error can be reported via (informative) CCS Status registers. Reporting and handling of Framing errors is out of scope of OpenHBI specification. Refer to Section 10.3 CCS Registers.

7.3.1 Intra-DW Framing

On TX side, the TX PHY layer gearbox serializes the wide data word into R data beats (e.g., $R=4:1$) carried by the TX DWORD data signals.

On RX side, the sampled “Framing” signal is expected to be 0b000....001, where Framing bit is the LSB (least significant bit), and it shall have (R-1) “zeros” trailing the “Framing bit = 1”.

Framing Error should be reported if:

1. If sampled Framing bit value is not as expected, i.e., $\text{LSB} = 0 \rightarrow \text{Framing Error}$
2. If sampled Framing bit has 1 in LSB (least significant bit) but the # of trailing zeros is different from (R-1) $\rightarrow \text{Framing Error}$

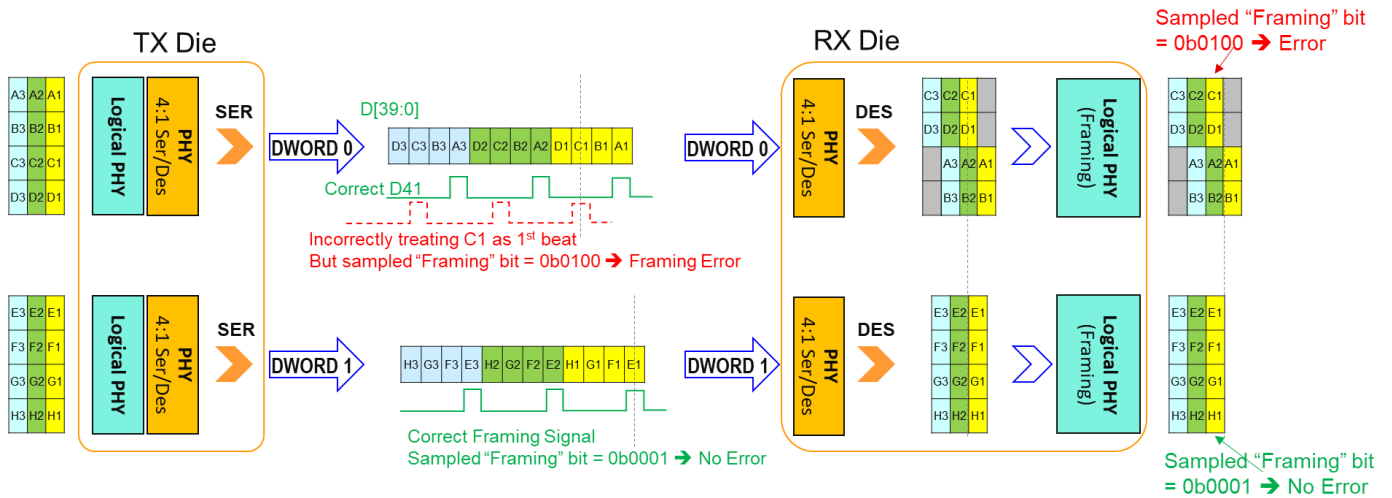


Figure 7-2: Example diagram showing correct and incorrect Intra-DW framing

7.3.2 Inter-DW Framing/Alignment

Figure 7-3 below illustrates the upper layer can optionally utilize the "Framing" signal to align or group multiple DWORDs for use cases that require DWORD channel aggregation.

All grouped / aligned DWORD channels must be in same clock domain.

Example shown is two DWORDs (or more) where the skews between DWORD's Clock/Data/Framing signals are Non-Zero.

Each Logical PHY will perform framing normally and pass the sampled "Framing" bit 0b000.....01 to upper layer which can use to align the First Beat of traffic on each DWORD Channel.

Note: This is an Optional feature that belongs to upper layer above "Logical PHY" and is outside of OpenHBI scope.

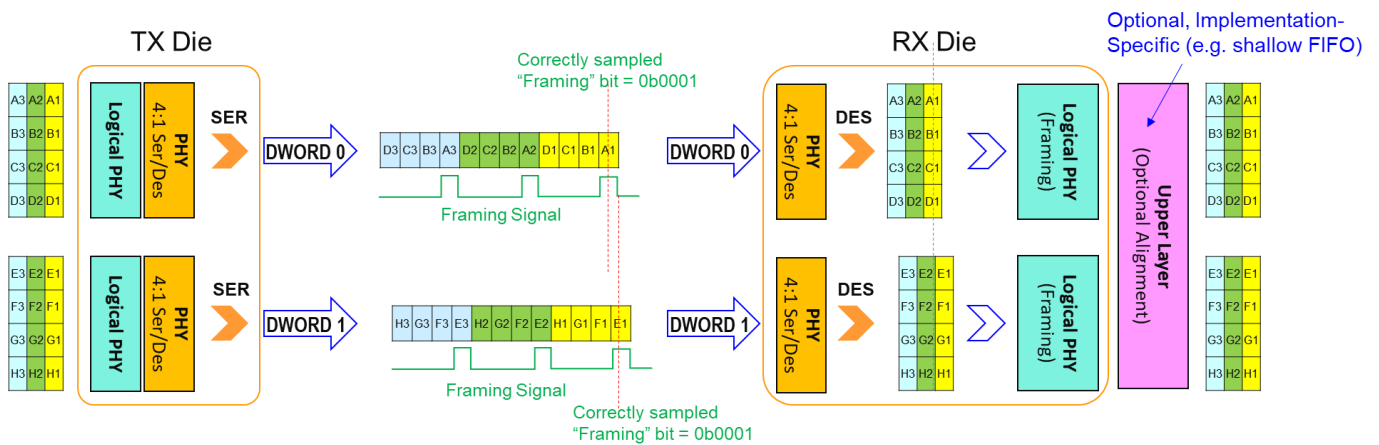


Figure 7-3: Example diagram showing correct Inter-DW framing

7.3.3 Programmable Framing pattern options

If Inter-DW skew is larger than Framing pattern period (e.g. 1ns @1GHz LogPHY clock), the upper layer can align two (or more) wide data words incorrectly as illustrated in the Figure 7-4 below.

This can be solved by optional programmable extended range of repeating pattern of “one followed by (N-1) zeros” where N = 32, 64, 128.

- where “N” can be determined by max Inter-DW skews to ensure no ambiguity in Inter-DW Alignment boundary`

The key benefit is to accommodate Inter-DW (DW-to-DW) skews. A small benefit is less toggling and slightly less power on “Framing” (D[41]) bit.

Note: This is an Optional feature that belongs to upper layer above “Logical PHY” and is outside of OpenHBI scope.

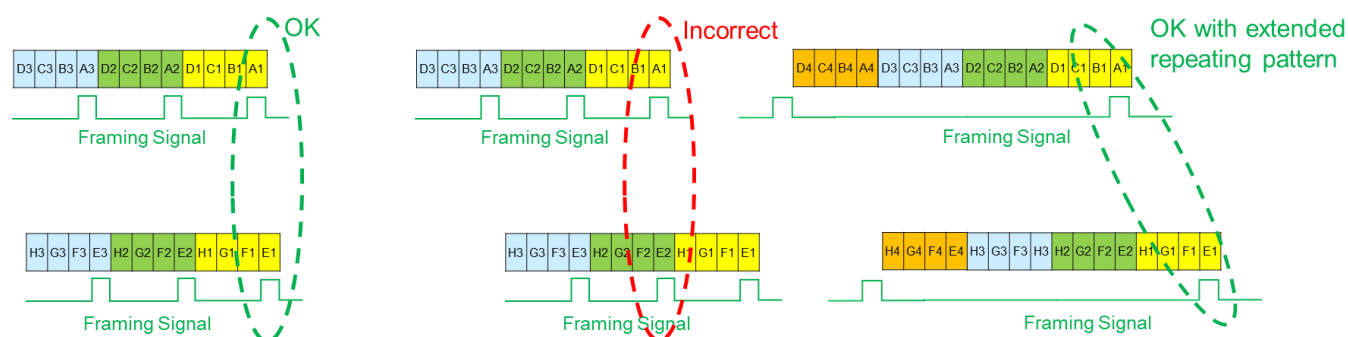


Figure 7-4: Example diagram showing how extended repeating pattern can handle large Inter-DW skew

7.4 Parity

OpenHBI Logical PHY layer provides EVEN parity protection. If enabled, each concurrent 41-bit word (consisting of D[41, 39:0] is protected with 1 parity bit (D[40]) which shall be aligned with the data.

On the TX side, parity is computed on all 41 bits after DBI. On the RX side, parity is computed on all 41 bits before DBI. RX side then performs parity checking and error logging on the receive (RX) side. The error(s) will be logged via error log / status registers.

OpenHBI Parity error reporting and handling are implementation-specific. If an OpenHBI chiplet supports Parity error reporting and notification and the feature is enabled, when the Chiplet RX DWORD detects a parity error, it shall generate an I3C IBI (In-Band Interrupt) with MDB to the Anchor, and the firmware / interrupt handler will check the interrupt source through I3C and handles the parity error accordingly.

The Logical PHY layer shall support the following modes

- Parity is enabled
- Parity is disabled, and the parity bit is inactive
- Parity is disabled, and the parity bit carries user/sideband data (not covered by DBI)

7.5 Options and Modes

Of the Logical PHY layer services, the following services consume interface bits and therefore affect the user-available data payload bus width

- Framing
- Parity
- DBI

These services are optional and can be disabled, freeing up additional useable payload bits. However, not all disabled service combinations (modes) are supported. Only the following Modes 0-4 in Table 7-1 are supported.

Table 7-1: Logical PHY layer modes

Mode	Framing	Parity	DBI	Payload bits
0 (default)	Enabled	Enabled	Enabled	36
1	-	-	Enabled	38
2	Enabled	Enabled	-	40
3	Enabled	-	-	41
4 (Logical PHY Bypass)	-	-	-	42

Mode 0 is the default mode in which all services are enabled. Mode 4 has all services disabled except for the optional Bit-reordering function can be controlled via CCS in Logical PHY layer bypass mode.

Note: Bit-reordering function can be independently enabled or disabled for all Mode 0-4. It does not affect the number of bits available to the upper layer.

7.6 Interface Pin Assignments per Mode

Depending on the mode, the pin assignment visible to the Protocol layer will be different, as illustrated in the table below. The Logical PHY layer is responsible for multiplexing the bits as needed to present this simple interface to the protocol layer.

Table 7-2: Pin assignment per mode

Mode	Gearbox 2:1	4:1	8:1	16:1
0 (Frm+Par+DBI)	Bit [71:0]	Bit [143:0]	Bit [287:0]	Bit [575:0]
1 (DBI)	Bit [75:0]	Bit [151:0]	Bit [303:0]	Bit [607:0]
2 (Frm+Par)	Bit [79:0]	Bit [159:0]	Bit [319:0]	Bit [639:0]
3 (Frm)	Bit [81:0]	Bit [163:0]	Bit [327:0]	Bit [655:0]
4 (Bypass)	Bit [83:0]	Bit [167:0]	Bit [335:0]	Bit [671:0]

8. Orientation and Routing

8.1 Orientation-Independent Routing

When connecting two OpenHBI chiplets together, there are orientation and routing considerations as discussed in Section 7.1.

OpenHBI specification supports orientation-independent chiplet-to-chiplet routing, True (i.e., native) orientation or R180 (i.e., Rotated 180°) orientation and their combinations (T-T, T-R180, R180-R180, R180-T) by exploiting DWORD-level symmetry and by means of the optional Logical PHY layer “Bit-reordering” functions and the PHY layer “Signal Swap” functions.

Logical PHY layer optional “Bit Reordering with Bypass option” :

- Bit Reordering Mode 0 (refer to Section 8.2)
 - No reordering. Bypass bit reordering mux, With or Without lane repair (in PHY layer)
- Bit Reordering Mode 1(a)-(e) (refer to Section 8.3)
 - Mode 1(a): Rotated Orientation, No lane repair (Section 8.3.1)
 - Mode 1(b)-1(e): Rotated Orientation, Lane Repair on TX byte 0, 1, 2, 3 respectively (Section 8.3.2)

PHY layer optional “Signal Swap” functions :

- Optional WDQS and RDQS Clock pairs swap
- Optional $_t_c$ (true/complement) differential Polarity swap for WDQS and RDQS
- Optional RD0 <-> D5 and RD1 <-> D36 swap

Section 8.2 and 8.3 describe the Same-Orientation and Rotated-Orientation routing respectively.

If RX PHY layer supports the swap functions, it shall perform the swap. If TX PHY layer also supports WDQS <-> RDQS clock pair swap and $_t_c$ polarity swap functions, then the combination can support the optional Clock lane repair feature as described in Section 8.4.

Figure 8-1 below shows the True (left) and R180 (right) orientation virtual bump map respectively.

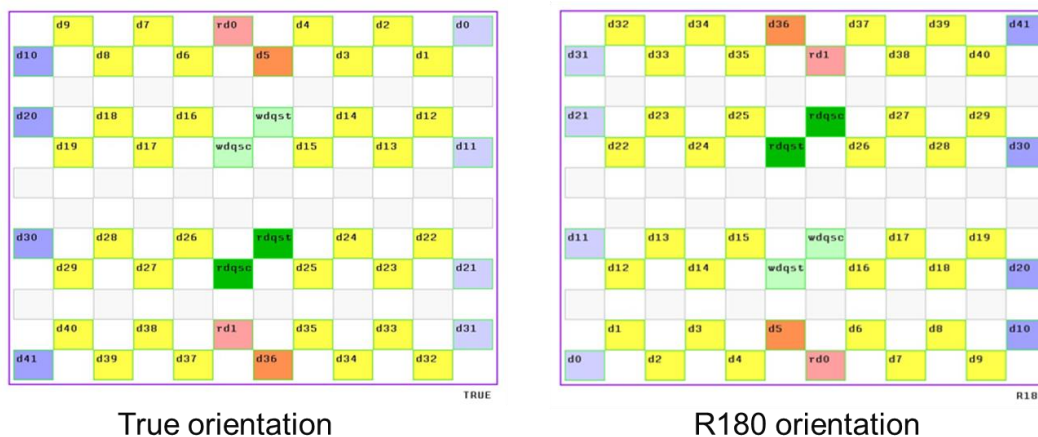


Figure 8-1: True (left) and R180 (right) orientation virtual bump map

8.2 Same Orientation Routing – Bit Reordering Mode 0 (bypass, no reordering)

Two C2C beachfronts are considered to be in “Same Orientation” when they are connected in True-True or R180-R180 orientation. Figure 8-2 below illustrates the concept.

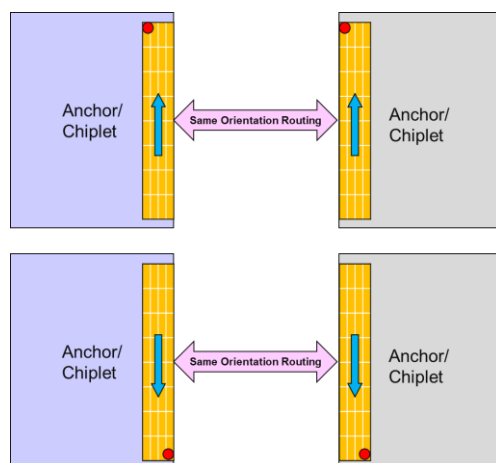


Figure 8-2: Two C2C beachfronts in SAME orientation

In OpenHBI, TX DWORD WDQS (TX_Clk) connects to RX DWORD RDQS (RX_Clk).

Figure 8-3 illustrates the Same Orientation connections when both TX and RX are in True orientation or when both are in R180 orientation. In Same Orientation, all signals connect seamlessly (straight connections) except WDQS_t/_c and RDQS_t/_c pairs connections can cause crisscross routing on the interposer as shown in the left diagram of Figure 8-3.

RX PHY may support the optional WDQS <-> RDQS clock pairs swap function to eliminate the crisscrosses as shown in the right diagram of Figure 8-3. Note that, interposer routing may also be able to mitigate the crisscrosses.

- TX RDQS and RX WDQS are not used typically and can be left as “No Connect” as long as the RDQS inputs on TX side are pulled to proper valid state and not floated.
- If RX DWORD supports clock pairs swap function, it shall perform the clock pairs swap.
- For Bit Reordering Mode 0, TX DWORD can optionally support WDQS <-> RDQS swap.
- For bidirectional configurable DWORD, it is strongly recommended to support WDQS <-> RDQS swap function.

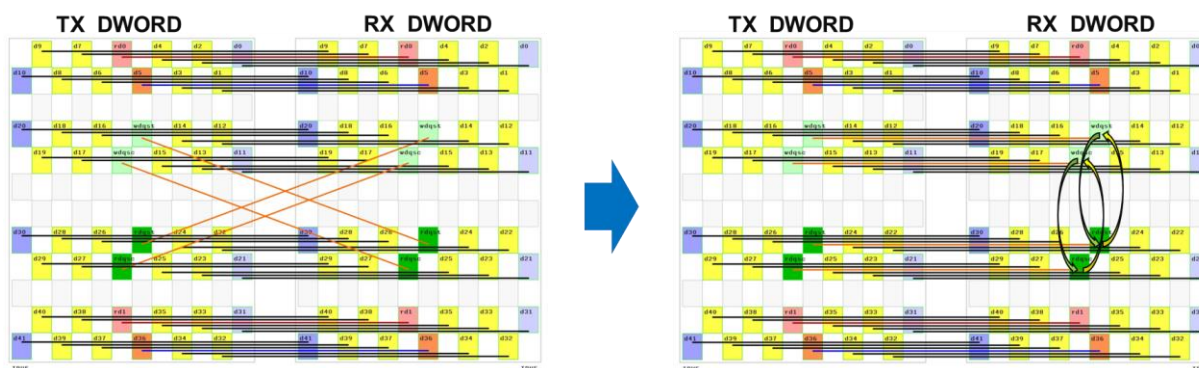


Figure 8-3: Same Orientation connections. Optional WDQS <-> RDQS swap

Table 8-1 shows the connection map when connecting the TX and RX DWORD in all straight connections, with the PHY layer of the RX DWORD performing the WDQS <-> RDQS pair swap.

Table 8-1: Same Orientation Connection Map

TX DW	RX DW	TX DW	RX DW	TX DW	RX DW	TX DW	RX DW	TX DW	RX DW
d0	d0	d11	d11	d21	d21	d31	d31	rd0	rd0
d1	d1	d12	d12	d22	d22	d32	d32	rd1	rd1
d2	d2	d13	d13	d23	d23	d33	d33	d5	d5
d3	d3	d14	d14	d24	d24	d34	d34	d36	d36
d4	d4	d15	d15	d25	d25	d35	d35	wdqst	wdqst
d6	d6	d16	d16	d26	d26	d37	d37	wdqsc	wdqsc
d7	d7	d17	d17	d27	d27	d38	d38	rdqst	rdqst
d8	d8	d18	d18	d28	d28	d39	d39	rdqsc	rdqsc
d9	d9	d19	d19	d29	d29	d40	d40		
d10	d10	d20	d20	d30	d30	d41	d41		

Logical PHY layer (typ. RX DW):
For same orientation, no reordering required

PHY layer (typ. RX DW):
For same orientation, can support optional WDQS <-> RDQS pairs swap to eliminate crisscrosses

8.3 Rotated Orientation Routing – Bit Reordering Mode 1

Two C2C beachfronts are considered to be in “Rotated Orientation” when they are connected in True-R180 or R180-True orientation. Figure 8-4 below illustrates the concept.

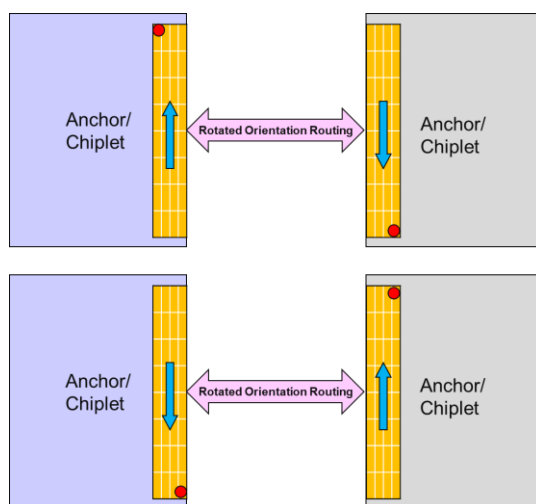


Figure 8-4: Two C2C beachfronts in ROTATED orientation

As shown in left diagram in Figure 8-5, almost all signals will be crisscrossed without bit reordering support. If the Anchor and Chiplet desire to support Rotated Orientation use case, the RX DWORD Logical PHY needs to support the optional Bit Reordering Mode 1 in order to be routable.

As shown in right diagram in Figure 8-5, even with bit reordering support, the RX PHY may need to support the optional `_t/_c` polarity swap and RD0<->D5 and RD1<->D36 swap functions in order to achieve no crisscross in rotated orientation.

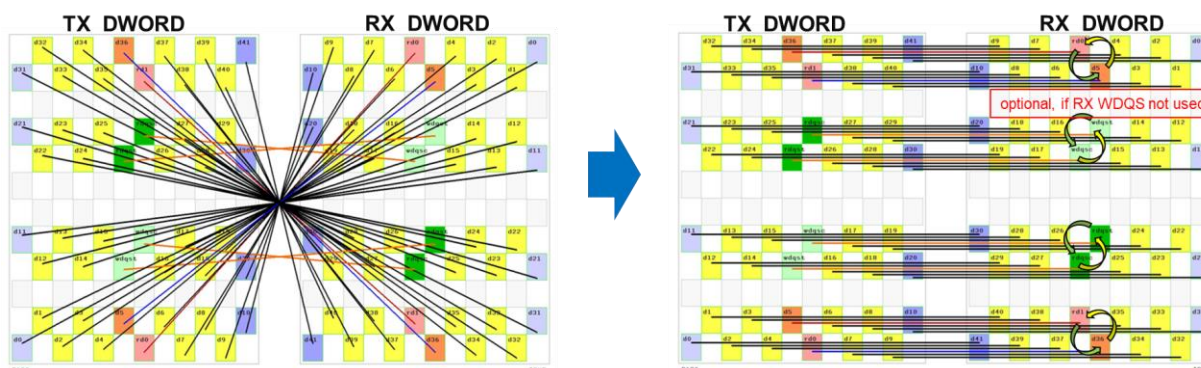


Figure 8-5: (left) Rotated Orientation connections. (right) Straight connections with bit-reordering

If lane repair is required, the bit reordering mux will need additional sub-modes to support seamless (straight) connections for all combinations of where the defective connection(s) are.

The 5 sub-modes of Bit Reordering Mode 1 are described in Section 8.3.1 and 8.3.2.

- Mode 1(a): Rotated, No Lane Repair
- Mode 1(b): Rotated, Lane Repair in TX Byte 0 / RX Byte 3
- Mode 1(c): Rotated, Lane Repair in TX Byte 1 / RX Byte 2
- Mode 1(d): Rotated, Lane Repair in TX Byte 2 / RX Byte 1
- Mode 1(e): Rotated, Lane Repair in TX Byte 3 / RX Byte 0

If the optional Bit Reordering Mode 1 is supported, all 5 sub-modes Mode 1(a) - 1(e) shall be supported.

8.3.1 Rotated Orientation - Bit Reordering Mode 1(a)

The Mode 1(a) should be applied if the TX DWORD and RX DWORD pair are in rotated orientation but do not require any lane repair.

Table 8-2 below shows the connection map when connecting the TX and RX DWORD in all straight connections. In this sub-mode 1(a), the RX PHY layer may support the following optional features:

- `_t/_c` (true/complement) differential polarity swap within WDQS and RDQS pair
- RD0 <-> D5 and RD1 <-> D36 swap which will result in RD0-to-RD1 and D5-to-D36 connections which then get reordered by Logical PHY Bit Reordering function

Table 8-2: Rotated Orientation Connection Map

TX DW	RX DW	TX DW	RX DW	TX DW	RX DW	TX DW	RX DW	TX DW	RX DW
d0	d41	d11	d30	d21	d20	d31	d10	rd0	d36
d1	d40	d12	d29	d22	d19	d32	d9	rd1	d5
d2	d39	d13	d28	d23	d18	d33	d8	d5	rd1
d3	d38	d14	d27	d24	d17	d34	d7	d36	rd0
d4	d37	d15	d26	d25	d16	d35	d6	wdqst	rdqsc
d6	d35	d16	d25	d26	d15	d37	d4	wdqsc	rdqst
d7	d34	d17	d24	d27	d14	d38	d3	rdqst	wdqsc
d8	d33	d18	d23	d28	d13	d39	d2	rdqsc	wdqst
d9	d32	d19	d22	d29	d12	d40	d1		
d10	d31	d20	d21	d30	d11	d41	d0		

Logical PHY layer (typ. RX DW):

For rotated orientation, Logical PHY can reorder most signals to eliminate crisscrosses, except RD0, RD1, causing local crisscrosses with D[5] and D[36]

PHY layer (typ. RX DW):

For rotated orientation, RX PHY can optionally support $_t_c$ polarity swap on its WDQS and RDQS pairs to eliminate crisscrosses

Figure 8-6 below illustrates the sequence of events from TX DWORD (on the left) towards RX DWORD (on the right) of one of the 10-bit byte group of a DWORD. The TX Byte 0 to RX Byte 3 is shown. The TX lane repair mux ("B") and RX lane repair mux ("F") will be bypassed, and Bit Reorder Mux ("H") will perform the bit reordering map as shown in Table 8-3 by reordering the input "G" to output "J".

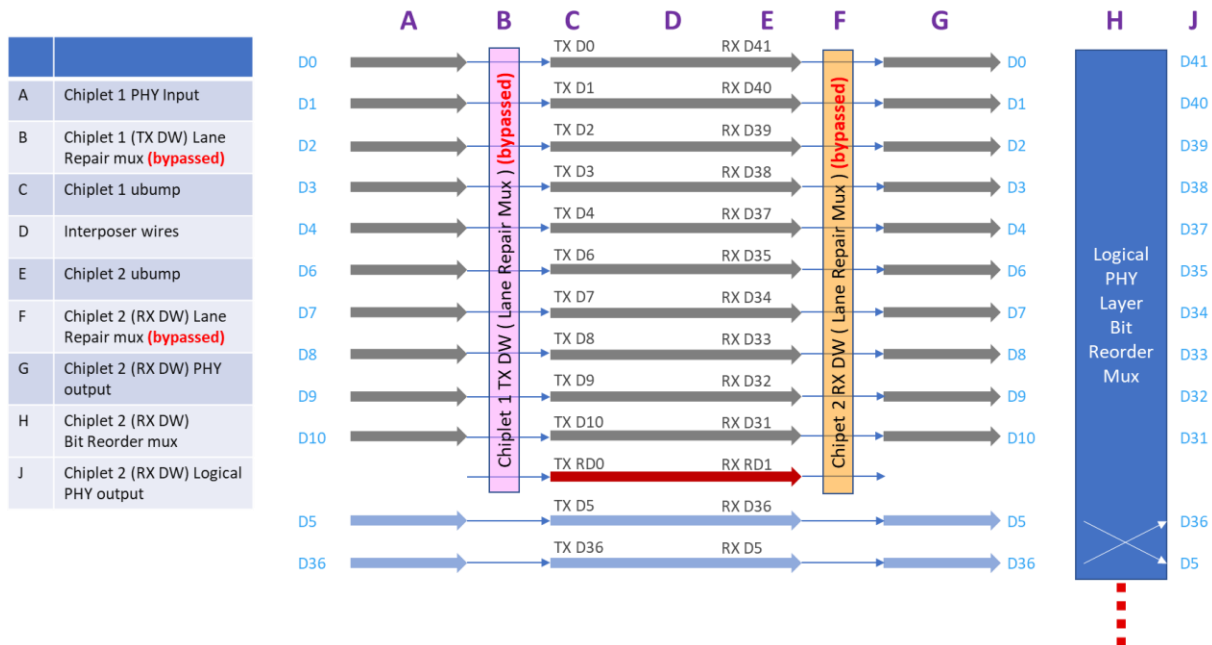


Figure 8-6: Event sequence from TX-to-RX with Lane repair and Bit Reordering Mode 1(a)

Table 8-3: Mode 1(a) signal reordering performed by the Bit Reorder Mux (“H”)

In “G”	Out “J”	In “G”	Out “J”	In “G”	Out “J”	In “G”	Out “J”	In “G”	Out “J”
d0	d41	d11	d30	d21	d20	d31	d10	d5	d36
d1	d40	d12	d29	d22	d19	d32	d9	d36	D5
d2	d39	d13	d28	d23	d18	d33	d8		
d3	d38	d14	d27	d24	d17	d34	d7		
d4	d37	d15	d26	d25	d16	d35	d6		
d6	d35	d16	d25	d26	d15	d37	d4		
d7	d34	d17	d24	d27	d14	d38	d3		
d8	d33	d18	d23	d28	d13	d39	d2		
d9	d32	d19	d22	d29	d12	d40	d1		
d10	d31	d20	d21	d30	d11	d41	d0		

Mode 1(a): Bit Reorder Map, No Lane Repair

8.3.2 Lane Repaired Bit-reordering Mode 1(b) – 1(e): Lane Repair in TX Byte n / RX Byte (3-n)

If there are defective lane connection(s), the OpenHBI lane repair mechanism can be deployed to program the lane repair “DWORD Remapping Table” to enable the signal remapping (shift) mechanism and is effectively an additional level of signal reordering. Refer to Section 6.3.5.

The Mode 1(b) to 1(e) sub-modes are used depending on which 10-bit byte group of a DWORD the defective connection is located.

Figure 8-7 below illustrates the sequence of events from TX (on the left) towards RX (on the right) of one of the 10-bit byte group of a DWORD. The TX Byte 0 to RX Byte 3 is shown.

The TX lane repair mux (“B”) and RX lane repair mux (“F”) will perform the lane repair remapping. For the example shown, the Bit Reorder Mux (“H”) will perform the Mode 1(b) bit reordering map according to Table 8-4 by reordering the input “G” to output “J”.

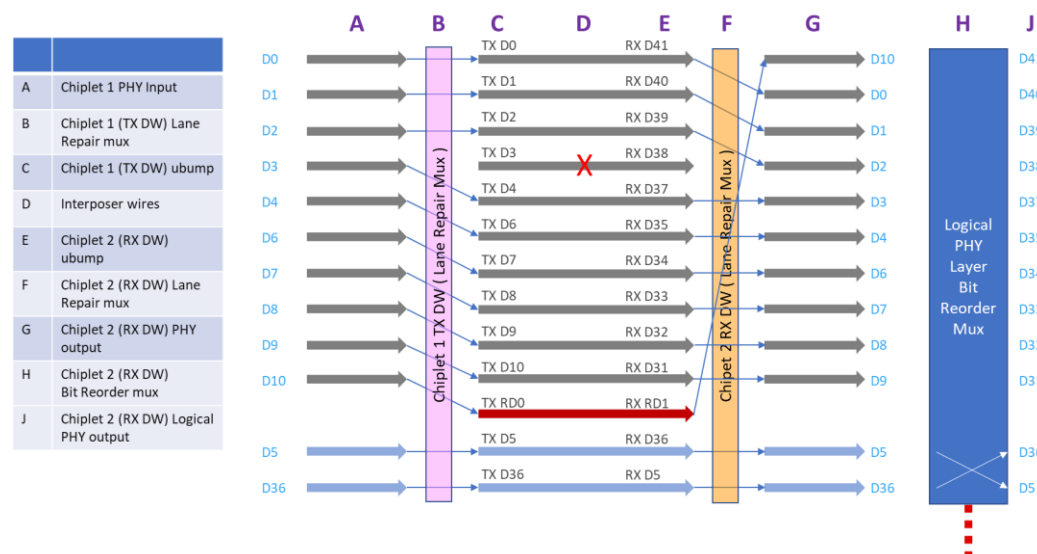


Figure 8-7: Event sequence from TX-to-RX with Lane repair and Bit Reordering Mode 1(b)

Table 8-4 to Table 8-7 summarize the Mode 1(b) – 1(e) signal reordering performed by the Bit Reorder Mux (“H”) for TX Byte (n) to RX Byte (3-n) respectively, where n=0..3.

Table 8-4: Mode 1(b) signal reordering performed by the Bit Reorder Mux (“H”)

In “G”	Out “J”	In “G”	Out “J”	In “G”	Out “J”	In “G”	Out “J”	In “G”	Out “J”
d0	d40	d11	d30	d21	d20	d31	d10	d5	d36
d1	d39	d12	d29	d22	d19	d32	d9	d36	D5
d2	d38	d13	d28	d23	d18	d33	d8		
d3	d37	d14	d27	d24	d17	d34	d7		
d4	d35	d15	d26	d25	d16	d35	d6		
d6	d34	d16	d25	d26	d15	d37	d4		
d7	d33	d17	d24	d27	d14	d38	d3		
d8	d32	d18	d23	d28	d13	d39	d2		
d9	d31	d19	d22	d29	d12	d40	d1		
d10	d41	d20	d21	d30	d11	d41	d0		

Mode 1(b): Bit Reorder Map, Rotated w/ Lane Repair in TX Byte 0 / RX Byte 3

Table 8-5: Mode 1(c) signal reordering performed by the Bit Reorder Mux (“H”)

In “G”	Out “J”	In “G”	Out “J”	In “G”	Out “J”	In “G”	Out “J”	In “G”	Out “J”
d0	d41	d11	d29	d21	d20	d31	d10	d5	d36
d1	d40	d12	d28	d22	d19	d32	d9	d36	D5
d2	d39	d13	d27	d23	d18	d33	d8		
d3	d38	d14	d26	d24	d17	d34	d7		
d4	d37	d15	d25	d25	d16	d35	d6		
d6	d35	d16	d24	d26	d15	d37	d4		
d7	d34	d17	d23	d27	d14	d38	d3		
d8	d33	d18	d22	d28	d13	d39	d2		
d9	d32	d19	d21	d29	d12	d40	d1		
d10	d31	d20	d30	d30	d11	d41	d0		

Mode 1(c): Bit Reorder Map, Rotated w/ Lane Repair in TX Byte 1 / RX Byte 2

Table 8-6: Mode 1(d) signal reordering performed by the Bit Reorder Mux (“H”)

In “G”	Out “J”	In “G”	Out “J”	In “G”	Out “J”	In “G”	Out “J”	In “G”	Out “J”
d0	d41	d11	d30	d21	d19	d31	d10	d5	d36
d1	d40	d12	d29	d22	d18	d32	d9	d36	D5
d2	d39	d13	d28	d23	d17	d33	d8		
d3	d38	d14	d27	d24	d16	d34	d7		
d4	d37	d15	d26	d25	d15	d35	d6		
d6	d35	d16	d25	d26	d14	d37	d4		
d7	d34	d17	d24	d27	d13	d38	d3		
d8	d33	d18	d23	d28	d12	d39	d2		
d9	d32	d19	d22	d29	d11	d40	d1		
d10	d31	d20	d21	d30	d20	d41	d0		

Mode 1(d): Bit Reorder Map, Rotated w/ Lane Repair in TX Byte 2 / RX Byte 1

Table 8-7: Mode 1(e) signal reordering performed by the Bit Reorder Mux ("H")

In "G"	Out "J"	In "G"	Out "J"	In "G"	Out "J"	In "G"	Out "J"	In "G"	Out "J"
d0	d41	d11	d30	d21	d20	d31	d9	d5	d36
d1	d40	d12	d29	d22	d19	d32	d8	d36	D5
d2	d39	d13	d28	d23	d18	d33	d7		
d3	d38	d14	d27	d24	d17	d34	d6		
d4	d37	d15	d26	d25	d16	d35	d4		
d6	d35	d16	d25	d26	d15	d37	d3		
d7	d34	d17	d24	d27	d14	d38	d2		
d8	d33	d18	d23	d28	d13	d39	d1		
d9	d32	d19	d22	d29	d12	d40	d0		
d10	d31	d20	d21	d30	d11	d41	d10		

Mode 1(e): Bit Reorder Map, Rotated w/ Lane Repair in TX Byte 3 / RX Byte 0

8.4 WDQS (TX_Clk) and RDQS (RX_Clk) Clock Lane Repair

If TX PHY supports WDQS <-> RDQS clock pair swap, and RX PHY supports both clock pair swap and _t/_c polarity swap functions, then the combination can support the optional Clock Lane repair feature.

For Same Orientation, only TX PHY performs the clock pair swap, and the DWORD pair will use the resulting "TX RDQS-to-RX RDQS connections" as TX_Clk / RX_Clk (within the red dotted rectangle) as illustrated in Figure 8-8 below.

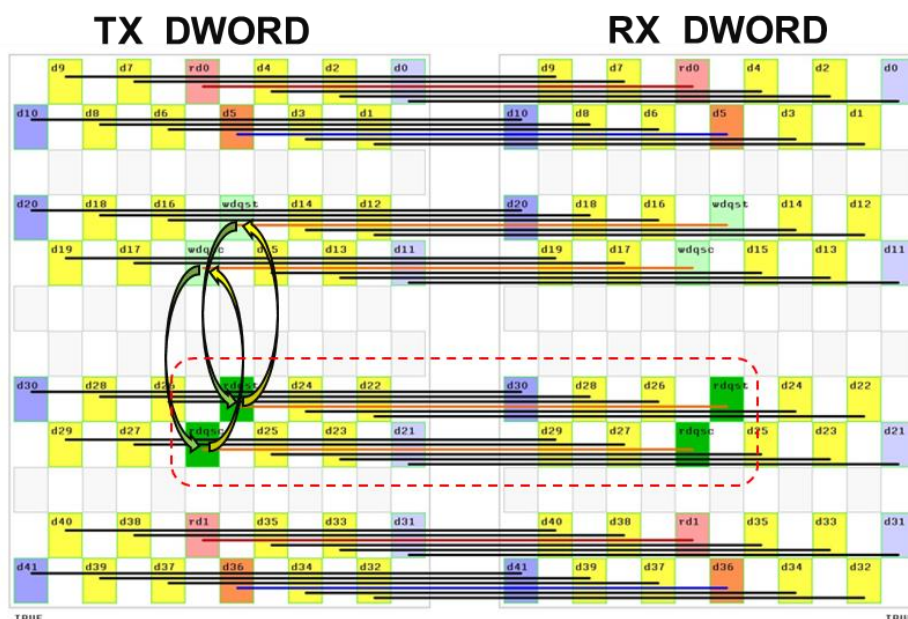


Figure 8-8: Optional WDQS / RDQS clock lane repair in Same Orientation

For Rotated Orientation, the TX PHY performs the clock pair swap, and RX PHY performs both the clock pair swap and polarity swap. The DWORD pair will use the resulting “TX RDQS-to-RX WDQS connections” as TX_Clk / RX_Clk respectively (within the red dotted rectangle) as illustrated in Figure 8-9 below.

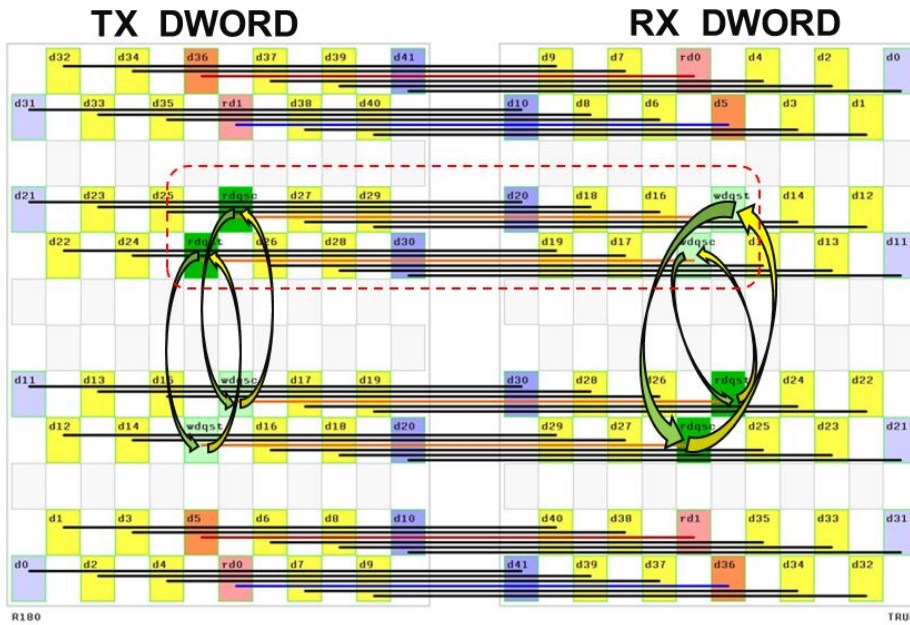


Figure 8-9: Optional WDQS / RDQS clock lane repair in Rotated Orientation

9. OpenHBI IO Electricals

OpenHBI based chiplet can interoperate with HBM3 electricals while optimized for higher data rates and energy efficiency for shorter reach, as well as possible optimization for interposer media and simpler IO designs.

The overall specs are divided into 2 categories: **Normative** and **Reference**, which are color-coded accordingly in the tables above.

Normative specifications are conformance requirements that must be met.

Reference specifications are recommendations for design reference and not a conformance requirement.

9.1 Transmitter Electrical Parameters

Definition	Min	Nominal	Max	Unit
Bit Rate	0.2		8	Gbps
IO Supply Voltage	0.38	0.4	0.42	V
Single Ended - Output High Voltage (Voh)	0.26			V
Single Ended - Output Low Voltage (Vol)			0.14	V
Maximum Driving strength	10	14	18	mA
TX timing budget*			0.24	UI

*Not including the jitter from PLL which is reflected in RX side timing budget.

Table 9-1: Transmitter Electrical Parameters

OpenHBI supports any data rate ranges from 200Mbps to 8Gbps.

The Voh and Vol are single-end voltage level defined at the far end (RX side) with maximum pin cap scenario. This needs to be satisfied to ensure interoperability.

The driving strength is defined as 14mA (nominal) to support up to 8Gbps operation, with +/- 4mA variation. Programmability could be considered to save power for lower speed operation.

The “TX timing budget” includes data/clock skew (after training), TX data-clock relative jitter effect and TX DCD effect (duty cycle distortion after correction), etc. It does not include the random jitter of the PLL and clocking path which are accounted for on the RX side. While the total budget is expected to be within 0.24UI, the spec aims to grant maximum design flexibility across the individual components listed above, while ensuring the link’s interoperability.

9.2 Receiver Electrical Parameters

Definition	Min	Nominal	Max	Unit
Bit Rate	0.2		8	Gbps
Single Ended Eye Height (peak-to-peak)	0.12			V
RX Eye Width			0.4	UI
Receiver Voltage Reference – DC Level		0.2		V
MAX PLL random jitter (1 sigma)*			0.6	ps

*random jitter refers to 5-cycle (10 UI) jitter spec

Table 9-2: Receiver Electrical Parameters

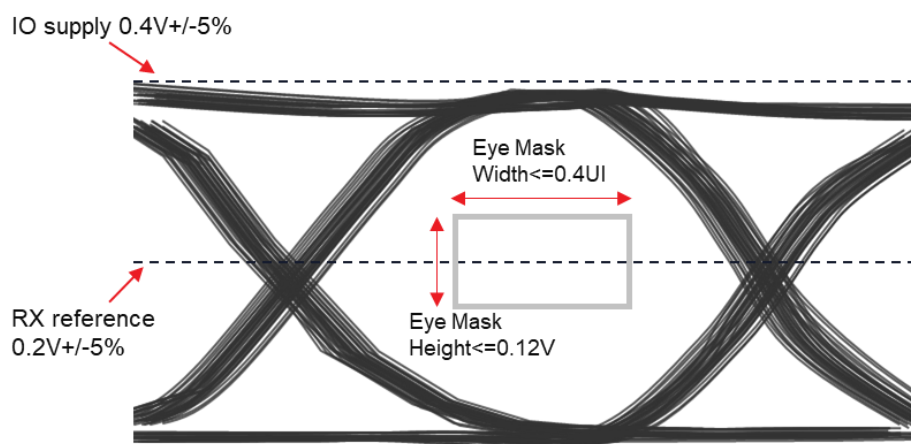


Figure 9-1: Receiver (RX) Data eye mask

Figure 9-1 illustrates the Receiver (RX) Data eye mask.

The single ended RX eye mask height (peak-to-peak) is defined to be 120mV with center around 0.2V (+/-5%) reference voltage level.

The RX eye mask width is defined to be less than or equal to 0.4 UI.

One of the key parameters defined is the max PLL random jitter. As specified in the Section 9.1 (TX Electrical Parameters), it is captured and defined in the RX section and with up to 10 UI only due to the nature of the source synchronized interface and the extra short reach and skew of the OpenHBI interface.

9.3 REFERENCE for Transceiver and Channel

Definition	MAX	Unit
Total PSIJ (peak-to-peak)	26	ps
DCD (peak-to-peak)	10	ps
Maximum channel reach	3~4	mm
Insertion loss	3	dB

Table 9-3: Reference for Transceiver and Channel

The parameters in Table 9-3 are included as a reference for design considerations.

Below describes the terms and definitions in Table 9-3:

- “PSIJ” refers to Power Supply Introduced Jitter.
- “DCD” refers to Duty Cycle Distortion.
- “Max channel reach” refers to a typical 2.5D interposer-like channel connection distance.
- “Insertion loss” refers to a typical 2.5D interposer-like channel case.

9.4 Channel Parameters

Definition	MAX	Unit
Channel timing budget (Crosstalk + ISI + Slew Rate)	0.36	UI
Total pin cap (including ESD)	0.5	pF

Table 9-4 : Channel Parameters

The channel is characterized in the transient domain in the spec to decouple it from a specific channel or packaging medium.

The “Channel timing budget” includes ISI (inter-symbol-interference), crosstalk effect and slew rate effect (degradation introduced around the top/bottom of RX eye mask). While the total budget is expected to be within 0.36UI, the spec aims to grant maximum design flexibility across the individual components listed above while ensuring the link’s interoperability.

The “Total pin cap”, including the ESD, should be designed to be within 0.5pF to ensure optimal performance of the link.

9.5 REFCLK Specifications

For best interoperability, the REFCLK should support following spec:

Definition	Value	Unit
Frequency Range	100~300	MHz
Signaling	Single-Ended	n/a

Table 9-5 : REFCLK Specifications

The reference clock (REFCLK) should be single-ended, 100MHz~300MHz to ensure interoperability.

9.6 BER spec

The BER (Bit Error Ratio) is defined as single-lane spec and the target value is 1e-25 or better. This BER will be achieved if a design is fully conformant to the OpenHBI IO electrical specifications.

OpenHBI IP designed based on this BER is expected to be sufficient for most of the use cases, including ultra large-scale AI systems without the need of any error protection and/or correction.

For mission critical designs or ultra-high reliability use cases, designers can add error protection / correction in the upper layer of OpenHBI to further enhance the data reliability.

10. Clocking, Initialization and Configurations

10.1 Clocking

This OpenHBI Clocking section is informative and is intended to illustrate a few high-level clocking example use cases.

Refer to Section 5.3, the TX_Clk and RX_Clk clock edges shall be edge-aligned with D[41:0] and RD[1:0] relative to the data eye.

Three different clocking scenarios are illustrated below:

- Mesochronous mode: Multiple or All DWORDs in a beachfront (Section 10.1.1)
- Sync mode: Single DWORD (Section 10.1.2)
- Sync mode: Two or more DWORDs (Section 10.1.3)

The Anchor and Chiplets should support Mesochronous clocking as baseline to ensure interoperability and broader use case support. Fixed function chiplets with known configurations may implement a specific clocking mode(s) as needed.

10.1.1 Mesochronous Clocking: All DWORDs can run at same clock

Figure 10-1 illustrates an example employing mesochronous clocking mode.

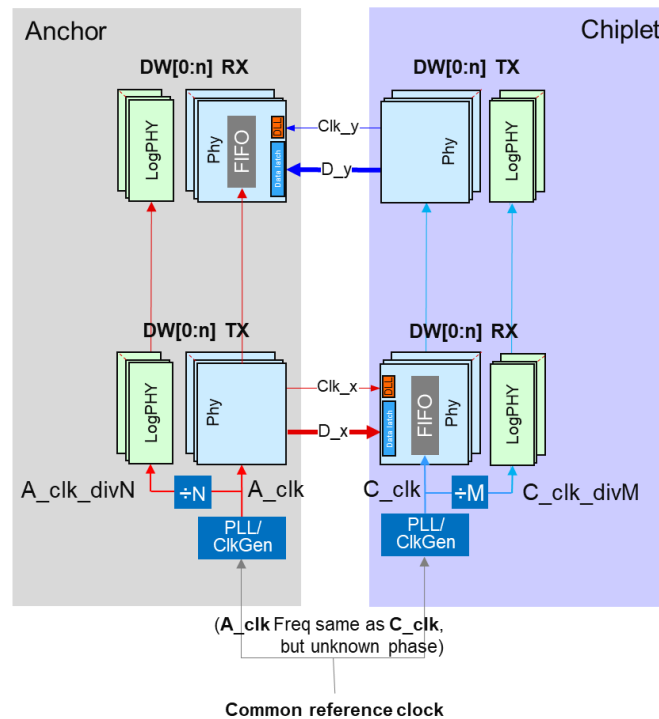


Figure 10-1: Example of Mesochronous clocking mode

Details:

- Anchor and Chiplet has its own PLLs to generate local A_Clk and C_Clk respectively, which has the same frequency but unknown relationship, based on a common clock source.
 - Note: The “Common reference clock” shown can be either fed by an external clock generator to both the Anchor and Chiplet through package C4 bumps or using CCT RefClk (Anchor: RefClk Output, Chiplet: RefClk Input).
- Anchor and Chiplet PHY layers operate as a mesochronous network
- Anchor and Chiplet each have a local clock divider to generate a local LogPHY clock. The divider factors N and M can either be the same or 2^n -multiple relationship (where $n = \text{Integer}$).
- As shown, all DWORD[n:0] on Anchor-side and Chiplet-side run at same PHY clock ($A_clk = C_clk$).
 - The divider N and M will not affect mesochronous network operation
- A FIFO in RX PHY is needed (as shown) to resolve phase uncertainty and absorbs the maximum long-term jitters between the two PLLs.

The advantage of mesochronous clocking is that all DWORD[n:0] are effectively synchronous with each other and FIFO effectively absorbs the jitters between Anchor and Chiplet PLLs and simplifies the alignment and aggregation logic in the upper layer.

10.1.2 Sync Mode: Single DWORD

Figure 10-2 below illustrates an example of fully synchronous clocking mode within a single DWORD.

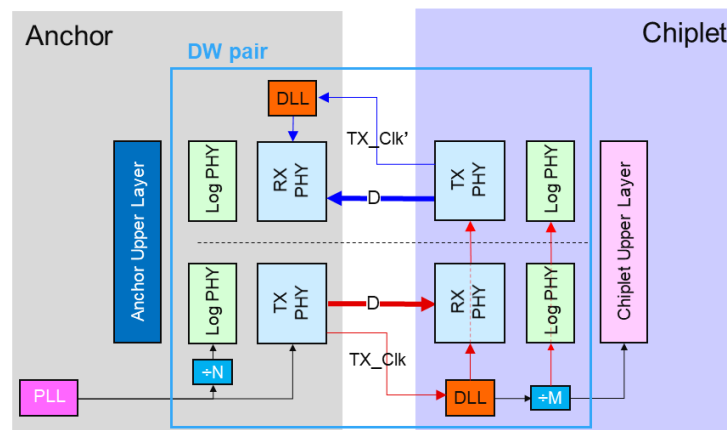


Figure 10-2: Example of Sync clocking mode: Single DWORD

Details:

- Anchor-side PLL generates a local clock which will be used to generate Anchor TX DWORD TX_Clk.
- The TX_Clk is edge-aligned with D[41:0] and RD[1:0]
- Each RX DWORD has one DLL.
- Chiplet RX DWORD trains the incoming RX_Clk with local DLL to generate a sampling clock centered with the data eye to capture the data
- Anchor TX and Chiplet RX each have a local clock divider to generate a local LogPHY clock. The divider factors N and M can either be the same or 2^n -multiple relationship (where $n = \text{Integer}$).
- Optionally, the Chiplet-side trained sampling clock and the “divided-by-M” LogPHY clock can be used by Chiplet TX DWORD, as shown in red dotted lines.

- Anchor RX DWORD will train the incoming RX_Clk with the local DLL to capture the incoming data.

This clocking architecture has the advantage of lower latency end-to-end, if designed properly.

Note that, in this example, the Anchor-side TX and RX DWORDs will have the same frequency, but the phase relationship will be dependent on both Anchor and chiplet-sides implementation and needs to be resolved at the Anchor upper layer.

10.1.3 Sync Mode: Two or more DWORDs

Figure 10-3 illustrates an example of fully synchronous clocking mode with an upper layer that uses two or more DWORDs.

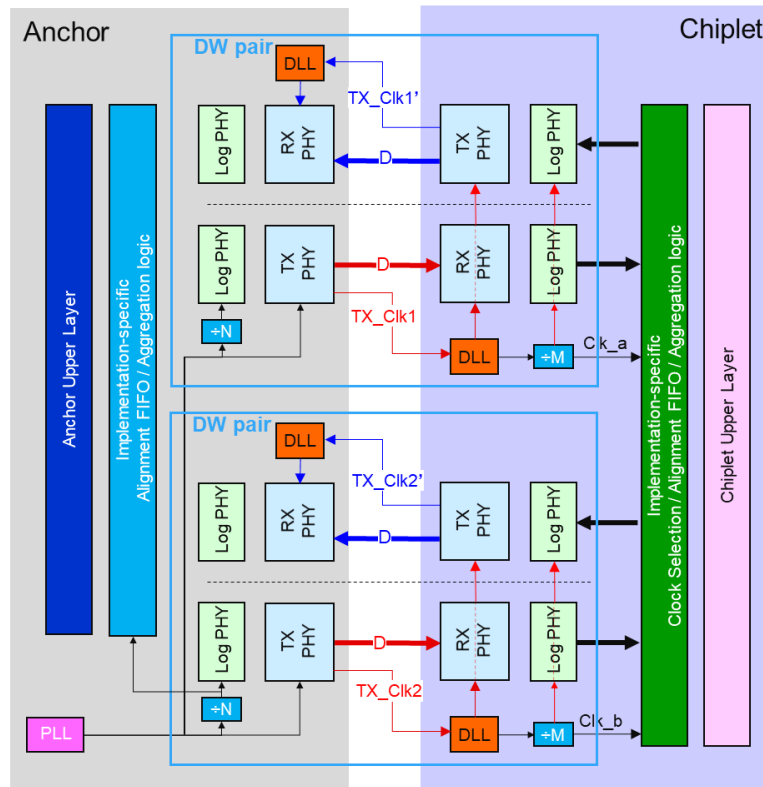


Figure 10-3: Example of Sync clocking mode: 2 or more DWORDs

Details:

- On a per-DWORD pair basis, the clocking details is the same as described in Section 10.1.2.
- Chiplet-side has two RX DWORDs which will output Clk_a and Clk_b respectively by local clock dividers. Clk_a and Clk_b can have a certain amount of skew.
- Chiplet-side has the Clock Selection logic (to select between Clk_a and Clk_b to be used for alignment and aggregation), Alignment FIFO and DWORD Channel Aggregation logic before presenting to the Chiplet upper layer.
- Similarly, Anchor side has an Alignment FIFO and DWORD Channel Aggregation logic before presenting to the Anchor upper layer.

This example is an extension of Section 10.1.2 and has the same advantage of low end-to-end latency, if designed properly.

10.2 Initialization Flow

All DWORDs of an OpenHBI instance must be properly powered up and initialized. The following high-level sequence and timing must be satisfied for the OpenHBI instance power up and initialization sequence. Please refer to Figure 10-4 and Table 10-1. Note: In the following, it is assumed chiplet power rails ramp up will not be later than Anchor power rails ramp up to stable state at t_0 .

1. Apply power to the OpenHBI instance supplies. During power supply ramp, PwrGood is de-asserted (LOW), Reset (i.e., CCT Reset) and all other input signals may be in an undefined state (driven LOW or HIGH, or Hi-Z).
2. After power stable (t_0), Anchor drives PwrGood and Reset HIGH. All other input signals may be in an undefined state (driven LOW or HIGH, or Hi-Z) at this point. Anchor shall drive PwrGood HIGH during the initialization flow and as long as stable power is maintained. Anchor shall drive Reset HIGH and maintain for a minimum of t_{INIT1} time. When Reset is asserted, each Anchor and Chiplet OpenHBI DWORD must drive TX_Clk_t to LOW and TX_Clk_c to HIGH respectively (TX_Clk = logical LOW state) no later than t_{INIT2} (min) before the Anchor de-asserts Reset (i.e., no later than ($t_{INIT1} - t_{INIT2}$) from Anchor asserting Reset HIGH). Anchor must drive RefClk and RefClk2 to a stable frequency no later than t_{INIT2} (min) before de-asserting Reset. RX_Clk_t and RX_Clk_c are input and are driven by another anchor or chiplet.
3. Wait for Chiplet ready to be configured (e.g., PLL or DLL settling time, chiplet specific).
4. Configure each OpenHBI DWORD as described in Section 10.3
5. Perform Training and Calibration of each OpenHBI DWORD as described in Section 10.5
6. The OpenHBI DWORDs are now ready for Mission mode.

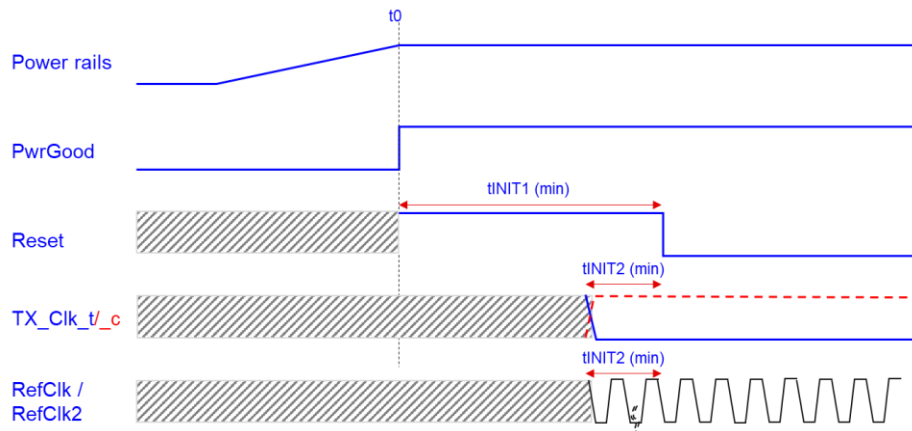


Figure 10-4: Initialization Flow Timing Parameters

Table 10-1: Initialization Timing Parameters

Parameter	Min	Unit
t_{INIT1}	200	usec
t_{INIT2}	10	nsec

10.3 Capability, Control and Status (CCS) Registers

Section 10.3.1 describes a normative set of I3C Bus and Device Characteristics Registers (BCR and DCR).

Section 10.3.2 to 10.3.5 describes an example set of registers of an OpenHBI chiplet. All Capability, Control and Status Registers, collectively called CCS Registers, must be accessible via CCT I3C interface by system firmware. The I3C address space is mapped into the device address space. Only the lower eight bits of the address are specified by I3C. The layout of the Chiplet and Anchor CCS Registers are implementation-specific.

Section 10.4 to 10.6 illustrates the typical initialization and training sequence. The registers and bit fields described are from the perspective of Chiplets (as configuration subordinate, with Anchor as configuration manager). All “TX” and “RX” described in this section imply “Chiplet-side TX DWORD” and “Chiplet-side RX DWORD” respectively, unless otherwise stated (e.g., Anchor-side TX, or Anchor-side RX).

For the sake of clarity, the flow described below applies to one DWORD only. All DWORDs in an OpenHBI instance must have been initialized, configured, have detected broken lanes and performed lane repair if needed, and have been trained and calibrated before it can enter Mission mode. It is recommended that all DWORDs can be configured and trained in parallel to reduce initialization time, but it is allowed to perform each DWORD or groups of DWORDs sequentially.

Note: The registers and the bit fields described in the examples below are not meant to be exhaustive and do not include all the necessary parameters during the actual sequence.

10.3.1 Required I3C Characteristics Registers

This section describes the normative I3C Characteristics Registers that are required for all I3C compliant devices. This includes the Bus Characteristics Register (BCR) and the Device Characteristics Register (DCR). For interoperability across OpenHBI devices, the settings of these registers must be as defined in this section. Note that the I3C Legacy Virtual Register (LVR) is not included in this section because OpenHBI devices are defined as I3C compliant and not as legacy I²C devices.

10.3.1.1 Bus Characteristics Register (BCR)

This is used to describe the I3C compliant device’s role and capabilities for use in dynamic address assignment and common command codes (CCCs).

Table 10-2: Bus Configuration Register (BCR)

Bits	Field Name	Attr	Description	Default
BCR[7:6]	Device role	R	2'b00 = I3C Target. 2'b01 = I3C Controller. 2'b10 to 2'b11 = Reserved for future use by MIPI.	-
BCR[5]	Advanced capabilities	R	0 = Does not support optional advanced capabilities. 1 = Supports optional advanced capabilities. This bit must be set to 1 for OpenHBI devices.	1
BCR[4]	Virtual target support	R	0 = Is not a virtual target and does not expose other downstream device(s). 1 = Is a virtual target or exposes other downstream device(s). This must be set to 0 for OpenHBI devices.	0
BCR[3]	Offline Capable	R	0 = Device will always respond to I3C bus commands. 1 = Device will not always respond to I3C bus	0

			commands. This must be set to 0 for OpenHBI devices.	
BCR[2]	IBI payload	R	0 = No data bytes follow the accepted IBI 1 = One data byte (MDB) shall follow the accepted IBI, and additional data bytes may follow. This must be set to 1 for OpenHBI devices.	1
BCR[1]	IBI request capable	R	0 – Not capable. 1 – Capable. This must be set to 1 for OpenHBI devices.	1
BCR[0]	Maximum data speed limitation	R	0 = No limitation. 1 = Limitation. This bit must be set to 0 for OpenHBI devices.	0

10.3.1.2 Device Characteristics Register (DCR)

This is used to describe the I3C compliant device type for use in dynamic address assignment and common command codes (CCCs).

Table 10-3: Device Characteristics Register (DCR)

Bits	Field Name	Attr	Description	Default
DCR[7:0]	Device ID	R	This must be set to 0x00 for OpenHBI devices to specify a Generic I3C Device.	0x00

10.3.2 Recommended (Informative) Control and Status Registers

This section describes a reference set of configuration control and status registers used during initialization, training and calibration sequences as shown in Section 10.5. For the sake of simplicity, the capability registers that the configuration manager can use to discover the chiplet's capabilities before starting the training and calibration sequence are not described in this section.

10.3.2.1 DWORD Configuration Register

This is used to configure the DWORD. There is one copy of this register for each DWORD.

Table 10-4: DWORD Configuration Register

Width (bit)	Field Name	Attr	Description	Default
1	TX/RX	RW	A high level "1" indicates this DWORD is TX A low level "0" indicates this DWORD is RX	0

10.3.2.2 TX - Training and Test Control Register (TX-TTCR)

This is used to control all chiplet-side TX DWORDs of the link during training and testing phase. There is one copy of this register for each OpenHBI instance.

Table 10-5: TX-TTCR (TX - Training and Test Control Register)

Width (bit)	Field Name	Attr	Description	Default
1	Training enable	RW	A high level "1" indicates training mode.	0
1	TX transmit start	RW	Configuration manager writes this bit to "1" to request TX to transmit data. The length of the transmitted data is controlled by the burst length register (BLR).	0

1	EXTEST enable	RW	Configuration manager writes this bit to “1” to put TX into EXTEST mode. TX transmits static data and Anchor-side RX captures the data into Anchor’s EXTEST registers (ETR0-5).	0
1	Mission mode	RW	A high level “1” indicates Mission mode.	0

Note: “Training enable” and “Mission mode” shall not be set to “1” at the same time.

10.3.2.3 RX - Training and Test Control Register (RX-TTCR)

This is used to control all chiplet-side RX DWORDs of the link during training and testing phase. There is one copy of this register for each OpenHBI instance.

Table 10-6: RX-TTCR (RX - Training and Test Control Register)

Width (bit)	Field Name	Attr	Description	Default
1	Training enable	RW	A high level indicates training mode.	0
1	RX data request	RW	RX will write this bit to indicate it is ready to receive data and request Anchor-side TX to transmit data. Chiplet can generate an I3C IBI (In-Band Interrupt) with MDB to Anchor, or the Anchor can check this bit. The Anchor-side TX burst length is controlled by Anchor’s burst length register which is Anchor-specific.	0
1	RX initialization done	RO	RX initialization done. If set, it indicates RX has completed all its trainings. Anchor will check this bit, or Chiplet can generate an I3C IBI with MDB when RX initialization is done.	0
1	EXTEST enable	RW	Configuration manager writes this bit to “1” to put RX into EXTEST mode. Anchor-side TX transmits static data and RX captures the data into the EXTEST registers (ETR0-5).	0
1	RX training error	RO	RX will set when there is an error after “RX initialization done” bit is set.	0
1	Mission mode	RW	A high level “1” indicates Mission mode.	0

Note: “Training enable” and “Mission mode” shall not be set to “1” at the same time.

10.3.2.4 MISR/LFSR Control Register (MLCR)

The MISR/LFSR Control Register is used to configure the OpenHBI MISR/LFSR. There is one copy of this register for each OpenHBI instance.

Table 10-7: MISR/LFSR Control Register

Width (bit)	Name	Attr	Description	Default
1	MISR/LFSR enable	RW	Enables OpenHBI MISR/LFSR. 0 = Disabled. 1 = Enabled.	0
3	MISR/LFSR control	RW	Selects the MISR/LFSR mode. 0 = Preset (0xAAAAAAAA) 1 = LFSR mode. 2 = Register mode. 3 = MISR mode. 4 = LFSR Compare mode. 5–7 = Reserved.	0

10.3.2.5 Burst Length Register (BLR)

The burst length register (BLR) specifies the burst length of chiplet-side TX data transmission during training and calibration (sent to the Anchor-side RX). There is one copy of this register for each OpenHBI instance.

Table 10-8: Burst Length Register

Width (bit)	Name	Attr	Description	Default
32	Burst length (BL)	RW	Specifies the number of data beats for each RX data request. Number of data beats transmitted = $8 \times (BL+1)$.	0

10.3.2.6 DWORD Address Register (DWAR)

The DWORD address register (DWAR) is used to specify the address of the DWORD from which the I3C register read accesses. There is one copy of this register for each OpenHBI instance.

Table 10-9: DWORD Address Register

Width (bit)	Name	Attr	Description	Default
8	DWORD address	RW	Specifies the DWORD number accessed during I3C register read access. Valid ranges: <ul style="list-style-type: none"> - Full instance: 0x00 to 0x1F - Half instance: 0x00 to 0x0F - Quarter instance: 0x00 to 0x07 All other values = Reserved.	0x00

10.3.2.7 EXTEST Register 0 to 5 (ETR0-5)

The EXTEST registers (ETR0-5) are used to capture the data during an EXTEST. The ordering of forty-eight signals in {ETR5...ETR0} can be implementation-specific but it is recommended to follow the signal ordering of [Reference 1] EXTEST_TX/RX WDR. These registers are only used when the DWORD is configured as RX. There is a separate copy of these registers for each DWORD.

Table 10-10: EXTEST Registers

Width (bit)	Name	Attr	Description	Default
8	ETR0	RW	EXTEST data	0x00
8	ETR1	RW	EXTEST data	0x00
8	ETR2	RW	EXTEST data	0x00
8	ETR3	RW	EXTEST data	0x00
8	ETR4	RW	EXTEST data	0x00
8	ETR5	RW	EXTEST data	0x00

10.3.2.8 LFSR Compare Sticky Registers (LCSR0-4)

This is to return the status of the LFSR sticky bits. The ordering of the forty LFSR sticky bits can be implementation-specific but it is recommended to follow the bit ordering of [Reference 1] READ_LFSR_COMPARE_STICKY WDR. There is a separate copy of these registers for each DWORD. These registers are only used when the DWORD is configured as RX.

Table 10-11: LFSR Compare Sticky Registers

Width (bit)	Name	Attr	Description	Default
8	LFSR sticky bit register 0	RO	Return the LFSR sticky bits.	0x00
8	LFSR sticky bit register 1	RO	Return the LFSR sticky bits.	0x00
8	LFSR sticky bit register 2	RO	Return the LFSR sticky bits.	0x00
8	LFSR sticky bit register 3	RO	Return the LFSR sticky bits.	0x00
8	LFSR sticky bit register 4	RO	Return the LFSR sticky bits.	0x00

10.3.2.9 MISR/LFSR Data Registers (MLR0-4)

This is to return the status of the MISR/LFSR data register bits for MISR Mode and Register Mode. Same bit ordering as [Reference 1]. There is a separate copy of these registers for each DWORD. These registers are only used when the DWORD is configured as RX.

Table 10-12: MISR/LFSR Data Registers

Width (bit)	Name	Attr	Description	Default
8	MISR/LFSR register 0	RO	Return the MISR/LFSR bits.	0x00
8	MISR/LFSR register 1	RO	Return the MISR/LFSR bits.	0x00
8	MISR/LFSR register 2	RO	Return the MISR/LFSR bits.	0x00
8	MISR/LFSR register 3	RO	Return the MISR/LFSR bits.	0x00
8	MISR/LFSR register 4	RO	Return the MISR/LFSR bits.	0x00

10.3.2.10 Lane Repair Registers (LRR0-3)

This register specifies the DWORD lane repair. There is a separate copy of these registers for each DWORD. Each 4-bit nibble has same encoding as DWORD Remapping Table in [Reference 1]. Refer to Section 6.3.5 for details of which data bits are covered by RD0 and RD1 if lane repair is enabled.

Table 10-13: Lane Repair Register

Width (bit)	Name	Attr	Description	Default
4	Lane repair register 0	RW	Specifies the lane repair information.	0xF
4	Lane repair register 1	RW	Specifies the lane repair information.	0xF
4	Lane repair register 2	RW	Specifies the lane repair information.	0xF
4	Lane repair register 3	RW	Specifies the lane repair information.	0xF

10.3.3 Additional Recommended Capability Registers

This section describes a set of additional recommended capability registers per OpenHBI instance to allow a configuration manager to discover the chiplet's capabilities in order to properly configure it. The list below is for reference only and not meant to be exhaustive.

Top level capabilities:

- Full / Half / Quarter instance

- Dual Mode Host support: Supported / Not supported (Anchor-side only)
- Directionality: Fixed TX / Fixed RX / Configurable Bidirectional
- Bump Pattern: Chiplet True / Chiplet R180

PHY layer:

- Gearbox ratios supported
- Clock pair swap (WDQS <> RDQS swap)
- Clock differential polarity swap (_t <> _c swap)
- RD0<>D5 and RD1<>D36 swap

Logical PHY layer:

- Max Logical PHY layer clock frequency (to determine optimal Gearbox ratio)
- Framing / Programmable Extended repeating pattern support
- Parity support
- Bit reordering Mode 1 and sub-modes support
- DBI support
- Logical PHY layer modes supported – one bit per Mode 0-3
- Logical PHY layer Bypass support – Mode 4

Additional Data Integrity support

Error reporting / logging support: examples: Parity, Framing errors

10.3.4 Additional Recommended Control Registers

This section describes a set of additional recommended Control registers per OpenHBI instance to allow a configuration manager to properly configure the chiplet. The list below is for reference only and not meant to be exhaustive.

Top level Control:

- Directionality: TX or RX (if Configurable as TX or RX)
- Dual Mode OpenHBI: Enable / Disable (if the feature is supported)

Clocking / PLL related Control (if provided)

Cold Reset (toggle PwrGood signal)

Software Reset (toggle Reset signal)

PHY layer:

- Gearbox ratio control
- Clock pair swap control
- Clock differential polarity swap control
- RD0<>D5 and RD1<>D36 swap control

Logical PHY layer:

- Logical PHY layer Mode 0-3 or Bypass (which will determine Framing, Parity, DBI enabled or not)
- Framing Programmable Extended repeating pattern control
- Bit reordering Mode 1 and sub-modes control

Additional Data Integrity control

Error logging enable / Clear Error log

10.3.5 Additional Recommended Status Registers

This section describes a set of additional recommended Status and Error Logging registers per OpenHBI instance to allow a configuration manager to read the status of the Chiplet DWORD or the OpenHBI instance. The list below is for reference only and not meant to be exhaustive.

Status Registers:

- TX transmission completion status
- Per Instance or Per DWORD Training Status
- Error Log register: example, Parity, Framing errors

10.4 MISR/LFSR Test Mode

OpenHBI adopts the same DWORD MISR/LFSR (multiple-input shift register /linear feedback shift register) defined in [Reference 1] for testing and training of the die-to-die link. All the defined MISR/LFSR modes are supported.

OpenHBI shall support per DWORD 40-bit Galois LFSR logic defined in [Reference 1]. OpenHBI shall support the “LFSR Compare” feature and the detected broken lane(s) are reported via LCSR0-4 registers described in Section 10.3.2.8.

Example training or lane repair flow:

1. Software sets the number of LFSR patterns on the TX-side to generate/transmit and the RX-side to receive. The number of LFSR patterns can range from thousands to millions of transmissions.
2. Preset both TX- and RX-side 40-bit LFSR to 0xAAAAAAAAA.
3. Configure TX-side into “LFSR mode”.
4. Configure RX-side into “LFSR Compare mode”
5. Start TX-side LFSR transmission. RX-side performs search for optimal EQ setting and training.
6. After the number of LFSR patterns are sent
 - a. If training is successful, RX will set all LFSR Compare Error Sticky bits to “0”.
 - b. If training is unsuccessful, RX will set RX-TTCR “RX training error” bit = 1. Also set any LFSR Compare Error Sticky bits appropriately.
 - i. Software checks if any “sticky error bit” is set to “1” indicating broken lane(s).
 - ii. Perform lane repair (if repairable).
 - iii. Perform re-training.
7. RX will set RX-TTCR “RX initialization done” bit = 1.
8. When the training of all RX DWORDs on a chiplet is done, the chiplet may send an IBI with MDB with “TRNDONE” or “TRNDONERR” Interrupt ID to the Anchor to notify the training is complete.

10.5 Training and Calibration

This chapter describes the OpenHBI PHY training and calibration sequence needed before entering the Mission mode.

10.5.1 Example Training and Calibration Sequence

Figure 10-5 shows an example Anchor TX DWORD to Chiplet RX DWORD training and calibration sequence.

The following step is done one time for the entire OpenHBI instance:

- 1 PLL is locked. PLL will serve as the clock source.

The following steps are done for each DWORD:

- 1 TX DWORD impedance driver strength is calibrated to ensure the driver strength meets the specification.
- 2 Low speed lane repair is performed.
- 3 Anchor configures TX and RX DWORD to high speed mode and prepares for training and calibration sequence.
- 4 The high-speed trainings can be divided into two phases. The first phase includes pattern detection, clock position training, VREF leveling, and data deskewing. The TX provides a data pattern to the RX. Clock position trainings can be done by on-chip CDR circuitry or externally, for example system firmware or software. The main purpose of the VREF leveling is to minimize the duty cycle distortion of the RX. The leveling is performed by sweeping the VREF levels and finding the optimal VREF levels that makes the eye-opening the highest. Data deskewing is a training to align all D[41:0] to remove static timing skews among different lanes. At the end of this phase, RX will notify TX that it is at the end of the first phase.
- 5 The Anchor (software) will perform lane repair, if necessary, before proceeding to start the second phase.
- 6 The second phase of high-speed training is for optimal clock positioning for data capture and configuring channel equalizing circuits. TX provides an LFSR pattern for the RX to go through the second training phase. This ensures the optimal position of the sampling edge at the center of the data eye under a given LFSR pattern. Clock position trainings can be done by on-chip CDR circuitry or externally, for example system firmware or software. EQ training and search algorithm determines the EQ settings that make the eye opening the widest. At the end of the second phase training, RX will notify TX that it is at the end of the second phase together with optional marginal lane information if the lane margin test is performed.
- 7 The Anchor (software) will perform marginal lane repair if needed.
- 8 After all the initialization sequence is complete, both TX and RX enters Mission Mode (normal operation mode).

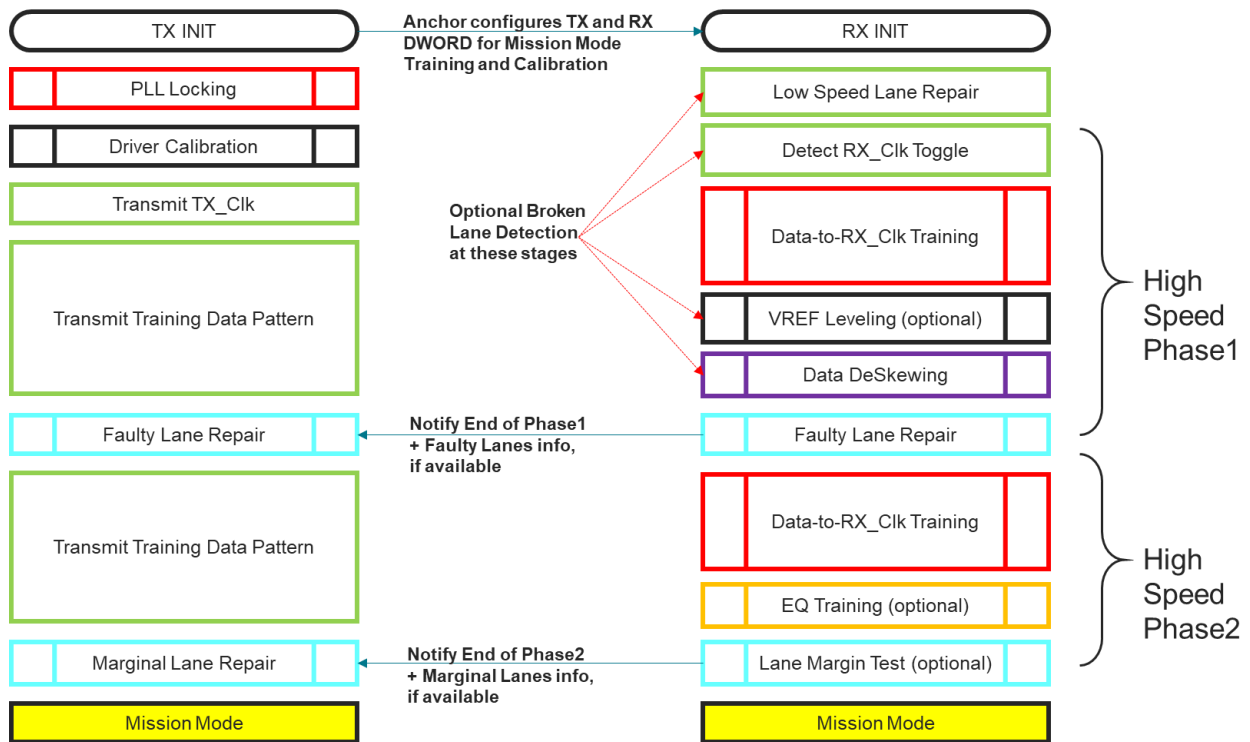


Figure 10-5: Training and Calibration Sequence

Note: In the above example, it can perform optional Broken Lane Detection prior to the “Low Speed Lane Repair” stage and can also optionally perform Broken Lane Detection during the “Detect RX_Clk Toggle”, “Data-to-RX_Clk Training” and “Data DeSkewing” stages. Refer to Section 10.5.2 for more details.

10.5.2 Broken Lane Detect

OpenHBI uses the defined LFSR and LFSR Compare features to perform broken lane detection. The broken lane detection circuits and Data-to-RX_Clk Training circuits can share logic. For training, it may optionally use other PRBS patterns instead of the defined LFSR structure.

The objective is to support any data rate between 200Mbps to 8Gbps during broken lane detection with minimal software involvement. After training success, the OpenHBI instance shall support the specified raw BER of $\leq 1E-25$. The boot time training is controlled via the I3C config interface. If necessary, optional run time to re-train can also be controlled via the I3C config interface.

Figure 10-6 below shows an example flow for broken lane detection and repair.

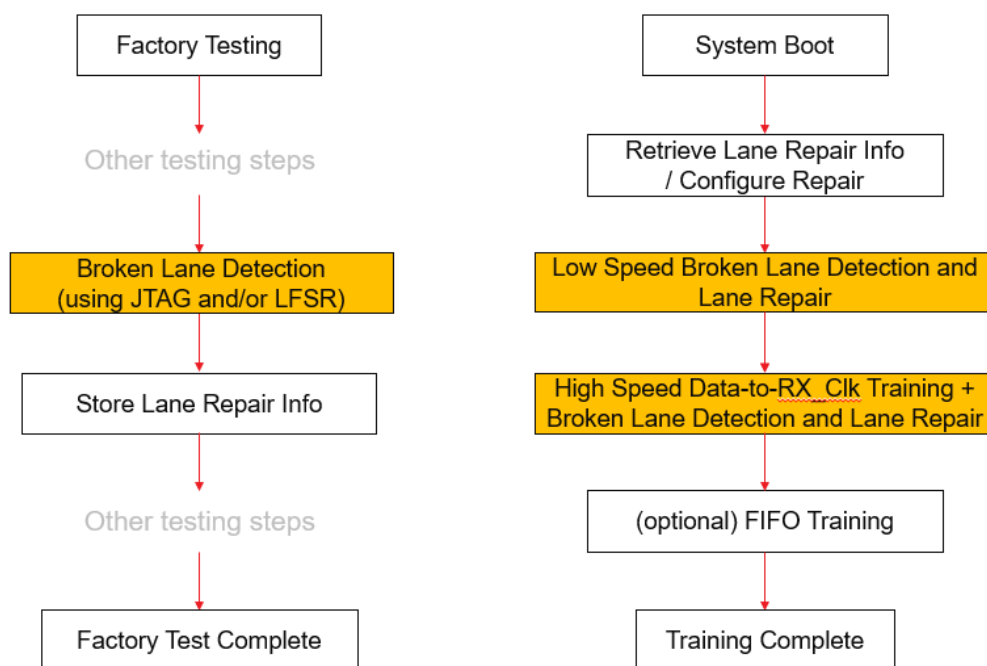


Figure 10-6: High Level Broken Lane Detection Flow

10.6 Continuous Training during operation

The OpenHBI link is conceived to require continuous uninterrupted operation. Training is performed at initialization of the link for lane deskew and sampling edge placement. Thereafter the link must continue to operate uninterrupted without training interruption. This requirement is put forth to align with the likely usage of the link to transport SerDes backhaul data traffic which also does not allow for interruption. Due to the extremely high bandwidth of the link, if the receive side were to ask for traffic interruption for retraining, an unfeasibly large memory would be required to store the unsent traffic in the transmitter. Lane status and power state information communicated between the transmit and receive ends are low bandwidth and not suitable for synchronously stopping traffic for the interruption.

The design must have sufficient margin to maintain BER with only initial deskew, VREF and sampling edge placement training, or a mechanism must be put in place to track those quantities and update them in response to voltage and temperature drift while traffic is flowing.

There are different ways to implement the continuous training mechanism during operation. Figure 10-7 illustrate an example implementation (informative) by placing a sampling circuit on the RX_Clk to continuously or periodically monitor the eye position and adjust the sampling edge, VREF or skew as needed. There is no requirement on data lines for transition density, scrambling or DC balanced patterns.

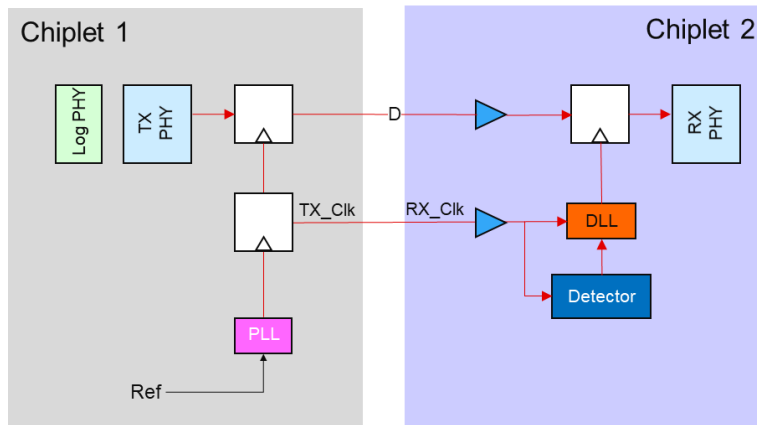


Figure 10-7: Example DLL-Detector loop for Continuous Training During Operation

11. Power Management

This version of OpenHBI only supports the following coarse grain power management mode:

- Coarse grain power management
 - It can be achieved by the Anchor or external controller terminating activities on both chips (stopping TX activities to quiesce the bus before disabling RX) and then powering down the OpenHBI link.
 - The powering up procedure from coarse grain power management will be the same as described in Section 10.2.

12. References

[1] JEDEC HBM3 DRAM v1.07a draft specification

[2] MIPI I3C[®] v1.1.1 (or later) Specifications - <https://www.mipi.org/specifications/i3c-sensor-specification>

[3] MIPI I3C BasicSM v1.0 Specifications - <https://resources.mipi.org/mipi-i3c-basic-v1-download>

Appendix A - Requirements for IC Approval (to be completed Contributor(s) of this Spec)

List all the requirements in one summary table with links from the sections.

Requirements	Details	Link to which Section in Spec
Contribution License Agreement	OWF CLA 1.0 (modified)	Please refer to Section 1
Are All Contributors listed in Sec 1: License?	Yes	
Did All the Contributors sign the appropriate license for this spec? Final Spec Agreement/HW License?	Yes	
Which 3 of the 4 OCP Tenets are supported by this Spec?	Openness Yes Efficiency Yes Impact Yes Scale Yes	Please refer to Section 2
Is there a Supplier(s) that is building a product based on this Spec? (Supplier must be an OCP Solution Provider)		
Will Supplier(s) have the product available for GENERAL AVAILABILITY within 120 days?	Seeking exception to have extended period for silicon availability. Test chips expected in 2H'2022 by multiple vendors.	

Appendix B-_____ - OCP Supplier Information (to be provided by each Supplier of Product)

Company:

Contact Info:

Product Name:

Product SKU#:

Link to Product Landing Page:

Please complete the following [2021 Supplier Requirements](#). This link will allow you to create a copy for your product-specific requirements.

For OCP Inspired™,

- All Suppliers must be a OCP Solution Provider.
- All Suppliers must run the Hardware Management Conformance Checks and all products must meet the [OCP Hardware Baseline Profile v1.0.0](#).
- All Suppliers must fill out a Security Profile (No Badge Level) for their product.

For OCP Accepted™, Supplier details are required.

- All Suppliers must be a OCP Solution Provider.
- All Suppliers must run the Hardware Management Conformance Checks and all products must meet the [OCP Hardware Baseline Profile v1.0.0](#).
- All Suppliers must fill out a Security Profile (No Badge Level) for their product.
- All Products must meet the Open System Firmware requirements.
- All Products must have source code for BMC, if applicable. This must be in the OCP Github repository.

List all the requirements in one summary table with links from the sections.

Requirements	Details	Links
Which Product recognition?	OCP Accepted™ or OCP Inspired™	Provide Marketplace Link
If OCP Accepted™, who provided the Design Package?		Link
2021 Supplier Requirements for your product(s)		Link