Universal Chiplet Interconnect Express™ (UCIe™)

Specification

August 6, 2024

Revision 2.0, Version 1.0

LEGAL NOTICE FOR THIS PUBLICLY AVAILABLE SPECIFICATION FROM UNIVERSAL CHIPLET INTERCONNECT EXPRESS, INC.

This Universal Chiplet Interconnect Express, Inc. Specification (this "**UCIe Specification**" or this "**document**") is owned by and is proprietary to Universal Chiplet Interconnect Express, Inc., a Delaware nonprofit corporation (sometimes referred to as "**UCIe**" or the "**UCIe Consortium**" or the "**Company**") and/or its successors and assigns.

NOTICE TO USERS WHO ARE MEMBERS OF UNIVERSAL CHIPLET INTERCONNECT EXPRESS, INC.:

If you are a Member of Universal Chiplet Interconnect Express, Inc. (herein referred to as a "**UCIe Member**"), and even if you have received this publicly available version of this UCIe Specification after agreeing to the UCIE Consortium's Evaluation Copy Agreement (a copy of which is available at www.uciexpress.org/specifications, each such UCIE Member must also be in compliance with all of the following UCIe Consortium documents, policies and/or procedures (collectively, the "**UCIe Governing Documents**") in order for such UCIe Member's use and/or implementation of this UCIe Specification to receive and enjoy all of the rights, benefits, privileges and protections of UCIe Consortium membership: (i) the UCIe Consortium's Intellectual Property Policy; (ii) the UCIe Consortium's Bylaws; (iii) any and all other UCIe Consortium policies and procedures; and (iv) the UCIe Member's Participation Agreement.

NOTICE TO NON-MEMBERS OF UNIVERSAL CHIPLET INTERCONNECT EXPRESS, INC.:

If you are **not** a UCIe Member and have received this publicly available version of this UCIe Specification, your use of this document is subject to your compliance with, and is limited by, all of the terms and conditions of the UCIe Consortium's Evaluation Copy Agreement (a copy of which is available at www.uciexpress.org/specifications).

In addition to the restrictions set forth in the UCIe Consortium's Evaluation Copy Agreement, any references or citations to this document must acknowledge Universal Chiplet Interconnect Express, Inc.'s sole and exclusive copyright ownership of this UCIe Specification. The proper copyright citation or reference is as follows: **"© 2022–2024 UNIVERSAL CHIPLET INTERCONNECT EXPRESS, INC. ALL RIGHTS RESERVED."** When making any such citation or reference to this document you are not permitted to revise, alter, modify, make any derivatives of, or otherwise amend the referenced portion of this document in any way without the prior express written permission of Universal Chiplet Interconnect Express, Inc.

Except for the limited rights explicitly given to a non-UCIe Member pursuant to the explicit provisions of the UCIe Consortium's Evaluation Copy Agreement which governs the publicly available version of this UCIe Specification, nothing contained in this UCIe Specification shall be deemed as granting (either expressly or impliedly) to any party that is not a UCIe Member: (i) any kind of license to implement or use this UCIe Specification or any portion or content described or contained therein, or any kind of license in or to any other intellectual property owned or controlled by the UCIe Consortium, including without limitation any trademarks of the UCIe Consortium; or (ii) any benefits and/or rights as a UCIe Member under any UCIe Governing Documents. For clarity, and without limiting the foregoing notice in any way, if you are **not** a UCIe Member but still elect to implement this UCIe Specification or any portion described herein, you are hereby given notice that your election to do so does not give you any of the rights, benefits, and/or protections of the UCIe Members, including without limitation any of the rights, benefits, privileges or protections given to a UCIe Member under the UCIe Consortium's Intellectual Property Policy.

LEGAL DISCLAIMERS AND ADDITIONAL NOTICE TO ALL PARTIES:

THIS DOCUMENT AND ALL SPECIFICATIONS AND/OR OTHER CONTENT PROVIDED HEREIN IS PROVIDED ON AN "**AS IS**" BASIS. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, UNIVERSAL CHIPLET INTERCONNECT EXPRESS, INC. (ALONG WITH THE CONTRIBUTORS TO THIS DOCUMENT) HEREBY DISCLAIM ALL REPRESENTATIONS, WARRANTIES AND/OR COVENANTS, EITHER EXPRESS OR IMPLIED, STATUTORY OR AT COMMON LAW, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, VALIDITY, AND/OR NON-INFRINGEMENT.

In the event this UCIe Specification makes any references (including without limitation any incorporation by reference) to another standard's setting organization's or any other party's ("**Third Party**") content or work, including without limitation any specifications or standards of any such Third Party ("**Third Party Specification**"), you are hereby notified that your use or implementation of any Third Party Specification: (i) is not governed by any of the UCIe Governing Documents; (ii) may require your use of a Third Party's patents, copyrights or other intellectual property rights, which in turn may require you to independently obtain a license or other consent from that Third Party in order to have full rights to implement or use that Third Party Specification; and/or (iii) may be governed by the intellectual property policy or other policies or procedures of the Third Party which owns the Third Party Specification. Any trademarks or service marks of any Third Party which may be referenced in this UCIe Specification are owned by the respective owner of such marks.

The **UCIe**™ and **UNIVERSAL CHIPLET INTERCONNECT EXPRESS**™ trademarks and logos (the "**UCIe Trademarks**") are trademarks owned by the UCIe Consortium. The UCIe Consortium reserves all rights in and to all of its UCIe Trademarks.

NOTICE TO ALL PARTIES REGARDING THE PCI-SIG UNIQUE VALUE PROVIDED IN THIS SPECIFICATION:
NOTICE TO USERS: THE UNIQUE VALUE THAT IS PROVIDED IN THIS SPECIFICATION FOR USE IN VENDOR DEFINED MESSAGE FIELDS, DESIGNATED VENDOR SPECIFIC EXTENDED CAPABILITIES, AND ALTERNATE PROTOCOL NEGOTIATION ONLY AND MAY NOT BE USED IN ANY OTHER MANNER, AND A USER OF THE UNIQUE VALUE MAY NOT USE THE UNIQUE VALUE IN A MANNER THAT (A) ALTERS, MODIFIES, HARMS OR DAMAGES THE TECHNICAL FUNCTIONING, SAFETY OR SECURITY OF THE PCI-SIG ECOSYSTEM OR ANY PORTION THEREOF, OR (B) COULD OR WOULD REASONABLY BE DETERMINED TO ALTER, MODIFY, HARM OR DAMAGE THE TECHNICAL FUNCTIONING, SAFETY OR SECURITY OF THE PCI-SIG ECOSYSTEM OR ANY PORTION THEREOF (FOR PURPOSES OF THIS NOTICE, "**PCI-SIG ECOSYSTEM**" MEANS THE PCI-SIG SPECIFICATIONS, MEMBERS OF PCI-SIG AND THEIR ASSOCIATED PRODUCTS AND SERVICES THAT INCORPORATE ALL OR A PORTION OF A PCI-SIG SPECIFICATION AND EXTENDS TO THOSE PRODUCTS AND SERVICES INTERFACING WITH PCI-SIG MEMBER PRODUCTS AND SERVICES).

Contents

Figures

Tables

Terminology

Table 1. Terms and Definitions (Sheet 1 of 8)

Term	Definition
Ack	Acknowledge
ACPI	Advanced Configuration and Power Interface
Addr	Address
Advanced Package	This packaging technology is used for performance optimized applications and short reach interconnects.
AFE	Analog Front End
ALMP	ARB/MUX Link Management Packet (as defined in *CXL Specification*)
APMW	Advanced Package Module Width
ARB/MUX	Arbiter/Multiplexer (as defined in *CXL Specification*)
Asset	Any data or mechanism used to access data that should be protected from illicit access, use, availability, disclosure, alteration, destruction, or theft.
ATE	Automated Test Equipment
B2B	Back-to-Back
BAR	Base Address Register
BDF	Bus Device Function
BE	Byte Enable
BEI	BAR Equivalent Indicator
BER	Bit Error Ratio
BFM	Bus Functional Model
bubble	Gap in data transfer and/or signal transitions. Measured in number of clock cycles.
bundle	Tx group or Rx group for UCIe-3D interconnects that contains data, clock, power, and ground. A 3D Module consists of a Tx bundle and an Rx bundle.
C4 bump	Controller Collapse Chip Connect bump
CA	Completer Abort
CDM	Charged Device Model
chiplet	Integrated circuit die that contains a well-defined subset of functionality that is designed to be combined with other chiplets in a package.
clear cleared	If clear or reset is used and no value is provided for a bit, it is interpreted as 0b.
CLM	Current Lane Map
CMLS	Common Maximum Link Speed
CoWoS	Chip on Wafer on Substrate
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CXL	Compute eXpress Link
CXL 68B Flit Mode	This term is used to reference 68B Flit Mode related Protocol features defined in *CXL Specification*.
CXL 256B Flit Mode	This term is used to reference 256B Flit Mode related Protocol features defined in *CXL Specification*.
D2C	Data-to-Clock
D2D	Die-to-Die
DCC	Duty Cycle Correction

Table 1. Terms and Definitions (Sheet 2 of 8)

Term	Definition
DDR	Double Data Rate Memory
DevID	Device ID
DFx	Design for Debug or Design for Test
DLLP	Data Link Layer Packet (as defined in *PCIe Base Specification*)
DLP	In Flit modes, the Data Link Layer Payload within a flit (as defined in *PCIe Base Specification*).
DMH	DFx Management Hub. DFx entity that provides enumeration/global control/status of DFx capabilities in a chiplet.
DMS	DFx Management Spoke. DFx entity that implements a specific test/debug functionality within a DMH.
DMS-ID	Static design time ID assigned to a DMS for ID-routed messages within a DMH. Interchangeably used with the term Spoke-ID.
Domain Reset (domain reset)	Used to refer to a hardware mechanism that sets or returns all UCIe registers and state machines associated with a given UCIe Link to their initialization values as specified in this document. It is required for both sides of the Link to have an overlapping time window such that they are both in domain reset concurrently.
DP	Downstream Port
DSP	Downstream Switch Port (as defined in *CXL Specification*)
DVFS	Dynamic Voltage Frequency Scaling
DVSEC	Designated Vendor-Specific Extended Capability (as defined in *PCIe Base Specification*)
DWORD	Double Word. Four bytes. When used as an addressable quantity, a Double Word is four bytes of data that are aligned on a four-byte boundary (i.e., the least significant two bits of the address are 00b).
E2E	End to end
EM	Eye Margin
EMIB	Embedded Multi-die Interconnect Bridge
EML	Eye Margin Left
EMR	Eye Margin Right
EMV	Eye Margin Valid
Encapsulated MTP eMTP	Encapsulated Management Transport Packet. The resulting packet after Encapsulation.
Encapsulation	Process of splitting an MTP or Vendor defined messages (exchanged between Management Port Gateways on both ends of a link) into smaller pieces to meet any required payload length restrictions or for any other reasons like credit availability, adding a 2-DWORD header to each piece and if required, adding a 1-DWORD data padding at the end of an MTP to transmit the MTP over sideband or mainband UCIe link. In the case of an MTP, the resulting packet after Encapsulation is called the Encapsulated MTP.
Endpoint EP	As defined in *PCIe Base Specification*.
eRCD	Exclusive Restricted CXL Device (as defined in *CXL Specification*)
eRCH	Exclusive Restricted CXL Host (as defined in *CXL Specification*)
ESD	Electro-Static Discharge
F2B	Face-to-Back
F2F	Face-to-Face
FDI	Flit-Aware Die-to-Die Interface
FEC	Forward Error Correction
FEXT	Far-End CrossTalk
FH	Flit Header

Table 1. Terms and Definitions (Sheet 3 of 8)

Term	Definition
FIFO	First In, First Out
FIR	Finite Impulse Response
FIT	Failure In Time. 1 FIT = 1 device failure in 10^9 hours.
Flit	Link Layer unit of transfer (as defined in *CXL Specification*).
Flit_Marker FM	Flit Marker (as defined in *PCIe Base Specification*)
FW	Firmware
FW-CLK	Forwarded Clock over the UCIe Link for mainband data Lanes
HMLS	Highest Maximum Link Speed of next-lower configuration.
Hub	See DMH.
HW	Hardware
IL	Insertion Loss
I/O	Input/Output
IP	Generic term used to refer to architecture blocks that are defined within the specification (e.g., D2D adapter, PHY, etc.).
IPA	Ignore Prohibited Access
ISI	Inter-Symbol Interference
KPI	Key Performance Indicator
Lane	A pair of signals mapped to physical bumps, one for Transmission, and one for Reception. A xN UCIe Link is composed of N Lanes.
LCLK	Refers to the clock at which the Logical Physical Layer, Adapter and RDI/FDI are operating.
LCRC	Link CRC
LFSR	Linear Feedback Shift Register
Link UCIe Link	A Link or UCIe Link refers to the set of two UCIe components and their interconnecting Lanes which forms a dual-simplex communications path between the two components.
LSM	Adapter Link State Machine
LTSM	Link Training State Machine
LTSSM	Link Training and Status State Machine (as defined in *PCIe Base Specification*)
LSB	Least Significant Bit
Mainband MB	Connection that constitutes the main data path of UCIe. Consists of a forwarded clock, a data valid pin, and N Lanes of data per module.
Management Bridge	Type of Management Entity that bridges a Management Network within an SiP to another network that may be internal or external to the SiP.
Management Director	Management Element that is responsible for discovering, configuring, and coordinating the overall management of the SiP and acts as the manageability Root of Trust (RoT).
Management Domain	One or more chiplets in an SiP that are interconnected by a Management Network and support UCIe Manageability.
Management Element	Type of Management Entity that can perform one or more management functions.
Management Entity	Addressable entity on the Management Network that can send and/or receive UCIe Management Transport packets. A Management Element, a Management Port, and a Management Bridge are all a type of Management Entity.
Management Flit	A Flit that carries a Management Port Message (MPM).
Management Link Encapsulation Mechanism	Mechanism that defines how UCIe Management Transport packets are transferred across a point-to-point management link.

Table 1. Terms and Definitions (Sheet 4 of 8)

Term	Definition
Management Network	Network within and between chiplets that is capable of transporting UCIe Management Transport packets.
Management Port	Management Entity that facilitates management communication between chiplets using a chiplet-to-chiplet management link.
Management Port Gateway (MPG)	Entity that provides the bridging functionality when transporting an MTP from/to a local SoC management fabric (which is an SoC-specific implementation) to/from a UCIe link.
Management Port Message (MPM)	Sideband or mainband message that relates to encapsulation.
Management Protocol	Protocol carried on top of the UCIe Management Transport.
Management Reset	Type of reset that causes all UCIe manageability and manageability structures in a chiplet to be reset to their default state.
MCLS	Number of active Modules Current Link Speed
MMIO	Memory mapped Input/Output
MMPL	Multi-module PHY Logic
Module	UCIe main data path on the physical bumps is organized as a group of Lanes called a Module. For Standard Package, 16 Lanes constitute a single Module. For Advanced Package, 64 Lanes constitute a single Module.
MSB	Most Significant Bit
MTP	Management Transport Packet
Nak	Negatively acknowledge
NEXT	Near-End CrossTalk
NOP	No Operation
NVMe	Non-Volatile Memory express
One-Time Programmable	Any data storage mechanism that is capable of being programmed only once (e.g., fuse).
P2P	Peer to peer
Packet	A block of data transmitted across a network.
PCIe (PCI Express)	Peripheral Component Interconnect Express (defined in *PCIe Base Specification*)
PCIe Flit Mode	This term is used to reference Flit Mode related Protocol features defined in *PCIe Base Specification*.
PCIe non-Flit Mode	This term is used to reference non-Flit Mode related Protocol features defined in *PCIe Base Specification*.
PDOS	Permanent Denial of Service
PDS	Pause of Data Stream
PHY	Physical Layer (PHY and Physical Layer are used interchangeable throughout the Specification)
PI	Phase Interpolator
PLL	Phase-Locked Loop
PM	Power Management states, used to refer to behavior and/or rules related to Power Management states (covers both L1 and L2).
PMO	Sideband Performant Mode Operation
QWORD	Quad Word. Eight bytes. When used as an addressable quantity, a Quad Word is eight bytes of data that are aligned on an eight-byte boundary (i.e., the least significant three bits of the address are 000b).
RAC	Read Access Control
RCD	Restricted CXL Device (as defined in *CXL Specification*)
RCH	Restricted CXL Host (as defined in *CXL Specification*)
RCiEP	Root Complex Integrated Endpoint

Table 1. Terms and Definitions (Sheet 5 of 8)

Term	Definition
RCKN_P RXCKN rxckn	Physical Lane for Clock Receiver Phase-2
RCKP_P RXCKP rxckp	Physical Lane for Clock Receiver Phase-1
RCRB	Root Complex Register Block
RDI	Raw Die-to-Die Interface
RD_P[N] RD_PN RXDATA[N] rxdataN	Nth Physical Lane for Data Receiver
remote Link partner	This term is used throughout this specification to denote the logic associated with the far side of the UCIe Link; to denote actions or messages sent or received by the Link partner of a UCIe die.
Replay Retry	Retry and Replay are used interchangeably to refer to the Link level reliability mechanisms.
Reserved	The contents, states, or information are not defined at this time. Using any Reserved area (for example, packet header bit-fields, configuration register bits) is not permitted. Reserved register fields must be read only and must return 0 (all 0s for multi-bit fields) when read. For packets transmitted and received over the UCIe Link (mainband or sideband), the Reserved bits must be cleared to 0b by the sender and ignored by the receiver. Reserved encodings for register and packet fields must not be used. Any implementation dependence on a Reserved field value or encoding will result in an implementation that is not UCIe-compliant. The functionality of such an implementation cannot be guaranteed in this or any future revision of this specification. For registers, UCIe uses the "RsvdP" or "RsvdZ" attributes for reserved fields, as well as Rsvd, and these follow the same definition as PCIe Base Specification for hardware as well as software.
reset	If reset or clear is used and no value is provided for a bit, it is interpreted as 0b.
RID	Revision ID
RL	Register Locator
Root Complex	As defined in *PCIe Base Specification*.
Root Port RP	As defined in *PCIe Base Specification*.
RoT	Root of Trust
RRDCK_P RXCKRD rxckRD	Physical Lane for redundant Clock/Track Receiver
RRD_P[N] RRD_PN RXDATARD[N] rxdataRD[N]	Nth Physical Lane for redundant Data Receiver
RRDVLD_P RXVLDRD rxvldRD	Physical Lane for redundant Valid Receiver
RTRK_P RXTRK rxtrk	Physical Lane for Track Receiver
RVLD_P RXVLD rxvld	Physical Lane for Valid Receiver
Rx	Receiver

Table 1. Terms and Definitions (Sheet 6 of 8)

Term	Definition
RXCKSB rxcksb	Physical Lane for sideband Clock Receiver
RXCKSBRD rxcksbRD	Physical Lane for redundant sideband Clock Receiver
RXDATASB rxdatasb	Physical Lane for sideband Data Receiver
RXDATASBRD rxdatasbRD	Physical Lane for redundant sideband Data Receiver
SBFE	Sideband Feature Extensions
{<SBMSG>}	Sideband message requests or responses are referred to by their names enclosed in curly brackets. See Chapter 7.0 for the mapping of sideband message names to relevant encodings. An asterisk in the <SBMSG> name is used to denote a group of messages with the same prefix or suffix in their name.
SC	Successful Completion
SD	Security Director. Management Element that may configure security parameters.
Segmentation	Process of taking a large MTP, splitting it into smaller "segments" and sending those segments on multiple sideband links or mainband stacks.
SERDES	Serializer/Deserializer
serial packet	A 64-bit serial packet is defined on the sideband I/O interface to the remote chiplet as shown in Figure 4-8.
set	If set is used and no value is provided for a bit, it is interpreted as 1b.
SFES	Sideband Feature Extensions Supported
Sideband SB	Connection used for parameter exchanges, register accesses for debug/compliance and coordination with remote partner for Link training and management. Consists of a forwarded clock pin and a data pin in each direction. The clock is fixed at 800 MHz regardless of the main data path speed. The sideband logic for the UCIe Physical Layer must be on auxiliary power and an "always on" domain. Each module has its own set of sideband pins.
SiP	System in Package. Collection of chiplets packaged as a unit.
SM	State Machine
SO	Sideband-only
SoC	System on a Chip
Spoke	See DMS.
Standard Package	This packaging technology is used for low cost and long reach interconnects using traces on organic package/substrate
Strobe	Used interchangeably with clock for sideband clock
SW	Software
TC	Traffic Class
TCKN_P TXCKN txckn	Physical Lane for Clock Transmitter Phase-2
TCKP_P TXCKP txckp	Physical Lane for Clock Transmitter Phase-1
TCM	Tightly coupled mode
TDPI	Test, Debug, Pattern, and Infrastructure

Table 1. Terms and Definitions (Sheet 7 of 8)

Term	Definition
TD_P[N] TD_PN TXDATA[N] txdataN	Nth Physical Lane for Data Transmitter
TLP	Transaction Layer Packet (as defined in *PCIe Base Specification*)
TRD_P[N] TRD_PN TXDATARD[N] txdataRD[N]	Nth Physical Lane for redundant Data Transmitter
TRDCK_P TXCKRD txckRD	Physical Lane for redundant Clock/Track Transmitter
TRDVLD_P TXVLDRD txvldRD	Physical Lane for redundant Valid Transmitter
Trx	Transceiver
TSV	Through-Silicon Via
TTRK_P TXTRK txtrk	Physical Lane for Track Transmitter
TVLD_P TXVLD txvld	Physical Lane for Valid Transmitter
Tx	Transmitter
TXCKSB txcksb	Physical Lane for sideband Clock Transmitter
TXCKSBRD txcksbRD	Physical Lane for redundant sideband Clock Transmitter
TXDATASB txdatasb	Physical Lane for sideband Data Transmitter
TXDATASBRD txdatasbRD	Physical Lane for redundant sideband Data Transmitter
TXEQ	Transmitter Equalization
UCIe	Universal Chiplet Interconnect express
UCIe-3D	Universal Chiplet Interconnect express for 3D packaging
UCIe-A	Used to denote x64 Advanced Package module.
UCIe-A x32	Used to denote x32 Advanced Package module. See Chapter 5.0 for UCIe-A x32 Advanced Package bump matrices, and interoperability between x32 to x32 and x32 to x64 module configurations.
UCIe-S	Used to denote x16 Standard Package module.
UCIe chiplet	A chiplet that complies with the UCIe specification.
UCIe DFx Architecture UDA	DFx architecture specified for chiplets and SiPs that implement UCIe.
UCIe DFx Message UDM	Generic term for all UCIe Management Transport packets with Protocol ID set to 'Test and Debug Protocols'.
UCIe die	This term is used throughout this specification to denote the logic associated with the UCIe Link on any given chiplet with a UCIe Link connection. It is used as a common noun to denote actions or messages sent or received by an implementation of UCIe.

Table 1. Terms and Definitions (Sheet 8 of 8)

Term	Definition
UCIe Flit Mode	Operating Mode in which CRC bytes are inserted and checked by the D2D Adapter. If applicable, Retry is also performed by the D2D Adapter.
UCIe Link	A UCIe connection between two chiplets. These chiplets are Link partners in the context of UCIe since they communicate with each other using a common UCIe Link.
UCIe Mainband Management Port	Chiplet port that implements the Management Link Encapsulation Mechanism and can transfer UCIe Management Transport packets across a point-to-point UCIe mainband link.
UCIe Management Transport Protocol	Protocol used to transfer UCIe Management Transport packets between management entities.
UCIe Raw Format	Operating format in which all the bytes of a Flit are populated by the Protocol Layer.
UCIe Sideband Management Port	Chiplet port that implements the Management Link Encapsulation Mechanism and can transfer UCIe Management Transport packets across a point-to-point UCIe sideband link.
UCLS	UCIe Link Structure
UEDT	UCIe Early Discovery Table
UHM	UCIe Link Health Monitor
UIE	Uncorrectable Internal Error
UiRB	UCIe Register Block
UiSRB	UCIe Structure Register Block
UMAP	UCIe Memory Access Protocol
Unit Interval UI	Given a data stream of a repeating pattern of alternating 1 and 0 values, the Unit Interval is the value measured by averaging the time interval between voltage transitions, over a time interval sufficiently long to make all intentional frequency modulation of the source clock negligible.
UP	Upstream Port
UR	Unsupported Request
USP	Upstream Switch Port
VH	Virtual Hierarchy (as defined in *CXL Specification*)
vLSM	Virtual Link State Machine
Vref	Reference voltage for receivers
VTF	Voltage Transfer Function
WAC	Write Access Control
zero	Numerical value of 0 in a bit, field, or register, of appropriate width for that bit, field, or register.

Table 2. Unit of Measure Symbols

Symbol	Unit of Measure
b	bit
B	byte
dB	decibel
fF	femtofarad
GB/s	gigabytes per second
GHz	gigahertz
GT/s	gigatransfers per second
KB/s	kilobytes per second
MB/s	megabytes per second
MHz	megahertz
mm	millimeter
ms	millisecond
MT/s	megatransfers per second
mUI	milli-Unit interval
mV	millivolt
mVpp	millivolt peak-to-peak
um	micrometer
us	microsecond
ns	nanosecond
pJ	picojoule
pk	peak
ppm	parts per million
ps	picosecond
s	second
TB/s	terabytes per second
V	volt

Reference Documents

Table 3. Reference Documents[a]

Document	Document Location
PCI Express® Base Specification (PCIe Base Specification) Revision 6.2	www.pcisig.com
Compute Express Link Specification (CXL Specification) Revision 3.1	computeexpresslink.org
ACPI Specification (version 6.5 or later)	www.uefi.org
Industry Council on ESD Targets white papers	www.jedec.org, reference JEP196 www.esdindustrycouncil.org/ic/en, reference White Paper 2 Part II www.esda.org, reference White Paper 2 Part II

a. References to these documents throughout this specification relate to the versions/revisions listed here.

Revision History

Table 4 lists the significant changes in different revisions.

Table 4. Revision History

Revision	Date	Description
2.0	August 6, 2024	• Chapter 6.0, UCIe-3D and related support for UCIe 3D • Chapter 8.0, System Architecture and related support for: — Section 8.1, "UCIe Manageability" — Section 8.2, "Management Transport Packet (MTP) Encapsulation" — Section 8.3, "UCIe Debug and Test Architecture (UDA)" • di/dt risk mitigation during clock gating • Incorporation of Errata and bug fixes over 1.1
1.1	July 10, 2023	• Streaming Flit Format Capabilities (Allows Streaming protocols to use Adapter Retry/CRC) • Enhanced Multi-Protocol Multiplexing (Allows dynamic multiplexing of different protocols on the same Adapter) • Support for x32 Advanced Package Modules • Support for UCIe Link Health Monitoring • Definition of Hardware capabilities to enable Compliance • di/dt risk mitigation during clock gating • Incorporation of Errata and bug fixes over 1.0
1.0	February 17, 2022	Initial release.

§ §

1.0 Introduction

This chapter provides an overview of the Universal Chiplet Interconnect express (UCIe) architecture. UCIe is an open, multi-protocol capable, on-package interconnect standard for connecting multiple dies on the same package. The primary motivation is to enable a vibrant ecosystem supporting disaggregated die architectures which can be interconnected using UCIe. UCIe supports multiple protocols (PCIe, CXL, Streaming, and a raw format that can be used to map any protocol of choice as long as both ends support it) on top of a common physical and Link layer. It encompasses the elements needed for SoC construction such as the application layer, as well as the form-factors relevant to the package (e.g., bump location, power delivery, thermal solution, etc.).

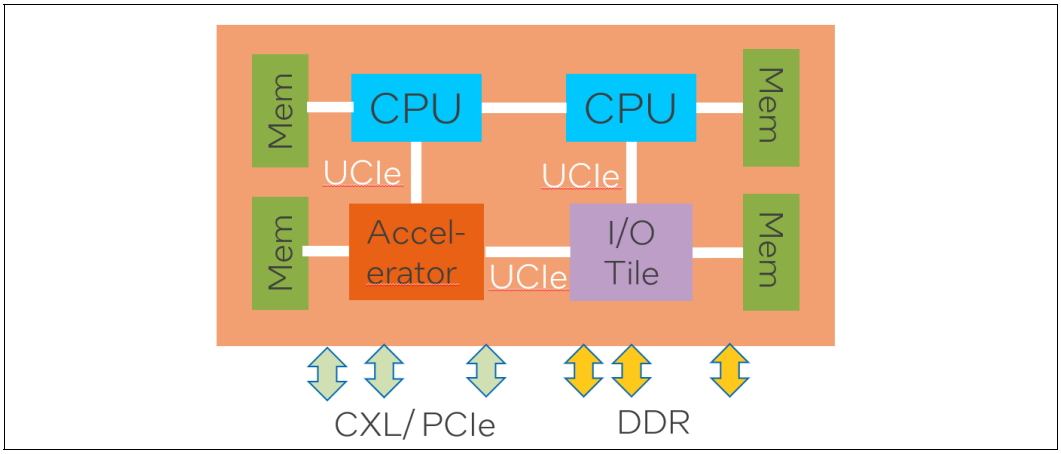
UCIe Manageability Architecture is an optional mechanism to manage a UCIe-based System-in-Package (SiP) by provisioning for a common manageability architecture and hardware/software infrastructure to be leveraged across implementations. UCIe DFx Architecture (UDA) leverages the UCIe Manageability Architecture to provide a standardized test and debug infrastructure for a UCIe-based SiP.

The specification is defined to ensure interoperability across a wide range of devices having different performance characteristics. A well-defined debug and compliance mechanism is provided to ensure interoperability. It is expected that the specification will evolve in a backward compatible manner.

While UCIe supports a wide range of usage models, some are provided here as an illustration of the type of capability and innovation it can unleash in the compute industry. The initial protocols being mapped to UCIe are PCIe, CXL, and Streaming. The mappings for all protocols are done using a Flit Format, including the Raw Format. Both PCIe and CXL are widely used and these protocol mappings will enable more on-package integration by replacing the PCIe SERDES PHY and the PCIe/CXL LogPHY along with the Link level Retry with a UCIe Adapter and PHY to improve the power and performance characteristics. UCIe provisions for Streaming protocols to also leverage the Link Level Retry of the UCIe Adapter, and this can be used to provide reliable transport for protocols other than PCIe or CXL. UCIe also supports a Raw Format that is protocol-agnostic to enable other protocols to be mapped; while allowing usages such as integrating a standalone SERDES/transceiver tile (e.g., ethernet) on-package. When using Raw Format, the Protocol Layer is responsible for reliable transport across the UCIe Link.

Figure 1-1 demonstrates an SoC package composed of CPU Dies, accelerator Die(s) and I/O Tile Die(s) connected through UCIe. The accelerator or I/O Tile can use CXL transactions over UCIe when connected to a CPU — leveraging the I/O, coherency, and memory protocols of CXL. The I/O tile can provide the external CXL, PCIe, and DDR pins of the package. The accelerator can also use PCIe transactions over UCIe when connected to a CPU. The CPU to CPU connectivity on-package can also use the UCIe interconnect, running coherency protocols.

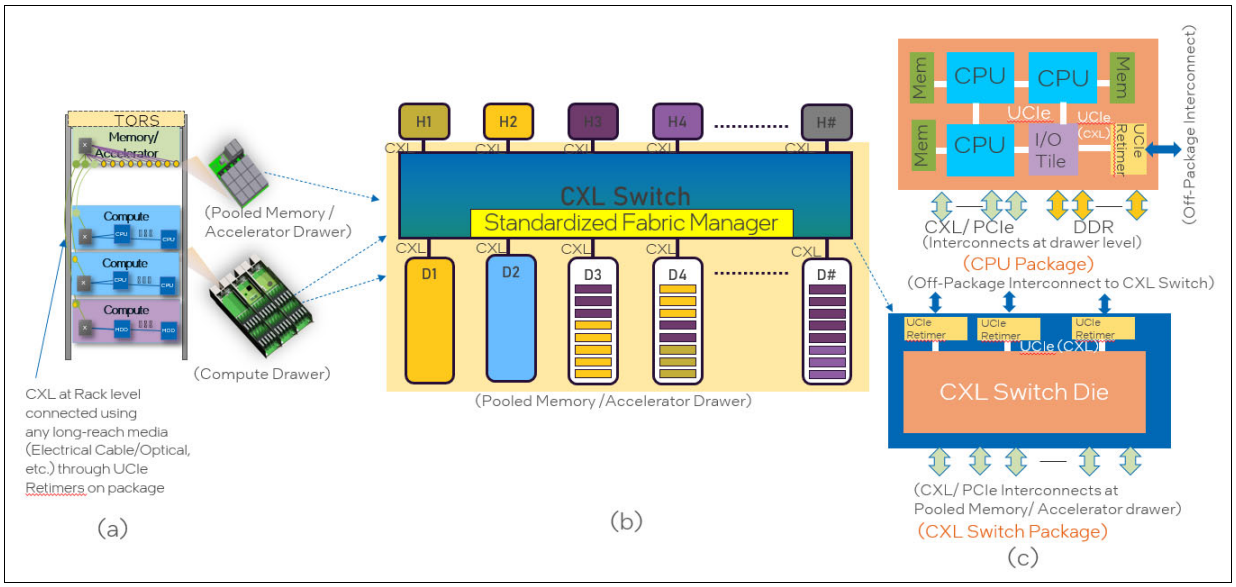
Figure 1-1. A Package Composed of CPU Dies, Accelerator Die(s), and I/O Tile Die Connected through UCIe



A UCIe Retimer may be used to extend the UCIe connectivity beyond the package using an Off-Package Interconnect. Examples of Off-Package Interconnect include electrical cable or optical cable or any other technology to connect packages at a Rack/Pod level as shown in Figure 1-2. The UCIe specification requires the UCIe Retimer to implement the UCIe interface to the Die that it connects on its local package and ensure that the Flits are delivered to the remote UCIe Die interface in the separate package following UCIe protocol using the channel extension technology of its choice.

Figure 1-2 demonstrates a rack/pod-level disaggregation using CXL protocol. Figure 1-2a shows the rack level view where multiple compute nodes (virtual hierarchy) from different compute chassis connect to a CXL switch which connects to multiple CXL accelerators/Type-3 memory devices which can be placed in one or more separate drawer. The logical view of this connectivity is shown in Figure 1-2b, where each "host" (H1, H2,…) is a compute drawer. Each compute drawer connects to the switch using an Off-Package Interconnect running CXL protocol through a UCIe Retimer, as shown in Figure 1-2c. The switch also has co-package Retimers where the Retimer tiles connect to the main switch die using UCIe and on the other side are the PCIe/CXL physical interconnects to connect to the accelerators/memory devices, as shown in Figure 1-2c.

Figure 1-2. UCIe enabling long-reach connectivity at Rack/Pod Level



UCIe permits three different packaging options: Standard Package (2D), and Advanced Package (2.5D), and UCIe-3D. This covers the spectrum from lowest cost to best performance interconnects.

1. **Standard Package** — This packaging technology is used for low cost and long reach (10 mm to 25 mm, when measured from a bump on one Die to the connecting bump of the remote Die) interconnects using traces on organic package/substrate, while still providing significantly better BER characteristics compared to off-package SERDES. Figure 1-3 shows an example application using the Standard Package option. Table 1-1 shows a summary of the characteristics of the Standard Package option with UCIe.

Figure 1-3. Standard Package interface

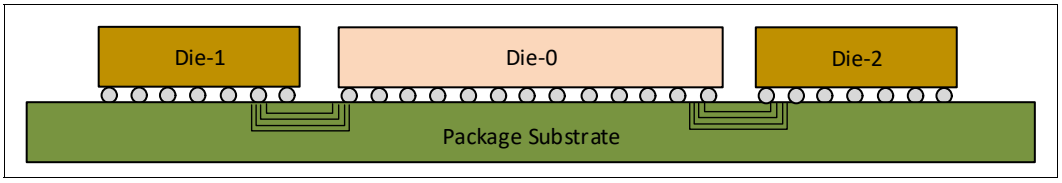


Table 1-1. Characteristics of UCIe on Standard Package

Index	Value
Supported speeds (per Lane)	4 GT/s, 8 GT/s, 12 GT/s, 16 GT/s, 24GT/s, 32 GT/s
Bump Pitch	100 um to 130 um
Channel reach (short reach)	10 mm
Channel reach (long reach)	25 mm
Raw Bit Error Rate (BER)[a]	1e-27 (<= 8 GT/s)
	1e-15 (>= 12 GT/s)

a. See Chapter 5.0 for details about BER characteristics.

2. **Advanced Package** — This packaging technology is used for performance optimized applications. Consequently, the channel reach is short (less than 2 mm, when measured from a bump on one Die to the connecting bump of the remote Die) and the interconnect is expected to be optimized for high bandwidth and low latency with best performance and power efficiency characteristics. Figure 1-4, Figure 1-5, and Figure 1-6 show example applications using the Advanced Package option.

   Table 1-2 shows a summary of the main characteristics of the Advanced Package option.

Table 1-2. Characteristics of UCIe on Advanced Package

Index	Value
Supported speeds (per Lane)	4 GT/s, 8 GT/s, 12 GT/s, 16 GT/s, 24 GT/s, 32 GT/s
Bump pitch	25 um to 55 um
Channel reach	2 mm
Raw Bit Error Rate (BER)[a]	1e-27 (<=12GT/s)
	1e-15 (>=16GT/s)

a. See Chapter 5.0 for details about BER characteristics.

Figure 1-4. Advanced Package interface: Example 1

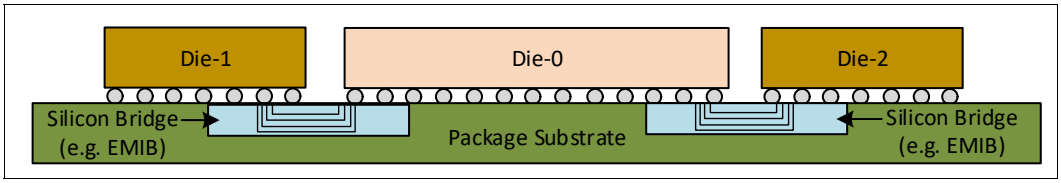


Figure 1-5. Advanced Package interface: Example 2

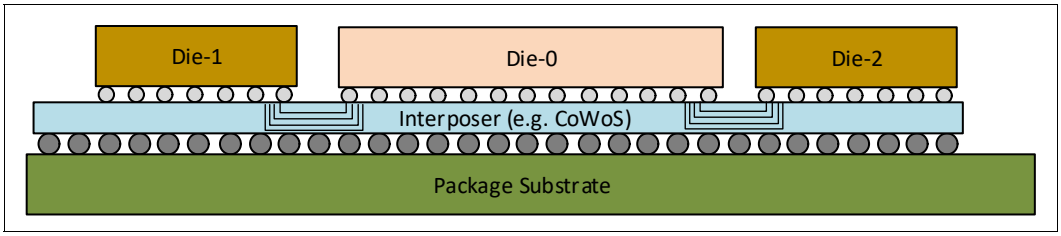
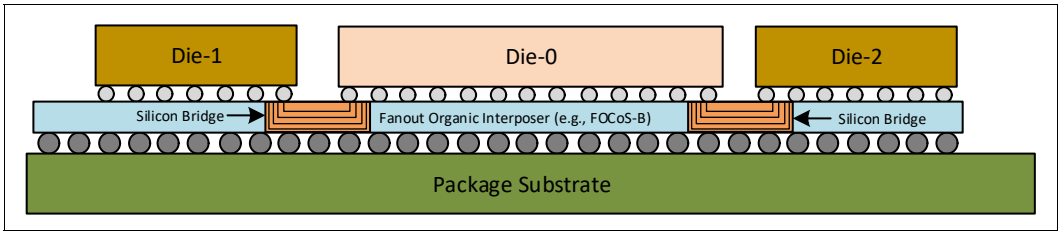


Figure 1-6. Advanced Package interface: Example 3



3. **UCIe-3D**: This packaging technology uses a two-dimensional array of interconnect bumps for data transmission between dies where one die is stacked on top of another. A menu of design options are provided for vendors to develop standard building blocks.

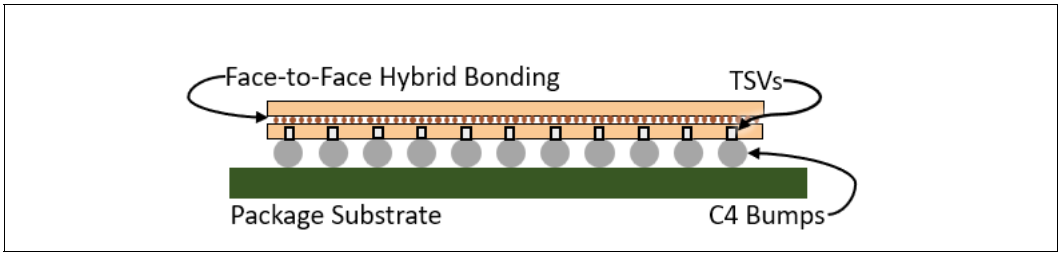
   Table 1-3 shows a summary of the main characteristics of UCIe-3D. Figure 1-7 shows an example of UCIe-3D. See Chapter 6.0 for a detailed description of UCIe-3D.

Table 1-3. Characteristics of UCIe-3D

Index	Value
Supported speed (per Lane)	up to 4 GT/s
Bump pitch	<10 um (optimized[a]) 10 to 25 um (functional[a])
Channel	3D vertical
Raw Bit Error Rate (BER)[b]	1e-27

a. Circuit Architecture is optimized for < 10 um bump pitches. 10 to 25 um are supported functionally.
b. See Chapter 6.0 for details about BER characteristics.

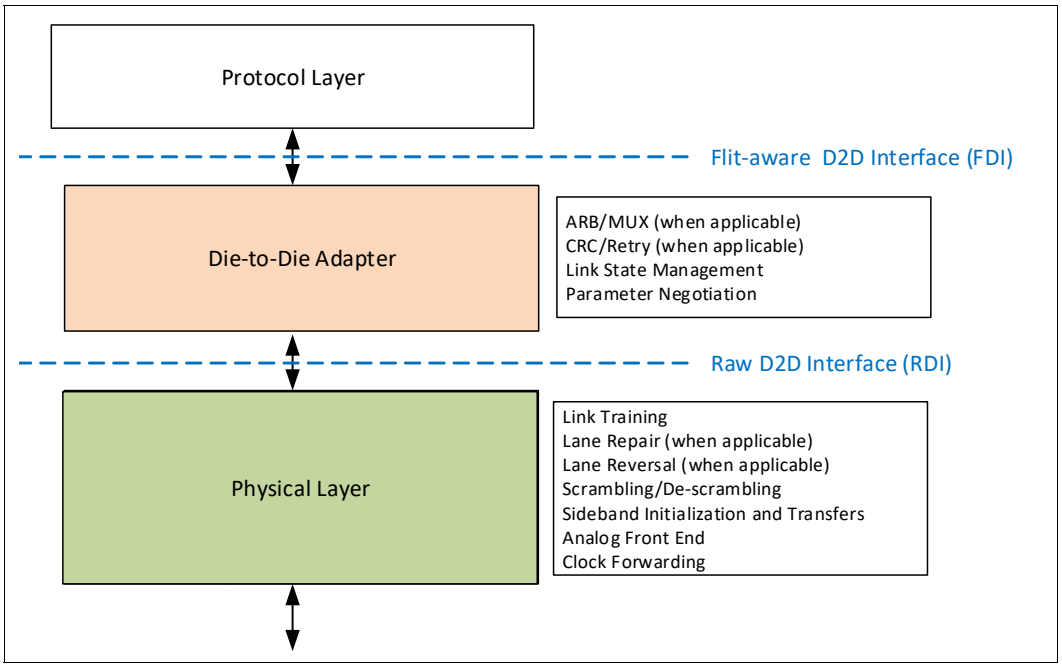
Figure 1-7. Example of UCIe-3D



1.1 UCIe Components

UCIe is a layered protocol, with each layer performing a distinct set of functions. Figure 1-8 shows the three main components of the UCIe stack and the functionality partitioning between the layers. It is required for every component in the UCIe stack to be capable of supporting the advertised functionality and bandwidth. Several timeouts and related errors are defined for different handshakes and state transitions. All timeout values specified are minus 0% and plus 50% unless explicitly stated otherwise. All timeout values must be set to the specified values after Domain Reset. All counter values must be set to the specified values after Domain Reset.

Figure 1-8. UCIe Layers and functionalities



1.1.1 Protocol Layer

While the Protocol Layer may be application specific, UCIe Specification provides examples of transferring CXL or PCIe protocols over UCIe Links. The following protocols are supported in UCIe for enabling different applications:

- PCIe from *PCIe Base Specification*.
- CXL from *CXL Specification*. Note that RCD/RCH/eRCD/eRCH are not supported.
- Streaming protocol: This offers generic modes for a user defined protocol to be transmitted using UCIe.
- UCIe Management Transport protocol[a]: This is an end-to-end media independent protocol(s) for management communication on the UCIe Management Network within the UCIe Manageability Architecture.

For each protocol, different optimizations and associated Flit transfers are available for transfer over UCIe. Chapter 2.0 and Chapter 3.0 cover the relevant details of different modes and Flit Formats.

a. UCIe Management Transport protocol can be encapsulated for transport over the UCIe sideband or the UCIe mainband. Section 8.1 covers the details of this protocol. Section 8.2 covers the details around encapsulation of this protocol over the UCIe sideband and the UCIe mainband.

1.1.2 Die-to-Die (D2D) Adapter

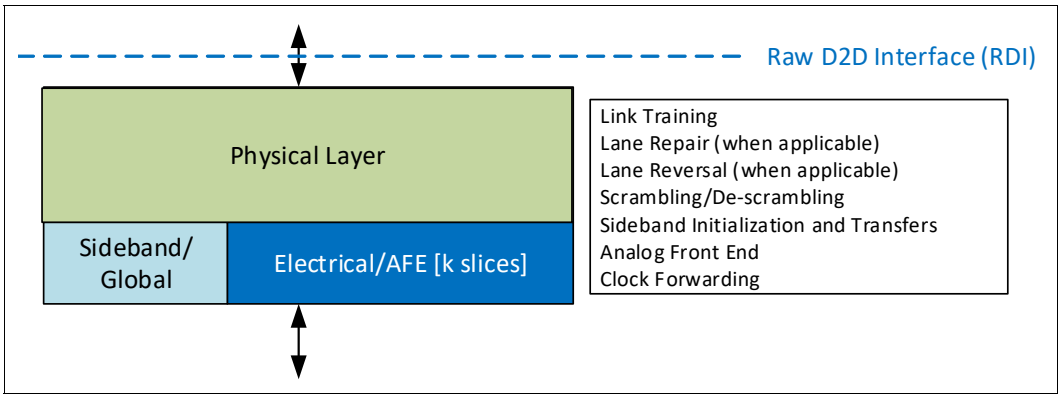
The D2D Adapter coordinates with the Protocol Layer and the Physical Layer to ensure successful data transfer across the UCIe Link. It minimizes logic on the main data path as much as possible, thus providing a low-latency, optimized data path for protocol Flits. When transporting CXL protocol, the ARB/MUX functionality required for multiple simultaneous protocols is performed by the D2D Adapter. For options where the Raw BER is more than 1e-27, a CRC and Retry scheme is provided in the UCIe Specification for PCIe, CXL, or Streaming protocol, which is implemented in the D2D Adapter. See Section 3.8 for Retry rules.

D2D Adapter is responsible for coordinating higher level Link state machine and bring up, protocol options related parameter exchanges with remote Link partner, and when supported, power management coordination with remote Link partner. Chapter 3.0 covers the relevant details for the D2D Adapter.

1.1.3 Physical Layer

The Physical Layer has three sub-components as shown in Figure 1-9.

Figure 1-9. Physical Layer components



The UCIe main data path on the physical bumps is organized as a group of Lanes called a Module. A Module forms the atomic granularity for the structural design implementation of the UCIe AFE. The number of Lanes per Module for Standard and Advanced Packages is specified in Chapter 4.0. A given instance of Protocol Layer or D2D adapter can send data over multiple modules where bandwidth scaling is required.

The physical Link of UCIe is composed of two types of connections:

1. **Sideband**:
   This connection is used for parameter exchanges, register accesses for debug/compliance and coordination with remote partner for Link training and management. It consists of a forwarded clock pin and a data pin in each direction. The clock is fixed at 800 MHz regardless of the mainband data rate. The sideband logic for the UCIe Physical Layer must be on auxiliary power and an "always on" domain. Each module has its own set of sideband pins.
   For the Advanced Package option, a redundant pair of clock and data pins in each direction is provided for repair.
2. **Mainband**:
   This connection constitutes the main data path of UCIe. It consists of a forwarded clock, a data valid pin, a track pin, and N Lanes of data per module.
   For the Advanced Package option, N=64 (also referred to as x64) or N=32 (also referred to as x32) and overall four extra pins for Lane repair are provided in the bump map.

For the Standard Package option, N=16 (also referred to as x16) or N=8 (also referred to as x8) and no extra pins for repair are provided.

The Logical Physical Layer coordinates the different functions and their relative sequencing for proper Link bring up and management (e.g., sideband transfers, mainband training and repair, etc.). Chapter 4.0 and Chapter 5.0 cover the details on Physical Layer operation.

1.1.4 Interfaces

UCIe defines the interfaces between the Physical Layer and the D2D Adapter (Raw D2D Interface), and the D2D Adapter and the Protocol Layer (Flit-aware D2D Interface) in Chapter 10.0. A reference list of signals is also provided to cover the interactions and rules of the Management Transport protocol between the SoC and the UCIe Stack.

The motivation for this is two-fold:

- Allow vendors and SoC builders to easily mix and match different layers from different IP providers at low integration cost and faster time to market. (For example, getting a Protocol Layer to work with the D2D Adapter and Physical Layer from any different vendor that conforms to the interface handshakes provided in the UCIe Specification.)
- Given that inter-op testing during post-silicon has greater overhead and cost associated with it, a consistent understanding and development of Bus Functional Models (BFMs) can allow easier IP development for this stack.

1.2 UCIe Configurations

This section describes the different configurations and permutations permitted for UCIe operation.

1.2.1 Single Module Configuration

A single Module configuration is a x64 or x32 data interface in an Advanced Package, as shown in Figure 1-10. A single module configuration is a x16 or a x8 data interface in a Standard Package, as shown in Figure 1-11. A x8 Standard Package module is only permitted for a single module configuration and is primarily provided for pre-bond test purposes. In multiple instantiations of a single module configuration where each module has its own dedicated Adapter, they operate independently (e.g., they could be transferring data at different data rates and widths).

Figure 1-10. Single module configuration: Advanced Package

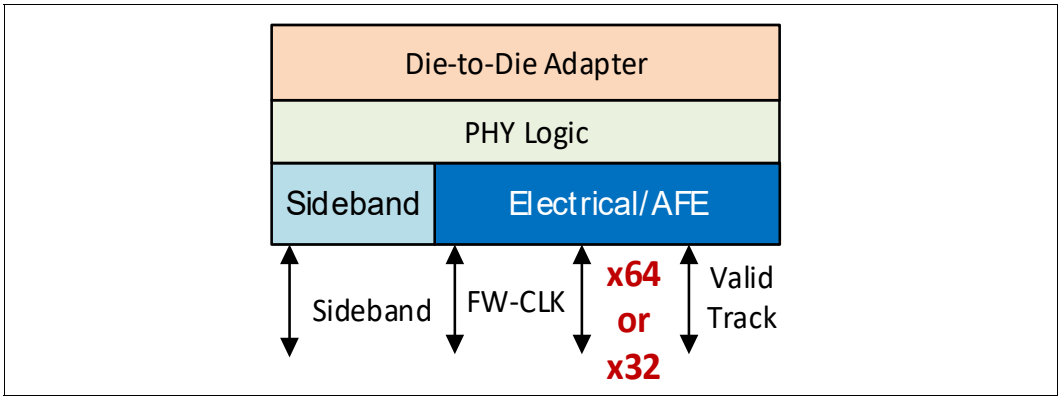
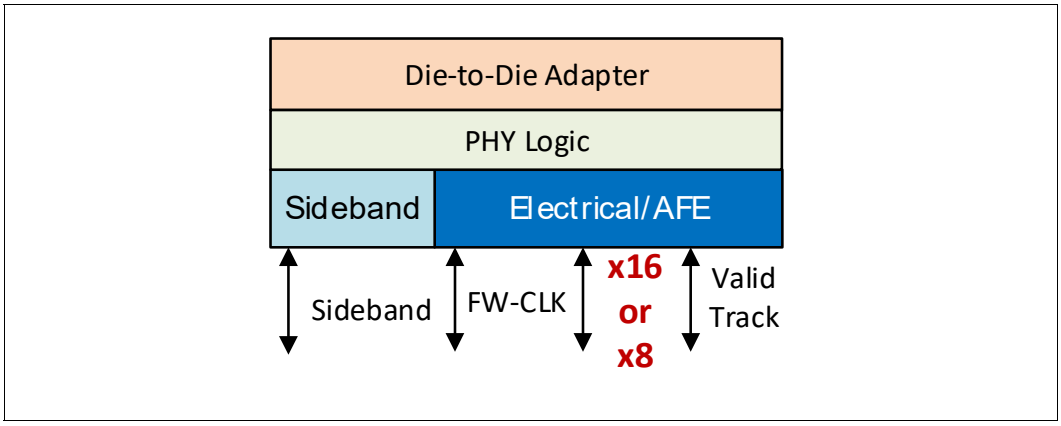


Figure 1-11. Single module configuration: Standard Package



1.2.2 Multi-module Configurations

This specification allows for two and four module configurations. When operating with a common Adapter, the modules in two-module and four-module configurations must operate at the same data rate and width. Examples of two-module and four-module configurations are shown in Figure 1-12 through Figure 1-14.

Figure 1-12. Two-module configuration for Standard Package

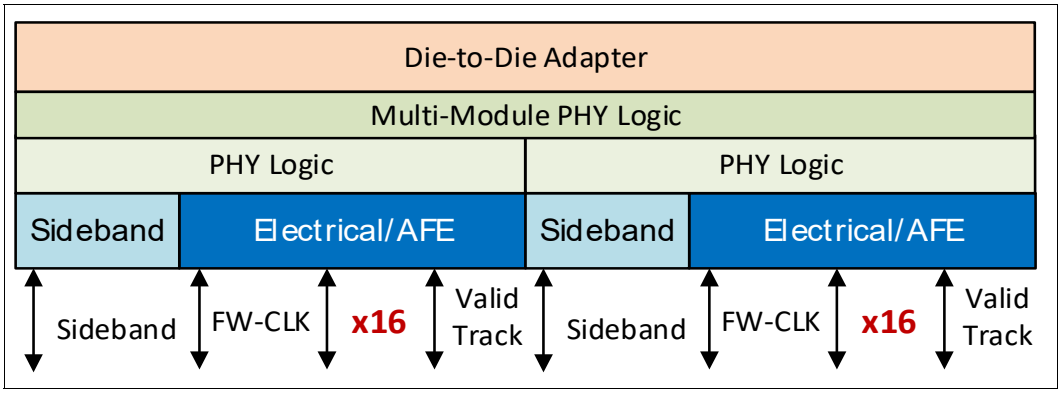


Figure 1-13. Four-module configuration for Standard Package

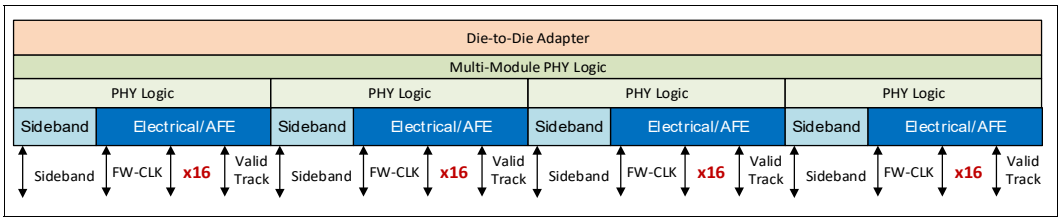
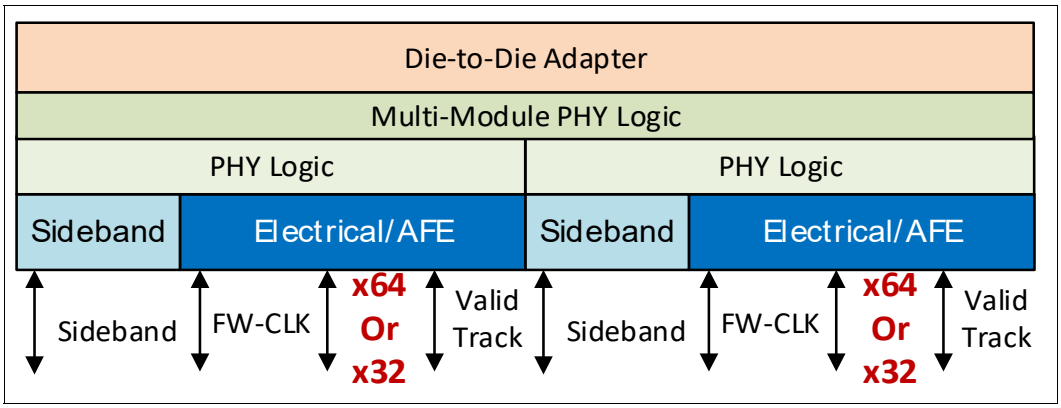


Figure 1-14. Example of a Two-module Configuration for Advanced Package



1.2.3 Sideband-only Configurations

A Standard Package UCIe sideband-only configuration is permitted for test or manageability purposes. This can be a one, two, or four sideband-only ports as part of the same UCIe sideband-only Link. Figure 1-15, Figure 1-16, and Figure 1-17 show examples of these configurations. See Section 5.7.4 for more details.

Figure 1-15. One-port Sideband-only Link

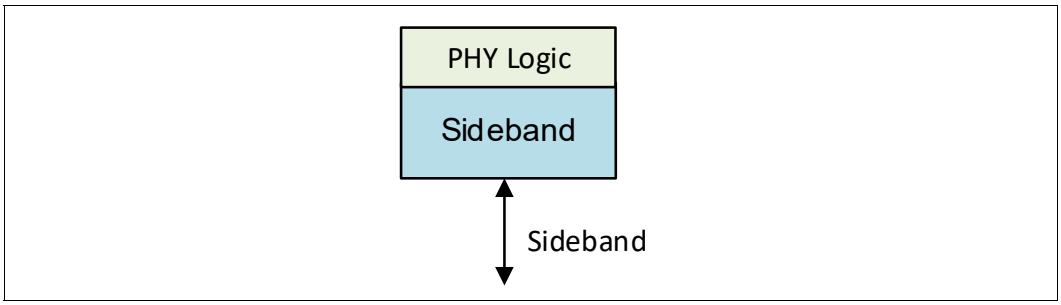


Figure 1-16. Two-port Sideband-only Link

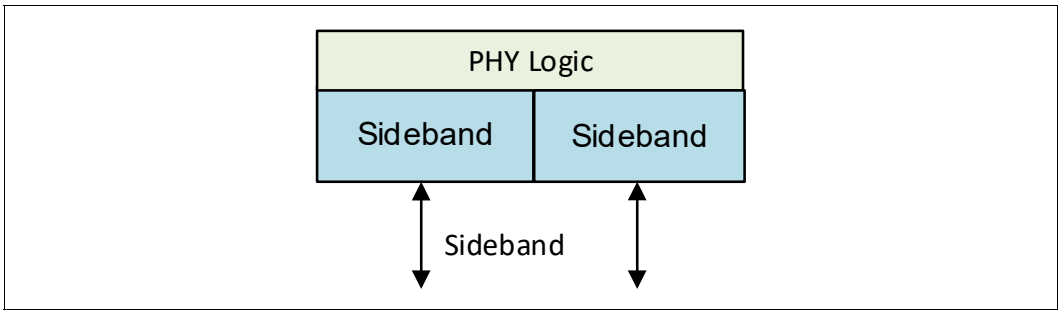
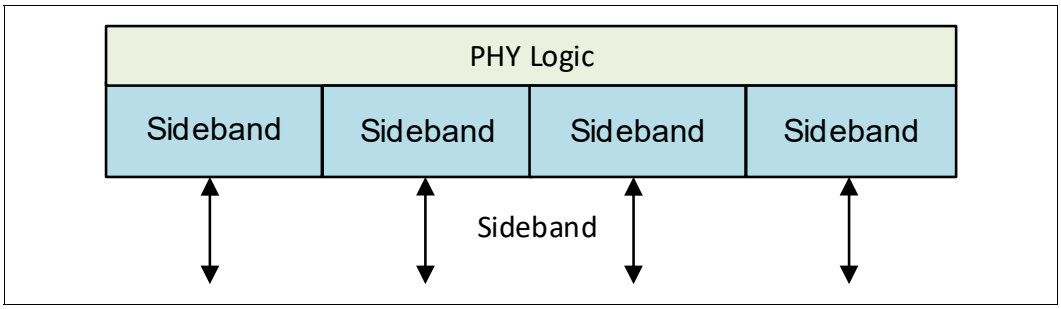


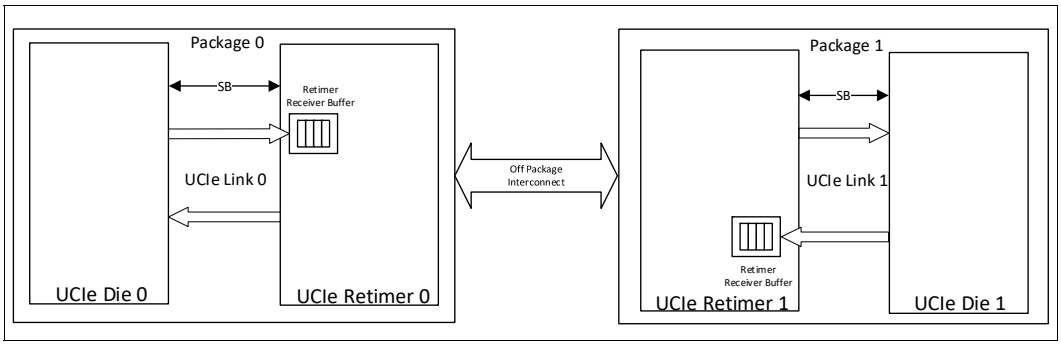
Figure 1-17. Four-port Sideband-only Link



1.3 UCIe Retimers

As described previously, UCIe Retimers are used to enable different types of Off Package Interconnects to extend the channel reach between two UCIe Dies on different packages. Each UCIe Retimer has a local UCIe Link connection to a UCIe die on-package as well as an external connection for longer reach. Figure 1-18 shows a high level block diagram demonstrating a system utilizing UCIe Retimers to enable an Off Package Interconnect between UCIe Die 0 and UCIe Die 1. UCIe Retimer 0 and UCIe Die 0 are connected through UCIe Link 0 within Package 0. UCIe Retimer 1 and UCIe Die 1 are connected through UCIe Link 1 within Package 1. The terminology of "remote Retimer partner" is used to reference the UCIe Retimer die connected to the far side of the Off Package Interconnect.

Figure 1-18. Block Diagram for UCIe Retimer Connection



The responsibility of a UCIe Retimer include:

- Reliable Flit transfer across the Off Package Interconnect. Three options are available for achieving this as described below:
  - The Retimer is permitted to use the FEC and CRC natively defined by the underlying specification of the protocol it carries (e.g., PCIe or CXL) as long as the external interconnect conforms to the underlying error model (e.g., BER and error correlation) of the specification corresponding to the protocol it transports. The UCIe Links would be setup to utilize the Raw Format to tunnel native bits of the protocol it transports (e.g., PCIe or CXL Flits). In this scenario, the queue sizes (Protocol Layer buffers) must be adjusted on the UCIe Dies to meet the underlying round trip latency.
  - The Retimer is permitted to provide the necessary FEC, CRC and Retry capabilities to handle the BER of the Off Package Interconnect. In this case, the Flits undergo three independent Links; each UCIe Retimer performs an independent ACK/NAK for Retry with the UCIe die within its package and a separate independent ACK/NAK for Retry with the remote Retimer

partner. For this scenario, protocols are permitted to use any of the applicable Flit Formats for transport over the UCIe Link.

  - The Retimer provides its own FEC by replacing the native PCIe or CXL defined FEC with its own, or adding its FEC in addition to the native PCIe or CXL defined FEC, but takes advantage of the built in CRC and Replay mechanisms of the underlying protocol. In this scenario, the queue sizes (Protocol Layer buffers, Retry buffers) must be adjusted on the UCIe Dies to meet the underlying round trip latency.

- Resolution of Link and Protocol Parameters with remote Retimer partner to ensure interoperability between UCIe Dies end-to-end (E2E). For example, Retimers are permitted to force the same Link width, speed, protocol (including any relevant protocol specific parameters) and Flit Formats on both Package 0 and Package 1 in Figure 1-18. The specific mechanism of resolution including message transfer for parameter exchanges across the Off Package Interconnect is implementation specific for the Retimers and they must ensure a consistent operational mode taking into account their own capabilities along with the UCIe Die capabilities on both Package 0 and Package 1. However, for robustness of the UCIe Links to avoid unnecessary timeouts in case the external interconnect requires a longer time to Link up or resolution of parameters with remote Retimer partner, UCIe Specification defines a "Stall" response to the relevant sideband messages that can potentially get delayed. The Retimers must respond with the "Stall" response within the rules of UCIe Specification to avoid such unnecessary timeouts while waiting for, or negotiating with remote Retimer partner. It is the responsibility of the Retimer to ensure the UCIe Link is not stalled indefinitely.
- Resolution of Link States for Adapter Link State Machine (LSM) or the RDI states with remote Retimer partner to ensure correct E2E operation. See Chapter 3.0 for more details.
- Flow control and back-pressure:
  - Data transmitted from a UCIe Die to a UCIe Retimer is flow-controlled using credits. These credits are on top of any underlying protocol credit mechanism (such as PH, PD credits in PCIe). These UCIe D2D credits must be for flow control across the two UCIe Retimers and any data transmitted to the UCIe Retimer must eventually be consumed by the remote UCIe die without any other dependency. Every UCIe Retimer must implement a Receiver Buffer for Flits that it receives from the UCIe die within its package. The Receiver buffer credits are advertised to the UCIe die during initial parameter exchanges for the D2D Adapter, and the UCIe die must not send any data to the UCIe Retimer if it does not have a credit for it. One credit corresponds to 256B of data (including any FEC, CRC, etc.). Credit returns are overloaded on the Valid framing (see Section 4.1.2). Credit counters at the UCIe Die are re-assigned to their initial advertised value whenever RDI states transition away from Active. UCIe Retimer must drain or dump (as applicable) the data in its receiver buffer before re-entering Active state.
  - Data transmitted from a UCIe Retimer to a UCIe die is not flow-controlled at the D2D adapter level. The UCIe Retimer may have its independent flow-control with the other UCIe Retimer if needed, which is beyond the scope of this specification.

1.4 UCIe Key Performance Targets

Table 1-4 gives a summary of the performance targets for UCIe Advanced and Standard Package configurations. Table 1-5 gives a summary of the performance targets for UCIe-3D.

Table 1-4. UCIe 2D and 2.5D Key Performance Targets

Metric	Link Speed/ Voltage	Advanced Package (x64)	Standard Package
Die Edge Bandwidth Density[a] (GB/s per mm)	4 GT/s	165	28
	8 GT/s	329	56
	12 GT/s	494	84
	16 GT/s	658	112
	24 GT/s	988	168
	32 GT/s	1317	224
Energy Efficiency[b] (pJ/bit)	0.7 V (Supply Voltage)	0.5 (<=12 GT/s)	0.5 (4 GT/s)
		0.6 (>=16 GT/s)	1.0 (<=16 GT/s)
		-	1.25 (32 GT/s)
	0.5 V (Supply Voltage)	0.25 (<=12 GT/s)	0.5 (<=16 GT/s)
		0.3 (>=16 GT/s)	0.75 (32 GT/s)
Latency Target[c]		<=2ns	

a. Die edge bandwidth density is defined as total I/O bandwidth in GB per sec per mm silicon die edge, with 45-um (Advanced Package) and 110-um (Standard Package) bump pitch. For a x32 Advanced Package module, the Die Edge Bandwidth Density is 50% of the corresponding value for x64.
b. Energy Efficiency (energy consumed per bit to traverse from FDI to bump and back to FDI) includes all the Adapter and Physical Layer-related circuitry including, but not limited to, Tx, Rx, PLL, Clock Distribution, etc. Channel reach and termination are discussed in Chapter 5.0.
c. Latency includes the latency of the Adapter and the Physical Layer (FDI to bump delay) on Tx and Rx. See Chapter 5.0 for details of Physical Layer latency. Latency target is based on 16 GT/s. Latency at other data rates may differ due to data rate-dependent aspects such as data accumulation and transfer time. Note that the latency target does not include the accumulation of bits required for processing; either within or across Flits.

Table 1-5. UCIe-3D Key Performance Targets

Metric	Link Speed/Voltage	UCIe-3D
Bandwidth Density[a] (GB/s/mm^2)	4 GT/s	4000
Energy Efficiency[b] (pJ/bit)	0.65 V (Supply Voltage)	0.05
Latency Target[c]		<= 125 ps

a. Bandwidth Density is provided for a 9-um bump pitch.
b. Energy Efficiency (energy consumed per bit) includes all the Tx, Rx, PLL, Clock Distribution, etc.
c. Latency includes the latency on Tx and Rx.

1.5 Interoperability

Package designers need to ensure that Dies that are connected on a package can inter-operate. This includes compatible package interconnect (e.g., Advanced vs. Standard), protocols, voltage levels, etc. It is strongly recommended that a Die adopts Transmitter voltage of less than 0.85 V so that the Die can inter-operate with a wide range of process nodes in the foreseeable future.

This specification comprehends interoperability across a wide range of bump pitch for Advanced Packaging options. It is expected that over time, the smaller bump pitches will be predominantly used. With smaller bump pitch, we expect designs will reduce the maximum advertised frequency (even though they can go to 32G) to optimize for area and to address the power delivery and thermal constraints of high bandwidth with reduced area. Table 1-6 summarizes these bump pitches across four groups. Interoperability is guaranteed within each group as well as across groups, based on the PHY dimension specified in Chapter 5.0. The performance targets provided in Table 1-4 are with the 45 um bump pitch, based on the technology widely deployed at the time of publication of UCIe 1.0 and UCIe 1.1 Specifications (2022 – 2023).

Table 1-6. Groups for different bump pitches

Bump Pitch (um)	Minimum Frequency (GT/s)	Expected Maximum Frequency (GT/s)
Group 1: 25 - 30	4	12
Group 2: 31 - 37	4	16
Group 3: 38 - 44	4	24
Group 4: 45 - 55	4	32

§ §

2.0 Protocol Layer

Universal Chiplet Interconnect express (UCIe) maps PCIe and CXL, as well as any Streaming protocol. Throughout the UCIe Specification, Protocol-related features are kept separate from Flit Formats and packetization. This is because UCIe provides different transport mechanisms that are not necessarily tied to protocol features (e.g., PCIe non-Flit mode packets are transported using CXL.io 68B Flit Format). Protocol features include the definitions of Transaction Layer and higher layers, as well as Link Layer features not related to Flit packing/Retry (e.g., Flow Control negotiations etc.).

The following terminology is used throughout this specification to identify Protocol-level features:

- PCIe Flit mode: To reference Flit mode-related Protocol features defined in *PCIe Base Specification*
- PCIe non-Flit mode: To reference non-Flit mode-related Protocol features defined in *PCIe Base Specification*
- CXL 68B Flit mode: To reference 68B Flit mode-related Protocol features defined in *CXL Specification*
- CXL 256B Flit mode: To reference 256B Flit mode-related Protocol features defined in *CXL Specification*

The following protocol mappings are supported over the UCIe mainband:

- PCIe Flit mode
- CXL 68B Flit mode, CXL 256B Flit Mode: If CXL is negotiated, each of CXL.io, CXL.cache, and CXL.mem protocols are negotiated independently.
- Streaming protocol: This offers generic modes for a user defined protocol to be transmitted using UCIe.
- Management Transport protocol: This allows transport of manageability packets.

Note: RCD/RCH/eRCD/eRCH are not supported. PCIe non-Flit Mode is supported using CXL.io 68B Flit Format as the transport mechanism.

The Protocol Layer requirements for interoperability are as follows:

- A Protocol Layer must support PCIe non-Flit mode if it is advertising the 68B Flit Mode parameter from Table 3-1.
- If a Protocol Layer supports CXL 256B Flit Mode, it must support PCIe Flit Mode and 68B Flit Mode as defined in *CXL Specification* for CXL.io protocol.
- A Protocol Layer advertising CXL is permitted to only support CXL 68B Flit Mode without supporting CXL 256B Flit Mode or PCIe Flit Mode

IMPLEMENTATION NOTE

Table 2-1 summarizes the mapping of the above rules from a specification version to a protocol mode.

Table 2-1. Specification to protocol mode requirements

Native Specification Supported[a]	PCIe Non-Flit Mode	CXL 68B Flit Mode	CXL 256B Flit Mode	PCIe Flit Mode
PCIe	Mandatory	N/A	N/A	Optional
CXL 2.0	Mandatory (for CXL.io)	Mandatory	N/A	N/A
CXL 3.0	Mandatory (for CXL.io)	Mandatory	Mandatory	Mandatory (for CXL.io)

a. The same table applies to derivative version numbers for the specifications.

The Die-to-Die (D2D) Adapter negotiates the protocol with the remote Link partner and communicates it to the Protocol Layer(s). For each protocol, UCIe supports multiple modes of operation (that must be negotiated with the remote Link partner depending on the advertised capabilities, Physical Layer Status as well as usage models). These modes have different Flit Formats and are defined to enable different trade-offs around efficiency, bandwidth and interoperability. The spectrum of supported protocols, advertised modes and Flit Formats must be determined at SoC integration time or during the Die-specific reset bring up flow. The Die-to-Die Adapter uses this information to negotiate the operational mode as a part of Link Training and informs the Protocol Layer over the Flit-aware Die-to-Die Interface (FDI). See Section 3.2 for parameter exchange rules in the Adapter.

The subsequent sections provide an overview of the different modes from the Protocol Layer's perspective, hence they cover the supported formats of operation as subsections per protocol. The Protocol Layer is responsible for transmitting data over FDI in accordance with the negotiated mode and Flit Format. The illustrations of the Flit Formats in this chapter show an example configuration of a 64B data path in the Protocol Layer mapped to a 64-Lane module of Advanced Package configuration on the Physical Link of UCIe. Certain Flit Formats have dedicated bit positions filled in by the Adapter, and details associated with these are illustrated separately in Chapter 3.0. For other Link widths, see the Byte to Lane mappings defined in Section 4.1.1. Figure 2-1 shows the legend for color-coding convention used when showing bytes within a Flit in the Flit Format examples in the UCIe Specification.

Figure 2-1. Color-coding Convention in Flit Format Byte Map Figures

Color Shading	Description
	Some bits populated by the Protocol Layer, some bits populated by the Adapter.
	All bits populated by Adapter.
	All bits populated by the Protocol Layer.

2.1 PCIe

UCIe supports the Flit Mode defined in *PCIe Base Specification*. See *PCIe Base Specification* for the protocol definition. UCIe supports the non-Flit Mode using the CXL.io 68B Flit Formats as the transport mechanism. There are five UCIe operating formats supported for PCIe, and these are defined in the subsections that follow.

2.1.1 Raw Format

This format is optional. All bytes are populated by the Protocol Layer. The intended usage is for UCIe Retimers transporting PCIe protocol. An example usage of this format is where a CPU and an I/O Device are in different Rack/chassis and connected through a UCIe Retimer using Off-Package Interconnect as shown in Figure 1-2. Retry, CRC and FEC (if applicable) are taken care of by the Protocol Layer when using Raw Format. It is strongly recommended for the UCIe Retimers to check and count errors using either the parity bits of the 6B FEC or the Flit Mode 8B CRC defined in *PCIe Base Specification* for this mode to help characterize the Off Package Interconnect (to characterize or debug the Link that is the dominant source of errors). See Section 3.3.1 as well.

2.1.2 Standard 256B End Header Flit Format

This format is mandatory when PCIe Flit Mode protocol is supported. It is the standard Flit Format defined in *PCIe Base Specification* for Flit Mode and the main motivation of supporting this Flit Format is to enable interoperability with vendors that only support the standard PCIe Flit Formats. The Protocol Layer must follow the Flit Formats for Flit transfer on FDI, driving 0 on the fields reserved for Die-to-Die Adapter. The PM and Link Management DLLPs are not used over UCIe. The other DLLPs (that are applicable for PCIe Flit Mode) and Flit Status definitions follow the same rules including packing as defined in *PCIe Base Specification*. It is strongly recommended for implementations to optimize out any 8b/10b, 128b/130b, and non-Flit Mode related CRC/Retry or framing logic from the Protocol Layer in order to obtain area and power efficient designs for UCIe applications. Portions of the DLP bytes must be driven by the Protocol Layer for Flit_Marker assignment; see Section 3.3.3 for details of the Flit Format.

2.1.3 68B Flit Format

This mode is mandatory when PCIe protocol or CXL protocol is supported. The transport mechanism for this is the same as CXL.io 68B Flit Formats. See Section 2.3.2 for the CXL.io DLLP rules that apply for Non-Flit Mode for PCIe as well. It is strongly recommended for implementations to optimize out any 8b/10b, 128b/130b and non-Flit Mode related CRC/Retry logic from the Protocol Layer in order to obtain area and power efficient designs for UCIe applications. To keep the framing rules consistent, Protocol Layer for PCIe non-Flit mode must still drive the LCRC bytes with a fixed value of 0, and the Receiver must ignore these bytes and never send any Ack or Nak DLLPs. Framing tokens are applied as defined for CXL.io 68B Flit Mode operation in *CXL Specification*. It is recommended for the transmitter to drive the sequence number, DLLP CRC, Frame CRC and Frame parity in STP to 0; the receiver must ignore these fields. Given that UCIe Adapter provides reliable Flit transport, framing errors, if detected by the Protocol Layer, are likely due to uncorrectable internal errors and it is permitted to treat them as such.

2.1.4 Standard 256B Start Header Flit Format

This is an optional format for PCIe Flit Mode, supported if Standard Start Header for PCIe protocol Capability is supported. The Protocol Layer must follow the Flit Formats for Flit transfer on FDI, driving 0 on the fields reserved for Die-to-Die Adapter. The PM and Link Management DLLPs are not used over UCIe. The other DLLPs (that are applicable for PCIe Flit Mode) and Flit Status definitions follow the same rules including packing as defined in *PCIe Base Specification*. It is strongly recommended for implementations to optimize out any 8b/10b, 128b/130b and non-Flit Mode related CRC/Retry or framing logic from the Protocol Layer in order to obtain area and power efficient designs for UCIe applications. Portions of the DLP bytes must be driven by the Protocol Layer for Flit_Marker assignment; see Section 3.3.3 for details of the Flit Format.

2.1.5 Latency-Optimized 256B with Optional Bytes Flit Format

This is an optional format for PCIe Flit Mode, supported if Latency-Optimized Flit with Optional Bytes for PCIe protocol capability is supported. It is the Latency-Optimized Flit with Optional Bytes Flit Format for PCIe, as defined in Section 3.3.4. The Protocol Layer must follow the Flit Formats for Flit transfer on FDI, driving 0 on the fields reserved for Die-to-Die Adapter. The PM and Link Management DLLPs are not used over UCIe. The other DLLPs (that are applicable for PCIe Flit Mode) and Flit Status definitions follow the same rules including packing as defined in *PCIe Base Specification*. It is strongly recommended for implementations to optimize out any 8b/10b, 128b/130b and non-Flit Mode related CRC/Retry or framing logic from the Protocol Layer in order to obtain area and power efficient designs for UCIe applications. Portions of the DLP bytes must be driven by the Protocol Layer for Flit_Marker assignment; see Section 3.3.4 for details of the Flit Format.

2.2 CXL 256B Flit Mode

See *CXL Specification* for details on the protocol layer messages and slot formats for "CXL 256B Flit Mode". There are four possible operational formats for this protocol mode (there are two formats in Section 2.2.3), defined in the subsections that follow. The light orange bytes are inserted by the Adapter (see Figure 2-1). In cases where these are shown as part of the main data path (e.g., in the Standard 256B Flit Format), the Protocol Layer must drive 0 on them on the Transmitter, and ignore them on the Receiver.

2.2.1 Raw Format

This format is optional. All bytes are populated by the Protocol Layer. The intended usage is for UCIe Retimers transporting CXL 256B Flit Mode protocol. An example usage of this format is where a CPU and an I/O Device are in different Rack/chassis and connected through a UCIe Retimer using Off-Package Interconnect. Retry, CRC and FEC are taken care of by the Protocol Layer. It is strongly recommended for the UCIe Retimers to check and count errors using either the parity bits of the 6B FEC or the Flit Mode 8B CRC or 6B CRC; depending on which Flit Format was enabled. This helps to characterize and debug the Off-Package Interconnect which is the dominant source of errors. For CXL.cachemem, Viral or poison containment (if applicable) must be handled within the Protocol Layer for this format. See Section 3.3.1 as well.

2.2.2 Latency-Optimized 256B Flit Formats

The support for this format is strongly recommended for "CXL 256B Flit Mode" over UCIe. Two Flit Formats are defined, which provide two independent operating points. These formats are derived from the Latency-Optimized Flits defined in *CXL Specification*. The only difference for the second Flit Format is that it gives higher Flit packing efficiency by providing Protocol Layer with extra bytes. For CXL.io this results in extra 4B of TLP information, and for CXL.cachemem it results in an extra 14B H-slot that can be packed in the Flit. This slot is ordered between Slots 7 and 8. It is included in both Groups B and C, similar to Slot 7. See *CXL Specification* for reference of the packing rules. Support for the first or second format is negotiated at the time of Link bring up. See Section 3.3.4 for the details for the Flit Formats.

The Latency-Optimized formats enable the Protocol Layer to consume the Flit at 128B boundary, reducing the accumulation latency significantly. When this format is negotiated, the Protocol Layer must follow this Flit Format for Flit transfer on FDI, driving 0 on the fields reserved for Die-to-Die Adapter.

The Ack, Nak, PM, and Link Management DLLPs are not used over UCIe for CXL.io for any of the 256B Flit Modes. The other DLLPs and Flit_Marker definitions follow the same rules as defined in *CXL Specification*. Portions of the DLP bytes must be driven by the Protocol Layer for Flit_Marker assignment; see Section 3.3.3 for details on how DLP bytes are driven.

For CXL.cachemem for this mode, FDI provides an **lp_corrupt_crc** signal to help optimize for latency while guaranteeing Viral containment. See Chapter 10.0 for details of interface rules for Viral containment.

2.2.3 Standard 256B Start Header Flit Format

This format is mandatory when "CXL 256B Flit Mode" protocol is supported. It is the Standard 256B Flit Format defined in *CXL Specification* for 256B Flit Mode and the main motivation of supporting this Flit Format is to enable interoperability with vendors that only support the Standard 256B Flit Formats. The Protocol Layer must follow the Flit Formats for Flit transfer on FDI, driving 0 on the fields reserved for Die-to-Die Adapter. The Ack, Nak, PM, and Link Management DLLPs are not used over UCIe for CXL.io. The other DLLPs and Flit Status definitions follow the same rules and packing as defined in *CXL Specification*. Portions of the DLP bytes must be driven by the Protocol Layer for Flit_Marker assignment; see Section 3.3.3 for details of the Flit Formats and on how DLP bytes are driven.

For CXL.cachemem in this format, FDI provides an **lp_corrupt_crc** signal to help optimize for latency while guaranteeing Viral containment. See Section 10.2 for details of interface rules for Viral containment.

See Section 3.3.3 for details about this Flit Format.

2.3 CXL 68B Flit Mode

The *CXL Specification* provides details on the protocol layer messages and slot formats for CXL 68B Flit Mode. There are two operational formats possible for this protocol, and these are defined in the subsections that follow. The light orange bytes are inserted by the Adapter (see Figure 2-1).

2.3.1 Raw Format

This format is optional. All bytes are populated by the Protocol Layer. The intended usage is for UCIe Retimers transporting "CXL 68B Flit Mode" protocol. An example usage of this format is where a CPU and an I/O Device are in different Rack/chassis and connected through a UCIe Retimer using an Off-Package Interconnect. Retry and CRC are taken care of by the Protocol Layer. See Section 3.3.1 as well.

2.3.2 68B Flit Format

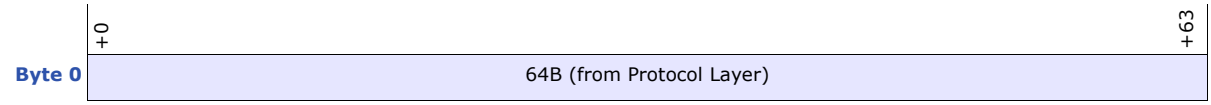
This format is mandatory when CXL 68B Flit Mode protocol is negotiated. This follows the corresponding 68B Flit Format defined in *CXL Specification* and the main motivation of supporting this Flit Format is to enable interoperability with vendors that only support the baseline CXL formats. The Protocol Layer presents 64B of the Flit (excluding the Protocol ID and CRC) on FDI (shown in Figure 2-2), and the Die-to-Die Adapter inserts a 2B Flit Header and 2B CRC and performs the byte shifting required to arrange the Flits in the format shown in Figure 3-11.

The Ack, Nak, and PM DLLPs are not used for CXL.io in this mode. Credit updates and other remaining DLLPs for CXL.io are transmitted in the Flits as defined in *CXL Specification*. For CXL.io, the Transmitter must not implement Retry in the Protocol Layer (because Retry is handled in the Adapter). To keep the framing rules consistent, Protocol Layer for CXL.io must still drive the LCRC bytes with a fixed value of 0, and the Receiver must ignore these bytes and never send any Ack or Nak DLLPs. Framing tokens are applied as defined for CXL.io 68B Flit Mode operation. It is recommended for the transmitter to drive the sequence number, DLLP CRC, Frame CRC and Frame parity in STP to 0. The receiver must ignore these fields. Given that UCIe Adapter provides reliable Flit

transport, framing errors, if detected by the Protocol Layer are likely due to uncorrectable internal errors and it is permitted to treat them as such.

For CXL.cachemem, the "Ak" field defined by *CXL Specification* in the Flit is reserved, and the Retry Flits are not used (because Retry is handled in the Adapter). Link Initialization begins with sending the INIT.Param Flit without waiting for any received Flits. Viral containment (if applicable) must be handled within the Protocol Layer for the 68B Flit Mode. *CXL Specification* introduced Error Isolation as a way to reduce the blast radius of downstream component fatal errors compared to CXL Viral Handling and provide a scalable way to handle device failures across a network of switches shared between multiple Hosts. Specifically, Viral relies on a complete host reset to recover whereas Error Isolation may recover by resetting the virtual hierarchy below the root port. Because CXL-defined Retry Flits (which carry the viral notification for 68B Flits in CXL) are not used in 68B Flit mode in UCIe, it is recommended for implementations to rely on error isolation at the CXL Root Port for fatal errors on CXL.cachemem downstream components in 68B Flit mode (similar to Downstream Port Containment for CXL.io).

Figure 2-2. 68B Flit Format on FDI[a]



a. See Figure 2-1 for color mapping.

2.4 Streaming Protocol

This is the default protocol that must be advertised if none of the PCIe or CXL protocols are going to be advertised and negotiated with the remote Link partner. If Streaming Flit Format capability is not supported, then the operational formats that can be used are either Raw Format or vendor defined extensions. Streaming Flit Format capability is supported if any of 68B Flit Format for Streaming Protocol, Standard 256B End Header Flit Format for Streaming Protocol, Standard 256B Start Header Flit Format for Streaming Protocol, Latency-Optimized 256B Flit Format without Optional Bytes for Streaming Protocol or Latency-Optimized 256B Flit Format with Optional Bytes for Streaming Protocol bits are set in the UCIe Link Capability register.

2.4.1 Raw Format

This is mandatory for Streaming protocol support in Adapter implementations. Protocol Layer interoperability is vendor defined. All bytes are populated by the Protocol Layer. See Section 3.3.1 as well.

2.4.2 68B Flit Format

This format is only applicable if Streaming Flit Format capability is supported. It is an optional format that permits implementations to utilize the 68B Flit Format from the Adapter for Streaming protocols. See Section 3.3.2 for details of the Flit Format.

The Protocol Layer presents 64B per Flit on FDI, and the Die-to-Die Adapter inserts a 2B Flit Header and 2B CRC and performs the byte shifting required to arrange the Flits in the format shown in Figure 3-11. On the receive data path, the Adapter strips out the Flit Header and CRC bytes to only present the 64B per Flit to the Protocol Layer on FDI.

2.4.3 Standard 256B Flit Formats

This format is only applicable if Streaming Flit Format capability is supported. Implementations are permitted to utilize the Standard 256B Start Header Flit Format or Standard 256B End Header Flit

Format from the Adapter for Streaming protocols. See Section 3.3.3 for details of the Flit Format and to see which of the reserved fields in the Flit Header are driven by the Protocol Layer. The Protocol Layer presents 256B per Flit on FDI, driving 0b on the bits reserved for the Adapter. The Adapter fills in the applicable Flit Header and CRC bytes. On the Rx datapath, the Adapter forwards the Flit received from the Link as it is, and the Protocol Layer must ignore the bits reserved for the Adapter (for example the CRC bits).

2.4.4 Latency-Optimized 256B Flit Formats

This format is applicable only when Streaming Flit Format capability is supported. Implementations are permitted to utilize the Latency-Optimized 256B with Optional Bytes Flit Format or Latency-Optimized 256B without Optional Bytes Flit Format for Streaming protocols. See Section 3.3.4 for details of the Flit Format and to see which of the reserved fields in the Flit Header are driven by the Protocol Layer. The Protocol Layer presents 256B per Flit on FDI, driving 0b on the bits reserved for the Adapter. The Adapter fills in the applicable Flit Header and CRC bytes. On the Rx datapath, the Adapter forwards the Flit received from the Link as is, and the Protocol Layer must ignore the bits reserved for the Adapter (e.g., the CRC bits).

2.5 Management Transport Protocol

This protocol is used to carry management network packets over the mainband. The format for these packets is shown in Section 8.2.2.2. The 68B Flit Format is not permitted for this protocol. Raw mode and any of the 256B Flit Formats are permitted for this protocol. When using the 256B Flit Formats, the Protocol Layer presents 256B per Flit on the FDI, driving 0 on the bits that are reserved for the Adapter. The Adapter fills in the applicable Flit Header and CRC bytes. On the Rx data path, the Adapter forwards the Flit received from the Link as is, and the Management Port Gateway must ignore the bits reserved for the Adapter (e.g., the CRC bits).

See Section 8.2.5.2.3 for details of mapping the Management Transport Packets (MTPs) over Management Flits.

§ §

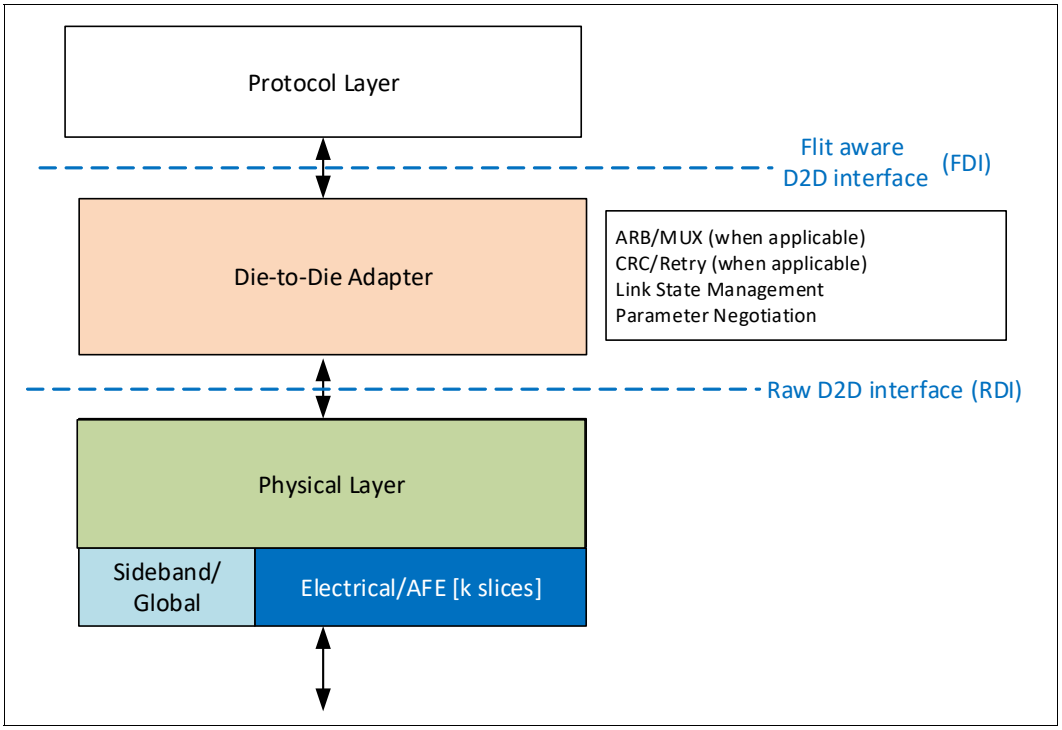
3.0 Die-to-Die Adapter

The Die-to-Die Adapter is responsible for:

- Reliable data transfer (performing CRC computation and Retry, or parity computation when applicable)
- Arbitration and Muxing (in case of multiple Protocol Layers)
- Link State Management
- Protocol and Parameter negotiation with the remote Link partner.

Figure 3-1 shows a high level description of the functionality of the Adapter.

Figure 3-1. Functionalities in the Die-to-Die Adapter



The Adapter interfaces to the Protocol Layer using one or more instances of the Flit-aware Die-to-Die interface (FDI), and it interfaces to the Physical Layer using the raw Die-to-Die interface (RDI). See Chapter 10.0 for interface details and operation.

The D2D Adapter must follow the same rules as the Protocol Layer for protocol interoperability requirements. Figure 3-2 shows example configurations for the Protocol Layer and the Adapter, where the Protocol identifiers (e.g., PCIe) only signify the protocol, and not the Flit Formats. To provide cost

and efficiency trade-offs, UCIe allows configurations in which two protocol stacks are multiplexed onto the same physical Link.

3.1 Stack Multiplexing

If the Multi_Protocol_Enable parameter is negotiated, two stacks multiplexed on the same physical Link is supported when each protocol stack needs half the bandwidth that the Physical Layer provides. Both stacks must be of the same protocol with the same protocol capabilities. When Multi_Protocol_Enable and Management Transport protocol are negotiated for mainband and the Protocol Layer implements the Management Port Gateway multiplexer (MPG mux), the MPG mux must be present on both stacks and the same protocols must be present in both stacks. For example, in Figure 8-27 the Multi_Protocol_Enable parameter can be negotiated for config b if both stacks in this configuration have PCIe or both stacks have Streaming. Similarly, the Multi_Protocol_Enable parameter can be negotiated for config d in Figure 8-27 if both CXL stacks are identical.

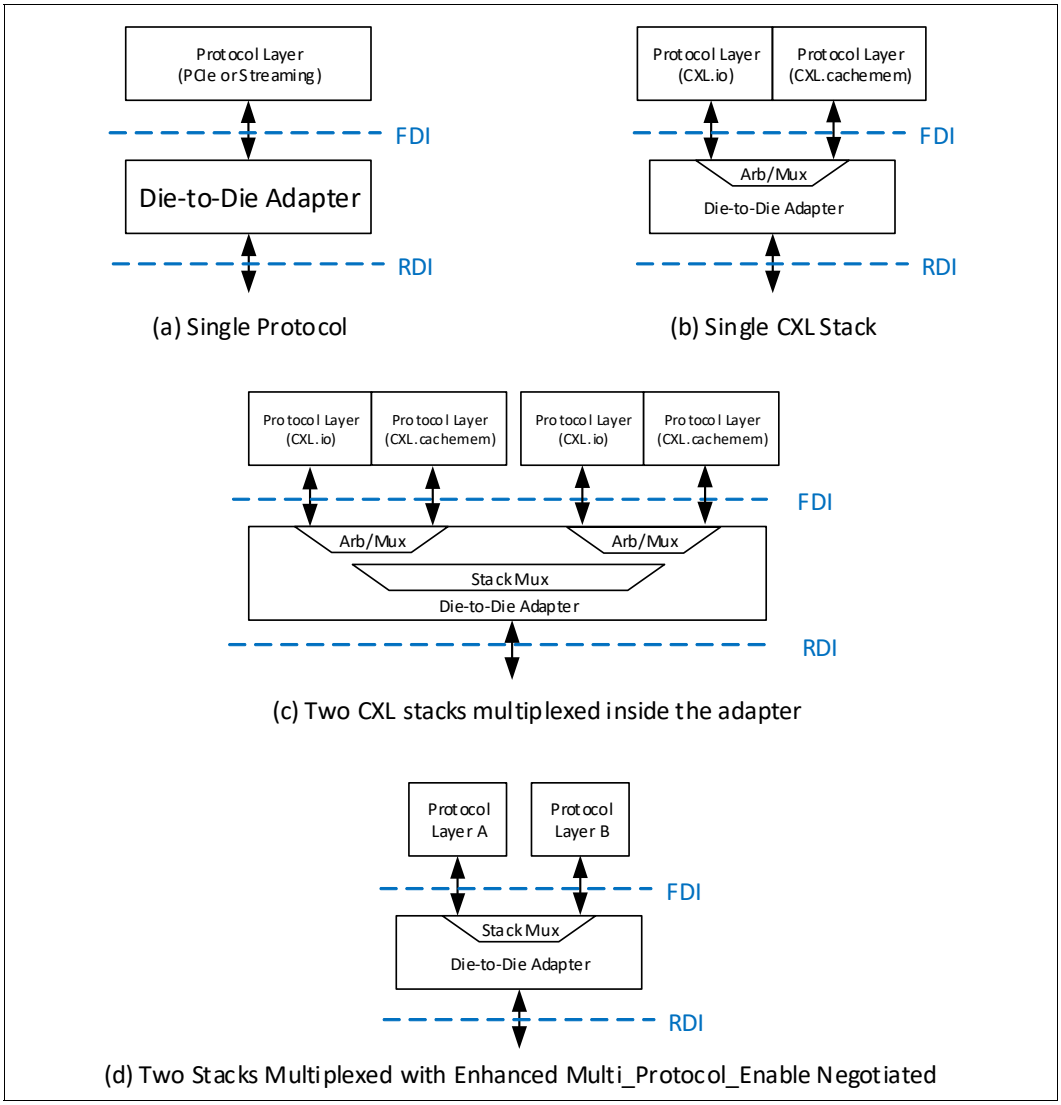
When Multi_Protocol_Enable is supported and negotiated, the Adapter must guarantee that it will not send consecutive flits from the same protocol stack on the Link. This applies in all cases including when Flits are sourced from FDI, from Retry Buffer, and when the data stream is paused and restarted. Adapter is permitted to insert NOP Flits to guarantee this (these Flits bypass the Tx Retry buffer, and are not forwarded to the Protocol Layer on the receiver). When Flits are transmitted from the Retry Buffer, it is required to insert NOP Flits as needed to avoid sending consecutive Flits from the same Protocol stack. When Management Transport protocol is negotiated for mainband with Multi_Protocol_Enable, the Management Flit carries the same stack identifier as the Protocol Layer it is multiplexed with. From the Adapter perspective, for the purposes of throttling and interleaving, it is treated the same as flits received from the corresponding Protocol Layer. Note that there is no fixed pattern of Flits alternating from different Protocol Layers. For example, a Flit from Protocol Stack 0 followed by a NOP Flit, followed by a Flit from Protocol Stack 0 is a valid transmit pattern. A NOP Flit is defined as a Flit where the protocol identifier in the Flit Header corresponds to the D2D Adapter, and the body of the Flit is filled with all 0 data (the NOP Flit is defined for all Flit Formats supported by the Adapter, for all cases when it is operating in Raw Format). It is permitted for NOP flits to bypass the Retry buffer, as long as the Adapter guarantees that it is not sending consecutive Flits for any of the Protocol Layers. On the receiving side, the Adapter must not forward these NOP flits to the Protocol Layer. The receiving Protocol Layer must be capable of receiving consecutive chunks of the same Flit at the maximum Link speed, but it will not receive consecutive Flits. In addition to the transfer rate, both protocol stacks must operate with the same protocol and Flit Formats. Multi_Protocol_Enable and Raw Format are mutually exclusive. Each stack is given a single bit stack identifier that is carried along with the Flit header for de-multiplexing of Flits on the Receiver. The Stack Mux shown maintains independent Link state machines for each protocol stack. Link State transition-related sideband messages have unique message codes to identify which stack's Link State Management is affected by that message.

> **IMPLEMENTATION NOTE**
>
> The primary motivation for enabling the Multi_Protocol_Enable parameter is to allow implementations to take advantage of the higher bandwidth provided by the UCIe Link for lower-bandwidth individual Protocol Layers, without the need to make a lot of changes to the UCIe Link. For example, two Protocol Layers that support the maximum bandwidth for CXL 68B Flit Mode (i.e., the equivalent of 32 GT/s CXL SERDES bandwidth) can be multiplexed over a UCIe Link that supports their aggregate bandwidth.

If the Enhanced Multi_Protocol_Enable parameter is negotiated, dynamic multiplexing between two stacks of the same or different protocols on the same physical Link is supported. When Enhanced

Multi_Protocol_Enable and Management Transport protocol are negotiated, each stack can have different protocols with or without MPG mux. For example, in Figure 8-27, the Enhanced Multi_Protocol_Enable parameter must be negotiated for configs e, f, and h. The parameter is also negotiated for configs b and d if the two stacks have different protocol pairs. Both protocol stacks and the Adapter must support a common Flit Format for this feature to be enabled. "Enhanced Multi_Protocol_Enable" and Raw Format are mutually exclusive. The Adapter must advertise the maximum percentage of bandwidth that the receiver for each Protocol Layer can accept. The Adapter transmitter must support 100% (no throttling) and throttling one or both Protocol Layer(s) to 50% of maximum bandwidth. When 50% of the maximum bandwidth is advertised for a stack by an Adapter, the remote Link partner must guarantee that it will not send consecutive Flits for the same stack on the Link. This applies in all cases including when Flits are sourced from FDI, from Retry Buffer, and when the data stream is paused and restarted. Adapter is permitted to insert NOP Flits to guarantee this (these Flits bypass the Tx Retry buffer, and are not forwarded to the Protocol Layer on the receiver). When Flits are transmitted from the Retry Buffer, it is required to insert NOP Flits as needed to avoid exceeding the negotiated maximum bandwidth. The receiving Protocol Layer must be capable of sinking Flits at the advertised maximum bandwidth percentage; in addition, Protocol Layer must be able to receive consecutive chunks of the same Flit at the maximum advertised Link speed. When this capability is supported, the Adapter must be capable of allowing each Protocol Layer to independently utilize 100% of the Link bandwidth. Furthermore, the arbitration is per Flit, and the Adapter must support round robin arbitration between the Protocol Layers if both of them are permitted to use 100% of the Link bandwidth. Additional implementation specific arbitration schemes are permitted as long as they are per Flit and do not violate the maximum bandwidth percentage advertised by the remote Adapter for a given stack. The Flit header has a single bit stack identifier to identify the destination stack for the flit. The Stack Mux maintains independent Link state machines for each protocol stack. Link State transition-related sideband messages have unique message codes to identify which stack's Link State Management is affected by that message.

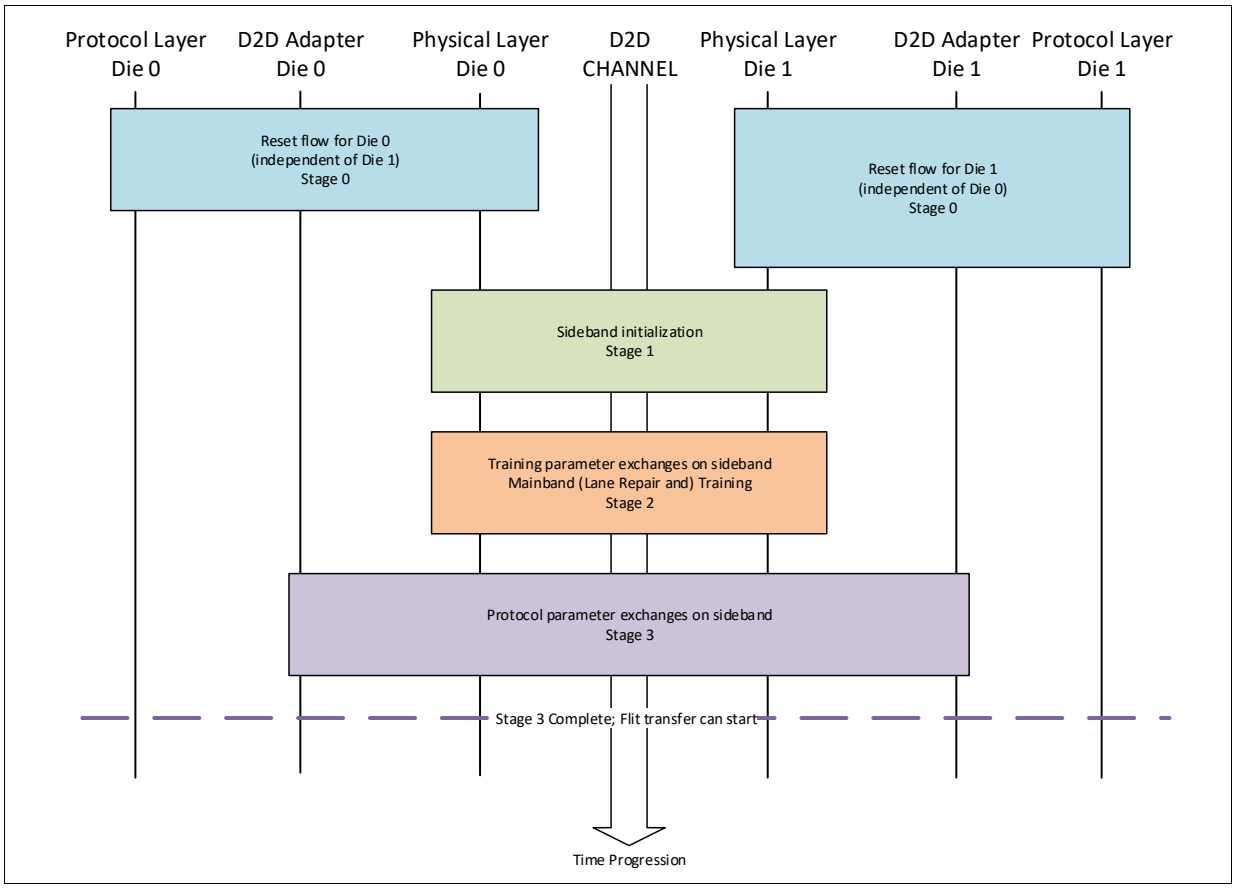
Figure 3-2. Example Configurations



3.2 Link Initialization

Link Initialization consists of four stages before protocol Flit transfer can begin on mainband. Figure 3-3 shows the high-level steps involved in the Link initialization flow for UCIe. Stage 0 is die-specific and happens independently for each die; the corresponding boxes in Figure 3-3 are of different sizes to denote that different die can take different amount of time to finish Stage 0. Stage 1 involves sideband initialization. Stage 2 involves mainband training and repair. Details of Stage 1 and Stage 2 are provided in Chapter 4.0. Stage 3 involves parameter exchanges between Adapters to negotiate the protocol and Flit Formats and is covered in Section 3.2.1.

Figure 3-3. Stages of UCIe Link Initialization



3.2.1 Stage 3 of Link Initialization: Adapter Initialization

Stage 2 is complete when the RDI state machine moves to Active State. The initialization flow on RDI to transition the state from Reset to Active is described in Section 10.1.6. Once Stage 2 is complete, the Adapter must follow a sequence of steps in order to determine Local Capabilities, complete Parameter Exchanges, and bring FDI state machine to Active.

3.2.1.1 Part 1: Determine Local Capabilities

The Adapter must determine the results of Physical Layer training and if Retry is needed for the given Link speed and configuration. See Section 3.8 for the rules on when Retry must be enabled for Link operation. If the Adapter is capable of supporting Retry, it must advertise this capability to the remote

Link partner during Parameter Exchanges. For UCIe Retimers, the Adapter must also determine the credits to be advertised for the Retimer Receiver Buffer. Each credit corresponds to 256B of Mainband data storage.

3.2.1.2 Part 2: Parameter Exchange with Remote Link Partner

The following list of capabilities must be negotiated between Link partners. The capabilities (if enabled) are transmitted to the remote Link partner using a sideband message. In the section below, "advertised" means that the corresponding bit is 1b in the {AdvCap.Adapter} sideband message.

Table 3-1. Capabilities that Must Be Negotiated between Link Partners (Sheet 1 of 2)

Capability	Description and Requirements
"Raw Format"	This parameter is advertised if the corresponding bit in the UCIe Link Control register is 1b. Software/Firmware enables this based on system usage scenario. If the PCIe or CXL protocols are not supported, and Streaming protocol is to be negotiated without any vendor-specific extensions and without Streaming Flit Format capability support, "Raw Format" must be 1b and advertised. If Streaming Flit Format capability or Enhanced Multi-Protocol capability is supported, then this must be advertised as 1b only if Raw Format is the intended format of operation. Software/firmware-based control on setting the corresponding UCIe Link Control is permitted to enable this.
"68B Flit Mode"	This is a protocol parameter. This must be advertised if the Adapter and Protocol Layer support CXL 68B Flit mode (mandatory for CXL) or PCIe Non-Flit mode (mandatory for PCIe). If PCIe Non-Flit mode is the final negotiated protocol, it will use the CXL.io 68B Flit mode formats as defined in *CXL Specification*. This is an advertised Protocol for Stack 0 if "Enhanced Multi_Protocol_Enable" is supported and enabled.
"CXL 256B Flit Mode"	This is a protocol parameter. This must be advertised if the Adapter and Protocol Layer support CXL 256B Flit mode. This is an advertised Protocol for Stack 0 if Enhanced Multi-Protocol capability is supported and enabled.
"PCIe Flit Mode"	This is a protocol parameter. This must be advertised if the Adapter and Protocol Layer support PCIe Flit mode. This is an advertised Protocol for Stack 0 if Enhanced Multi-Protocol capability is supported and enabled.
"Streaming"	This is a protocol parameter. This must be advertised if the Adapter and Protocol Layer support Streaming protocol in Raw Format or Streaming Flit Format capability is supported and the corresponding capabilities are enabled. This is an advertised Protocol for Stack 0 if Enhanced Multi-Protocol capability is supported and enabled. PCIe or CXL protocol must not be advertised if Streaming is advertised for a given Protocol Layer.
"Retry"	This must be advertised if the Adapter supports Retry. With the exception of the Link operating in Raw Format, the Link cannot be operational if the Adapter has determined Retry is needed, but "Retry" is not advertised or negotiated. See also Section 3.8.
"Multi_Protocol_Enable"	This must only be advertised if the Adapter is connected to multiple FDI instances corresponding to two sets of Protocol Layers. It must only be advertised if the Adapter (or SoC firmware in Stage 0 of Link Initialization) has determined that the UCIe Link must be operated in this mode. Both "Stack0_Enable" and "Stack1_Enable" must be 1b if this bit is advertised.
"Stack0_Enable"	This must be advertised if the Protocol Layer corresponding to Stack 0 exists and is enabled for operation with support for the advertised protocols.
"Stack1_Enable"	This must be advertised if the Protocol Layer corresponding to Stack 1 exists and is enabled for operation with support for the advertised protocols.
"CXL_LatOpt_Fmt5"	This must be advertised if the Adapter and Protocol Layer support *Format 5* defined in Section 3.3.4. The Protocol Layer does not take advantage of the spare bytes in this Flit Format. This must not be advertised if CXL protocol and CXL 256B Flit mode are not supported or enabled.
"CXL_LatOpt_Fmt6"	This must be advertised if the Adapter and Protocol Layer support *Format 6* defined in Section 3.3.4. The Protocol Layer takes advantage of the spare bytes in this Flit Format. This must not be advertised if CXL protocol and CXL 256B Flit mode are not supported or enabled.
"Retimer"	This must be advertised if the Adapter of a UCIe Retimer is performing Parameter Exchanges with a UCIe Die within its package.
"Retimer_Credits"	This is a 9-bit value advertising the total credits available for Retimer's Receiver Buffer. Each credit corresponds to 256B data. This parameter is applicable only when "Retimer" is 1b.

Table 3-1. Capabilities that Must Be Negotiated between Link Partners (Sheet 2 of 2)

Capability	Description and Requirements
"DP"	This is set by Downstream Ports to inform the remote Link partner that it is a Downstream Port. It is useful for Retimers to identify whether they are connected to a Downstream Port UCIe die. It is currently only applicable for PCIe and CXL protocols; however, Streaming protocols are not precluded from utilizing this bit. If Enhanced Multi-Protocol capability is supported, this bit is applicable if either of the Protocol Layers is PCIe or CXL. This bit must be set to 0b if "Retimer" is set to 1b.
"UP"	This is set by Upstream Ports to inform the remote Link partner that it is an Upstream Port. It is useful for Retimers to identify whether they are connected to an Upstream Port UCIe die. It is currently only applicable for PCIe and CXL protocols; however, Streaming protocols are not precluded from utilizing this bit. If Enhanced Multi-Protocol capability is supported, this bit is applicable if either of the Protocol Layers is PCIe or CXL. This bit must be set to 0b if "Retimer" is set to 1b.
"68B Flit Format"	This must be advertised if any of the following are true: • Enhanced Multi-Protocol capability is supported and enabled, AND the 68B Flit Format is supported and enabled • The 68B Flit Format for Streaming Protocol capability is supported and enabled Otherwise, it must be set to 0b.
"Standard 256B End Header Flit Format"	This must be advertised if any of the following are true: • Enhanced Multi-Protocol capability is supported and enabled, AND the Standard 256B End Header Flit Format is supported and enabled • The Standard 256B End Header Flit Format for Streaming Protocol capability is supported and enabled Otherwise, it must be set to 0b.
"Standard 256B Start Header Flit Format"	This must be advertised if any of the following are true: • PCIe Flit mode is advertised and Standard Start Header for PCIe protocol capability is supported and enabled • Enhanced Multi-Protocol capability is supported and enabled, AND the Standard 256B Start Header Flit Format is supported and enabled • The Standard 256B Start Header Flit Format for Streaming Protocol capability is supported and enabled Otherwise, it must be set to 0b.
"Latency-Optimized 256B without Optional Bytes Flit Format"	This must be advertised if any of the following are true: • Enhanced Multi-Protocol capability is supported and enabled, AND the Latency-Optimized 256B without Optional Bytes Flit Format is supported and enabled • The Latency-Optimized 256B without Optional Bytes Flit Format for Streaming Protocol capability is supported and enabled Otherwise, it must be set to 0b.
"Latency-Optimized 256B with Optional Bytes Flit Format"	This must be advertised if any of the following are true: • PCIe Flit mode is advertised and Latency-Optimized Flit with Optional Bytes for PCIe protocol capability is supported and enabled • Enhanced Multi-Protocol capability is supported and enabled, AND the Latency-Optimized 256B with Optional Bytes Flit Format is supported and enabled • The Latency-Optimized 256B with Optional Bytes Flit Format for Streaming Protocol capability is supported and enabled Otherwise, it must be set to 0b.
"Enhanced Multi_Protocol_Enable"	This must only be advertised if the Adapter is connected to multiple FDI instances corresponding to two sets of Protocol Layers. The two sets of Protocol Layers are permitted to be different protocols, but must support at least one common Flit Format. This must only be advertised if the Enhanced Multi-Protocol capability is supported and enabled; otherwise, it must be set to 0b. Both "Stack0_Enable" and "Stack1_Enable" must be 1b if this bit is advertised.
"Stack 0 Maximum Bandwidth_Limit"	This must be advertised if Enhanced Multi_Protocol_Enable is advertised and the Stack 0 protocol Receiver is limited to 50% of the maximum bandwidth; otherwise, it must be set to 0b.
"Stack 1 Maximum Bandwidth_Limit"	This must be advertised if Enhanced Multi_Protocol_Enable is advertised and the Stack 1 protocol Receiver is limited to 50% of the maximum bandwidth; otherwise, it must be set to 0b.
"Management Transport Protocol"	This bit must be set to 1 if the Protocol Layer and Adapter both support Management Transport protocol (either as the only protocol or multiplexed with one of CXL.io, PCIe, or Streaming). The mechanism by which this bit is set to 1 is implementation-specific.

Once local capabilities are established, the Adapter sends the {AdvCap.Adapter} sideband message advertising its capabilities to the remote Link partner.

If PCIe or CXL protocol support is going to be advertised, the Upstream Port (UP) Adapter must wait for the first {AdvCap.Adapter} message from the Downstream Port (DP) Adapter, review the capabilities advertised by DP and then send its own sideband message of advertised capabilities. UP is permitted to change its advertised capabilities based on DP capabilities. Once the DP receives the capability advertisement message from the UP, the DP responds with the Finalized Configuration using {FinCap.Adapter} sideband message to the UP as shown in Figure 3-4. See Section 7.1.2.3 to see the message format for the relevant sideband messages.

Final determination for Protocol parameters:

- If "68B Flit Mode" is advertised by both Link partners, it is set to 1 in the {FinCap.Adapter} message
- If "CXL 256B Flit Mode" is advertised by both Link partners, it is set to 1 in the {FinCap.Adapter} message
- If "PCIe Flit Mode" is advertised by both Link partners, "PCIe Flit Mode" bit is set to 1 in the {FinCap.Adapter} message
- If Streaming protocol is negotiated, no {FinCap.Adapter} messages are exchanged for that stack.

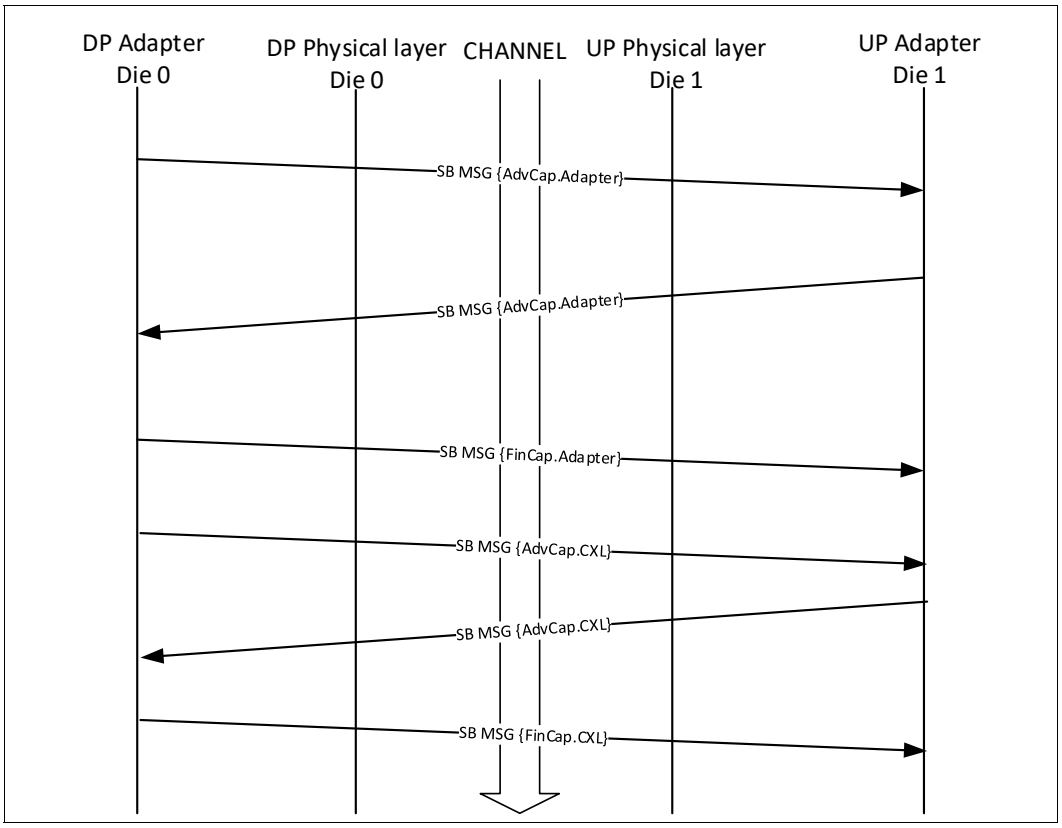
If "68B Flit Mode" or "CXL 256B Flit Mode" is set in the {FinCap.Adapter} message, there must be another handshake of Parameter Exchanges using the {AdvCap.CXL} and the {FinCap.CXL} messages to determine the details associated with this mode. Note that because CXL 68B Flit Mode protocol is mandatory for CXL, and because PCIe Non-Flit Mode protocol is mandatory for PCIe, the "68B Flit Mode" parameter is always set to 1 for CXL or PCIe protocols. This additional handshake is shown in Figure 3-4. The combination of {FinCap.CXL} and {FinCap.Adapter} determine the Protocol and Flit Format. See Section 7.1.2.3 for the message format of the relevant sideband messages. See Section 3.4 for how Protocol and Flit Formats are determined.

Final determination for other parameters if CXL or PCIe protocol is negotiated:

- If "Raw Format" is advertised by both Link partners, "Raw Format" is set to 1 in the {FinCap.Adapter} message.
- If both Link partners advertised "Retry" and "Raw Format" is not negotiated, Adapter Retry is enabled and "Retry" is set to 1 in the {FinCap.Adapter} message.
- If both Link partners advertised "Enhanced Multi_Protocol_Enable", both Stack 0 and Stack 1 are enabled by the adapter, and all three parameters ("Enhanced Multi_Protocol_Enable", "Stack0_Enable" and "Stack1_Enable") are each set to 1 in the {FinCap.Adapter} message (if a {FinCap.Adapter} message is required to be sent).
- If both Link partners advertised "Multi_Protocol_Enable" and "Enhanced Multi_Protocol_Enable" is not negotiated, both Stack 0 and Stack 1 are enabled by the Adapter, and all three parameters ("Multi_Protocol_Enable", "Stack0_Enable", and "Stack1_Enable") are each set to 1 in the {FinCap.Adapter} message.
- If neither "Enhanced Multi_Protocol_Enable" nor "Multi_Protocol_Enable" is negotiated, then the lowest common denominator is used to determine whether Stack 0 or Stack 1 is enabled, and the corresponding bit is set to 1 in the {FinCap.Adapter} message. If both Stack enables are advertised, then Stack 0 is selected for operational mode and only Stack0_Enable is set to 1 in the {FinCap.Adapter} message.
- If CXL_LatOpt_Fmt5 is advertised by both Link partners, then it is set to 1 in the {FinCap.Adapter} message.

- If CXL_LatOpt_Fmt6 is advertised by both Link partners, then it is set to 1 in the {FinCap.Adapter} message.

Figure 3-4. Parameter Exchange for CXL or PCIe (i.e., "68B Flit Mode" or "CXL 256B Flit Mode" is 1 in {FinCap.Adapter})



If Streaming protocol is negotiated, there is no notion of DP and UP for parameter exchanges and each side independently advertises its capabilities. Additional Vendor Defined sideband messages are permitted to be exchanged to negotiate vendor-specific extensions. See Table 7-8 and Table 7-10 for additional descriptions of Vendor Defined sideband messages. Similarly, if Management Transport protocol is negotiated on a stack without "Streaming protocol," "CXL 256B Flit mode," or "PCIe Flit mode," there is no notion of DP and UP for parameter exchanges and each side independently advertises its capabilities.

The Finalized Configuration is implicitly determined based on the intersection of capabilities advertised by each side:

- Flit Formats are chosen based on the Truth Table resolution provided in Table 3-10
- If both Link partners advertised Retry, and "Raw Format" is not negotiated, then Adapter Retry is enabled
- If "Multi_Protocol_Enable" is negotiated, both Stack 0 and Stack 1 are enabled by the adapter
- If neither "Multi_Protocol_Enable" nor "Enhanced Multi_Protocol_Enable" is advertised by at least one of the Link partners, then the lowest common denominator is used to determine whether Stack 0 or Stack 1 is enabled (i.e., if both Stack enables are advertised, then Stack 0 is selected for operational mode)

{FinCap.*} messages are not sent for Streaming protocol. Adapter must determine vendor specific requirements in an implementation specific manner.

If "Enhanced Multi_Protocol_Enable" is negotiated, the {AdvCap.Adapter} and if applicable, the {FinCap.Adapter} messages determine the negotiated Flit Format of operation as well as the protocol for Stack 0. The Adapter uses {MultiProtAdvCap.Adapter} and if applicable, the {MultiProtFinCap.Adapter} sideband messages to negotiate the Stack 1 protocol. For Stack 1, if PCIe or CXL protocol support is going to be advertised, the UP Adapter must wait for the first message from the DP Adapter, review the capabilities advertised by DP and then send its own sideband message of advertised capabilities. UP is permitted to change its advertised capabilities based on DP capabilities. In the section below, "advertised" means that the corresponding bit is 1b in the {MultiProtAdvCap.Adapter} sideband message.

- "68B Flit Mode": This must be advertised if the Adapter and Protocol Layer support CXL 68B Flit Mode or PCIe Non-Flit Mode on Stack 1.
- "CXL 256B Flit Mode": This must be advertised if the Adapter and Protocol Layer support CXL 256B Flit Mode on Stack 1.
- "PCIe Flit Mode": This must be advertised if the Adapter and Protocol Layer support PCIe Flit Mode on Stack 1.
- "Streaming": This must be advertised if the Adapter and Protocol Layer support Streaming Flit Mode on Stack 1.
- "Management Transport Protocol": This must be advertised if the Protocol Layer supports Management Transport protocol on Stack 1.

If "68B Flit Mode" or "CXL 256B Flit Mode" is set in the {MultiProtFinCap.Adapter} message, there must be another handshake of Parameter Exchanges using the {AdvCap.CXL} and the {FinCap.CXL} messages to determine the details associated with this mode. The non-Stall {*.CXL} messages are sent with a MsgInfo encoding of 0001h indicating that these messages are for Stack 1 negotiation.

Figure 3-5 to Figure 3-9 represent examples of different scenarios where Stack 0 and Stack 1 are of different protocols.

Figure 3-5. Both Stacks are CXL or PCIe

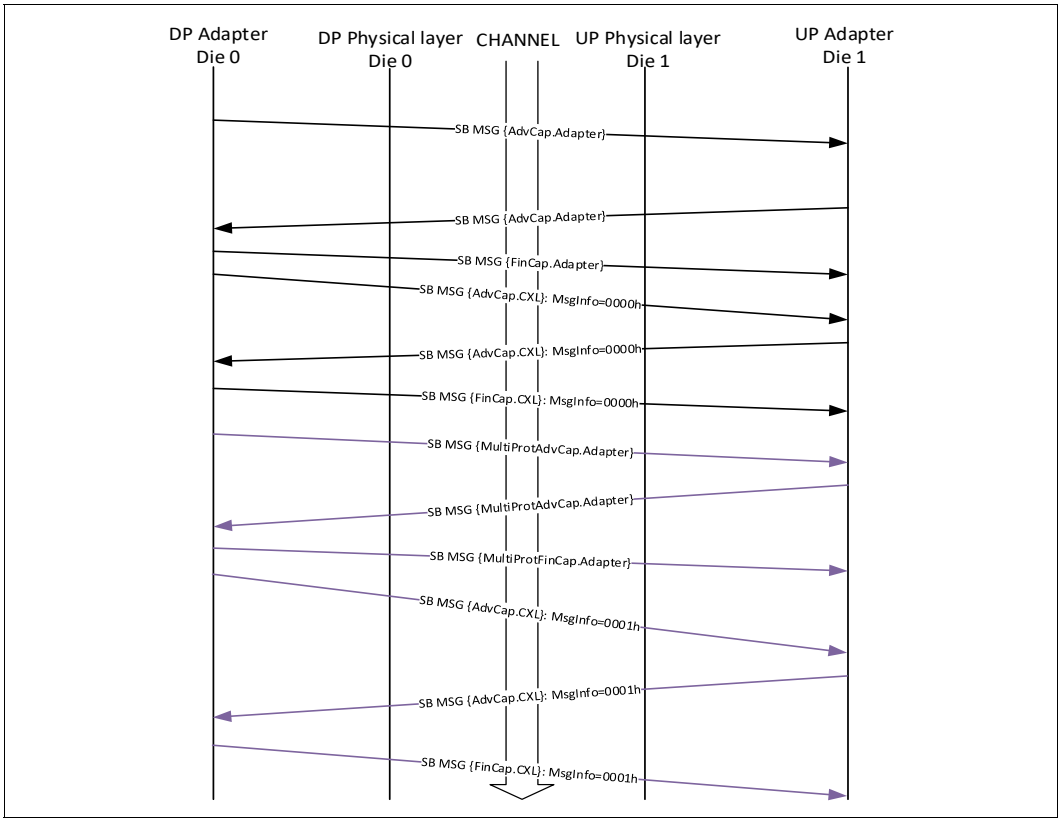


Figure 3-6. Stack 0 is PCIe, Stack 1 is Streaming

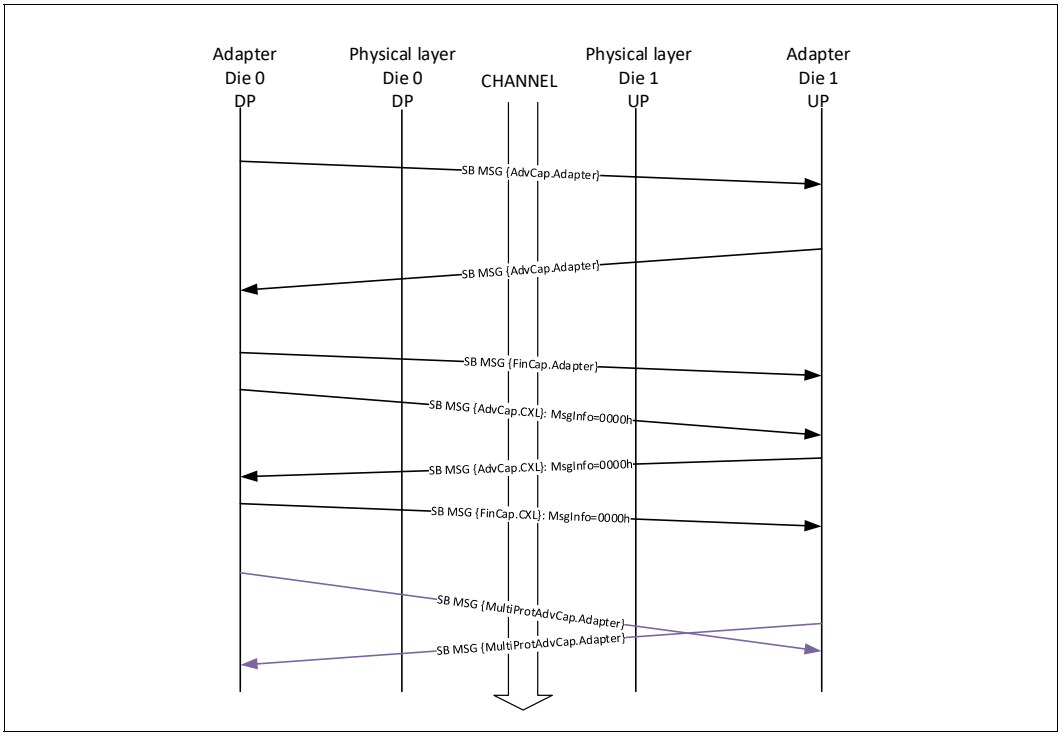


Figure 3-7. Stack 0 is Streaming, Stack 1 is PCIe

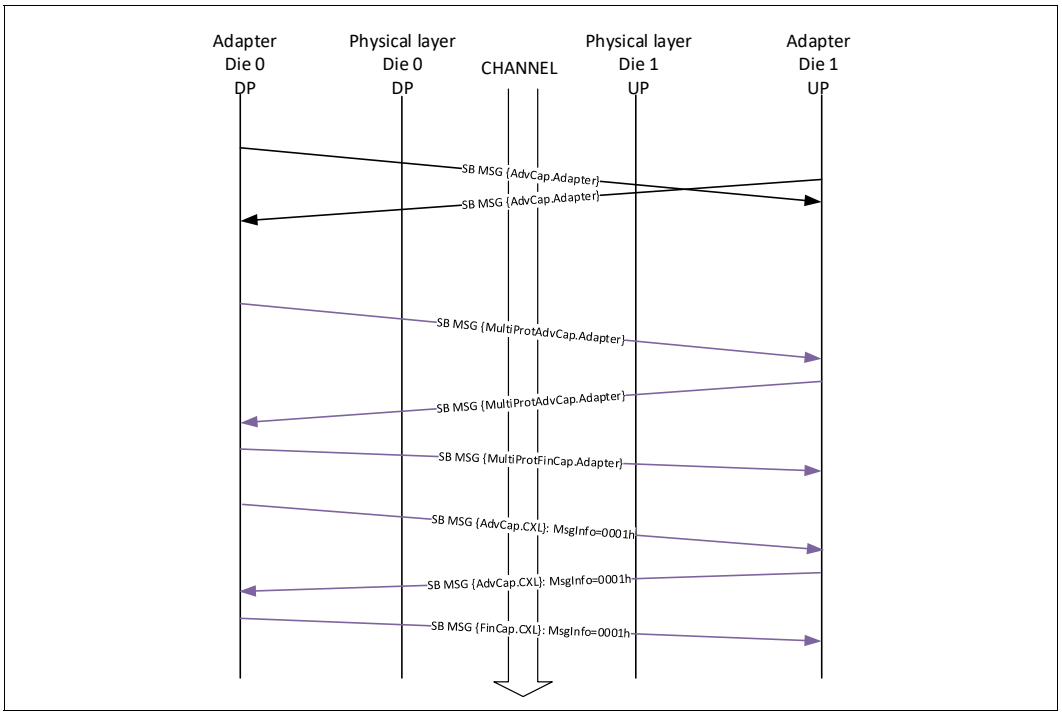


Figure 3-8. Both Stacks are Streaming

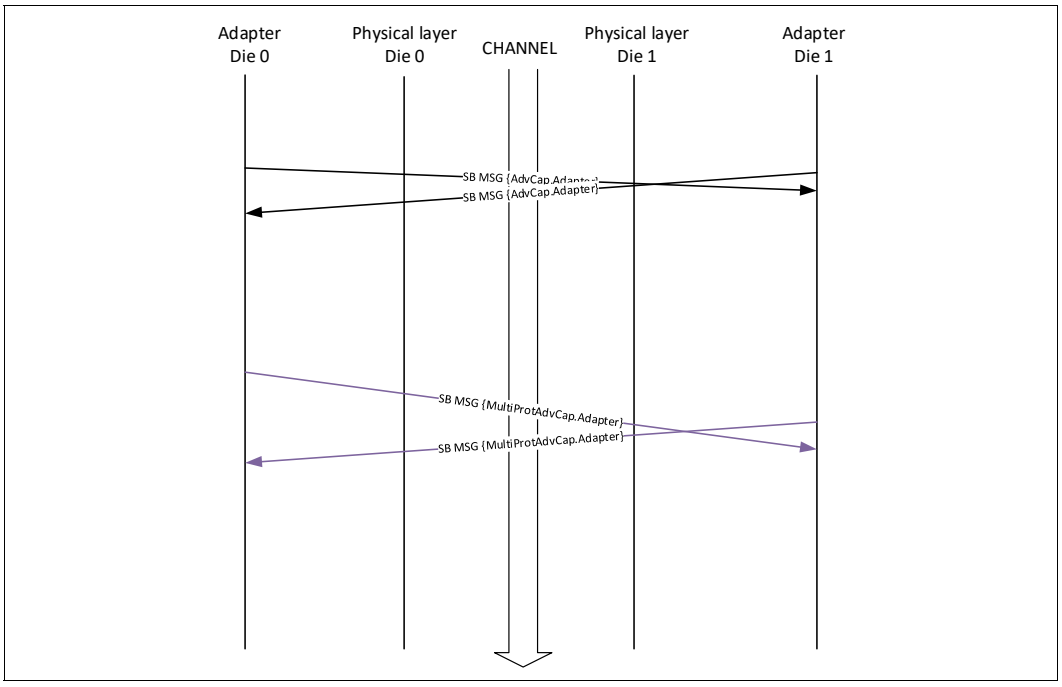
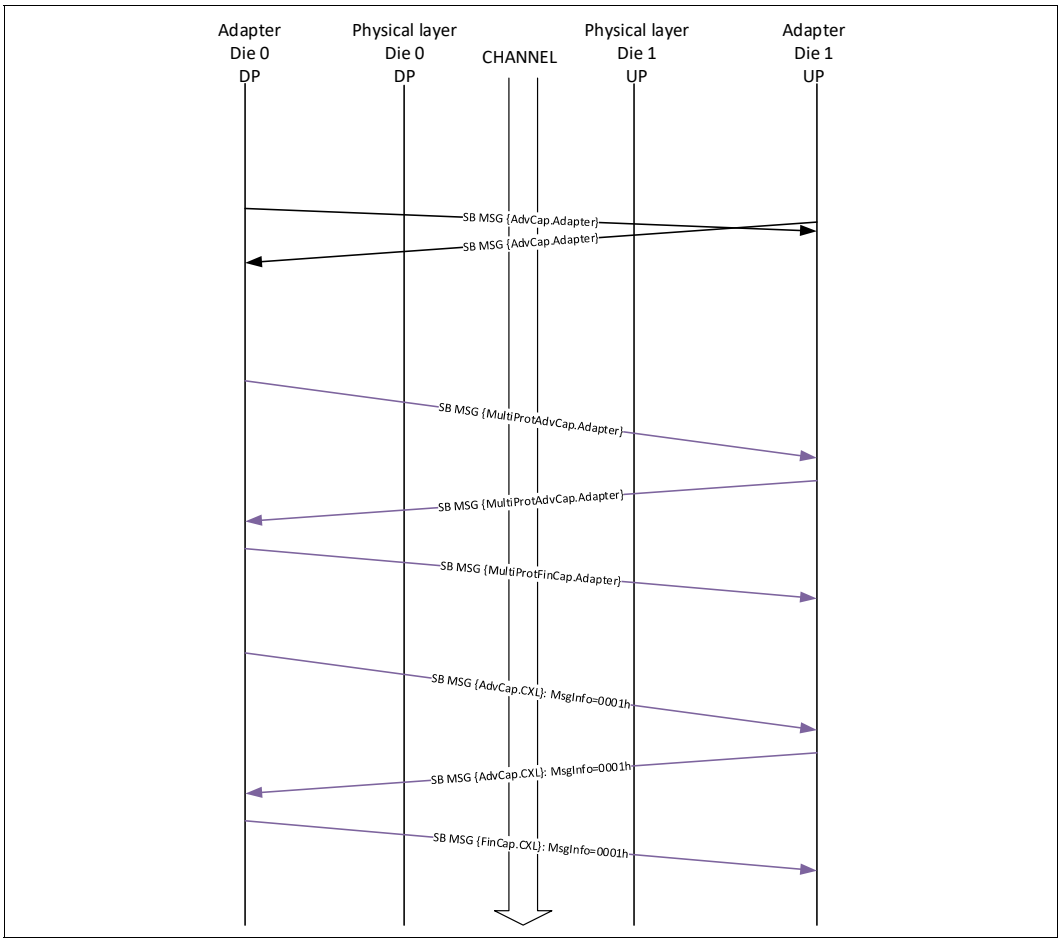


Figure 3-9. Stack 0 is Streaming, Stack 1 is CXL



The Adapter must implement a timeout of 8 ms (-0%/+50%) for successful Parameter Exchange completion. For the purposes of measuring a timeout for Parameter Exchange completion, all steps in Part 1 and Part 2 of Stage 3 of Link Initialization are included. The timer only increments while RDI is in Active state. The timer must reset if the Adapter receives an {AdvCap.*.Stall}, {FinCap.*.Stall}, {MultiProtAdvCap.*.Stall}, or {MultiProtFinCap.*.Stall} message from the remote Link partner. The 8-ms timeouts for Parameter Exchanges or Link State Machine transitions are treated as UIE and the Adapter must take the RDI to LinkError state. UCIe Retimers must ensure that they resolve the capability advertisement with remote Retimer partner (and merge with their own capabilities) before responding/initiating parameter exchanges with the UCIe die within its package. While resolution is in progress, they must send the corresponding stall message once every 4 ms to ensure that a timeout does not occur on the UCIe die within its package.

3.2.1.3 Part 3: FDI bring up

Once Parameter Exchanges have successfully completed, the Adapter reflects the result to the Protocol Layers on FDI, and moves on to carry out the FDI bring up flow as defined in Section 10.2.8. Once FDI is in Active state, it concludes Stage 3 of Link Initialization and protocol Flit transfer can begin. When multiple stacks are enabled on the same Adapter, each stack may finish the FDI bring up flow (see Section 10.2.8) at different times.

The data width on FDI is a function of the frequency of operation of the UCIe stack as well as the total bandwidth being transferred across the UCIe physical Link (which in turn depends on the number of Lanes and the speed at which the Lanes are operating). The data width on RDI is fixed to at least one byte per physical Lane per module that is controlled by the Adapter. The illustrations of the formats in this chapter are showing an example configuration of RDI mapped to a 64 Lane module of Advanced Package configuration on the Physical Layer of UCIe.

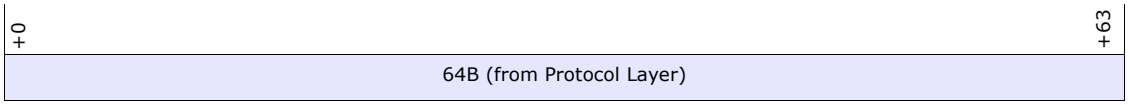
3.3 Operation Formats

In subsequent sections, when referring to CRC computation, a byte mapping of the Flit to CRC message (CRC message is the 128B input to CRC computation logic) is provided. See Section 3.7 for more details.

3.3.1 Raw Format for All Protocols

Raw Format can only be used for scenarios in which Retry support from the Adapter is not required. If Raw Format is negotiated for CXL or PCIe protocols, the Adapter transfers data from Protocol Layer to Physical Layer without any modification. Figure 3-10 shows an example of this for a 64B data path on FDI and RDI. This is identified as *Format 1* during parameter negotiation.

Figure 3-10. Format 1: Raw Format[a]



a. See Figure 2-1 for color mapping.

3.3.2 68B Flit Format

This Flit Format is identified as *Format 2* on UCIe. Support for this is mandatory when CXL 68B Flit Mode protocol or PCIe Non-Flit Mode protocol is supported. 68B Flit Format support is optional for Streaming protocols.

The Protocol Layer sends 64B of protocol information. The Adapter adds a two byte prefix of Flit Header and a two byte suffix of CRC. Table 3-3 gives the Flit Header format for *Format 2* when Retry from the Adapter is required. If Retry from the Adapter is not required, then the Flit Header format is as provided in Table 3-2.

Even if Retry is not required, the Adapter still computes and drives CRC bytes — the Receiver is strongly recommended to treat a CRC error as an Uncorrectable Internal Error in this situation. For CRC computation, Flit Byte 0 (i.e., Flit Header Byte 0) is assigned to CRC message Byte 0, Flit Byte 1 (i.e., Flit Header Byte 1) is assigned to CRC message Byte 1 and so on until Flit Byte 65 is assigned to CRC message Byte 65.

Retry is performed over this 68B Flit.

Table 3-2. Flit Header for Format 2 without Retry

Byte	Bit	Description	
		PCIe or CXL	Streaming Protocol
Byte 0	[7:6]	Protocol Identifier: 2'b00 : D2D Adapter NOP Flit or PDS Flit Header 2'b01 : CXL.io Flit 2'b10 : CXL.cachemem Flit 2'b11 : ARB/MUX Flit (Reserved encoding for PCIe)	Protocol Identifier: 2'b00 : D2D Adapter NOP Flit 2'b01 : Protocol Layer Flit Remaining encodings are Reserved.
	[5]	Stack Identifier: 1'b0 : Stack 0 1'b1 : Stack 1	
	[4]	1'b0 : Regular Flit Header 1'b1 : Pause of Data Stream (PDS) Flit Header	
	[3:0]	Reserved	
Byte 1[a]	[7]	1'b0 : Regular Flit Header 1'b1 : Pause of Data Stream (PDS) Flit Header	
	[6:0]	Reserved	

a. For a Test Flit, bits [7:6] of Byte 1 are 01b. See Section 11.2 for more details.

Table 3-3. Flit Header for Format 2 with Retry

Byte	Bit	Description	
		PCIe or CXL	Streaming Protocol
Byte 0	[7:6]	Protocol Identifier: 2'b00 : D2D Adapter NOP Flit or PDS Flit Header 2'b01 : CXL.io Flit 2'b10 : CXL.cachemem Flit 2'b11 : ARB/MUX Flit (Reserved encoding for PCIe)	Protocol Identifier: 2'b00 : D2D Adapter NOP Flit 2'b01 : Protocol Layer Flit Remaining encodings are Reserved.
	[5]	Stack Identifier: 1'b0 : Stack 0 1'b1 : Stack 1	
	[4]	1'b0 : Regular Flit Header 1'b1 : Pause of Data Stream (PDS) Flit Header	
	[3:0]	The upper four bits of Sequence number "S" (i.e., S[7:4])	
Byte 1[a]	[7:6]	2'b00 : Regular Flit Header 2'b11 : Pause of Data Stream (PDS) Flit Header Other encodings are reserved	
	[5:4]	Ack or Nak information 2'b00 : Explicit Sequence number "S" of Flit if not PDS, otherwise the bitwise inverted value of "NEXT_TX_FLIT_SEQ_NUM - 1". (See *PCIe Base Specification* for the definition of NEXT_TX_FLIT_SEQ_NUM and the subtraction operation for sequence numbers) 2'b01 : Ack. The Sequence number "S" carries the Ack'ed sequence number. 2'b10 : Nak. The Sequence number "S" carries 255 if N=1, otherwise it carries N-1, where N is the Nak'ed sequence number. 2'b11 : Reserved	
	[3:0]	The lower four bits of Sequence number "S" (i.e., S[3:0]) Sequence number 0 is reserved and if present, it implies no Ack or Nak is sent.	

a. For a Test Flit, bits [7:6] of Byte 1 are 01b. See Section 11.2 for more details.

3.3.2.1 68B Flit Format Alignment and Padding Rules

Because of the four bytes added by D2D Adapter, the alignment of the Flit does not always match the number of Lanes of the physical Link. The bytes added by D2D Adapter require the Adapter to shift the data arriving over FDI by four bytes for consecutive Flits transmitted over RDI. Data is always transferred in multiples of 256B (note that Retimer credits have a 256B data granularity). A mechanism to Pause the Data Stream is provided as a way to save power when the Link is idle. Before pausing the data stream, the data stream is terminated with a Pause of Data Stream (PDS) Flit Header followed by 0b padding to the next 64B count multiple boundary and at least two subsequent 64B chunks of all 0 value data. If the transfer is not at a 256B count multiple boundary, additional 64B chunks of all 0 value data are required to bring the transferred bytes to a 256B count multiple. The subsequent transfers of all 0 data mentioned above give the Receiver at least two 64B chunks to reset the receiving byte shifter. The PDS Flit Header and the 0 padding bytes following it must not be forwarded to the Protocol Layer. The PDS token is a variable-size Flit that carries a 2B special Flit Header (referred to as the PDS Flit Header), and 0 bytes padded as described above. The Transmitter of PDS drives the following on the Flit header:

1. Bit [4] of Byte 0 as 1
2. Bit [7] of Byte 1 as 1
3. Bit [6] of Byte 1 as 1
4. Bits [5:4] of Byte 1 as 00b and the sequence number[7:0] is matching the expected value for a PDS Flit Header in this position as defined in Table 3-3.

The Adapter may optionally insert continuous NOPs instead of terminating the data stream with a PDS when no other flits are available to transmit. There is a trade-off between the longer idle latency for a new flit to be transmitted after a PDS vs. the power consumption of continuously transmitting NOP flits. It is the responsibility of the transmitting Adapter to make the determination between transmitting NOP flits vs. inserting a PDS in an implementation-specific manner.

If Retry is enabled, the Receiver must interpret this Flit header as PDS if any two of the above four conditions are true. If Retry is disabled, the Receiver must interpret this Flit header as a PDS if conditions (1) and (2) are true.

A PDS must be inserted when Retry is triggered or RDI state goes through Retrain. The transmitter must insert PDS Flit Header and corresponding padding of 0s as it would for an actual PDS and start the replayed Flit from fresh alignment (i.e., flit begins from a 256B-aligned boundary). Note that for Retry, this should occur before the Transmitter begins replaying the Flits from the Retry buffer; and for Retrain entry, this should occur before asserting **`lp_stallack`** to the Physical Layer.

For Retry and Retrain scenarios, the Receiver must also look for the expected sequence number in Byte 0 and Byte 1 of the received data bus with a corresponding valid Flit (i.e., CRC passes). Note that for a Retrain scenario, a PDS might not be received at the receiver before the RDI state changes to Retrain, and the Adapter must discard any partially received 68B Flits after state change.

When resuming the data stream after a PDS token (i.e., a PDS Flit Header and the corresponding padding of 0s), the first Flit is always 256B aligned; any valid Flit transfer after a PDS token will resume the data stream. After a PDS Flit Header has been transmitted, the corresponding padding of 0b to satisfy the PDS token padding requirements must be finished before resuming the data stream with new Flits.

IMPLEMENTATION NOTE — BIT ERRORS AND ALIASING

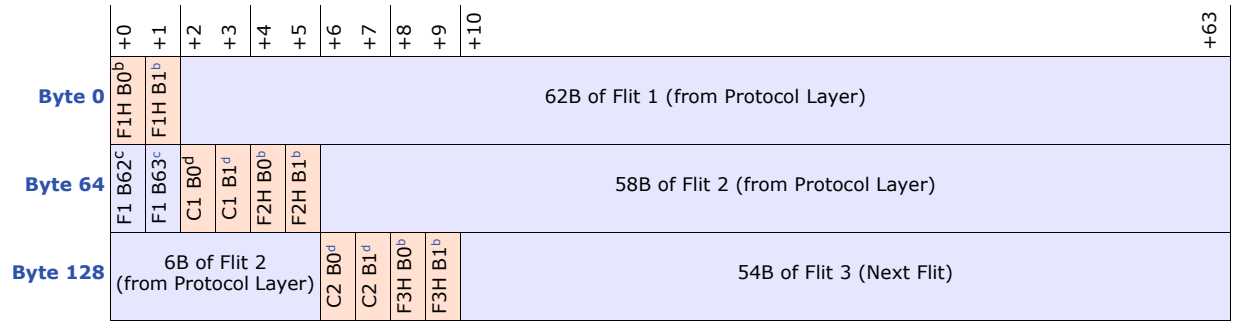
When Retry is disabled, the BER of the Link is 1E-27 or lower. In these cases, any bit error is an uncorrectable error for the Link. As a best practice, it is strongly recommended for receiver implementations to have an uncorrectable internal error condition for scenarios in which neither a valid Flit Header nor a valid PDS Flit Header is detected.

When Retry is enabled, the BER is 1E-15 or lower, which results in the probability of two or more bit errors within the Flit Header is very low. However, implementations must consider the following two scenarios:

- **PDS Flit Header aliasing to a regular Flit Header**: Checking for two out of the four conditions guarantees that at least three bit errors must occur within the two bytes of the PDS Flit Header for it to alias to a regular Flit Header. Even for three bit errors, there will be a CRC which will result in a retry and will be handled seamlessly through the retry rules.
- **Regular Flit Header aliasing to a PDS Flit Header**: It is possible for two bit errors to cause a Regular Flit Header to alias to a PDS Flit Header. This will likely result in a CRC error for future Flits. However, to reduce the probability of a data corruption that escapes CRC even further, it is strongly recommended that if a PDS Flit Header was detected without all four conditions being satisfied (i.e., two out of four or three out of four were satisfied), the receiver checks for an explicit sequence number Flit with the expected sequence number in Byte 0 and Byte 1 of the first received data transfer and that it is a valid Flit (i.e., CRC passes) after the PDS (including the PDS token and the corresponding padding) have completed; and triggers a Retry if it does not pass the check. Note that this is the same check a Receiver performs after a Retry or Retrain.

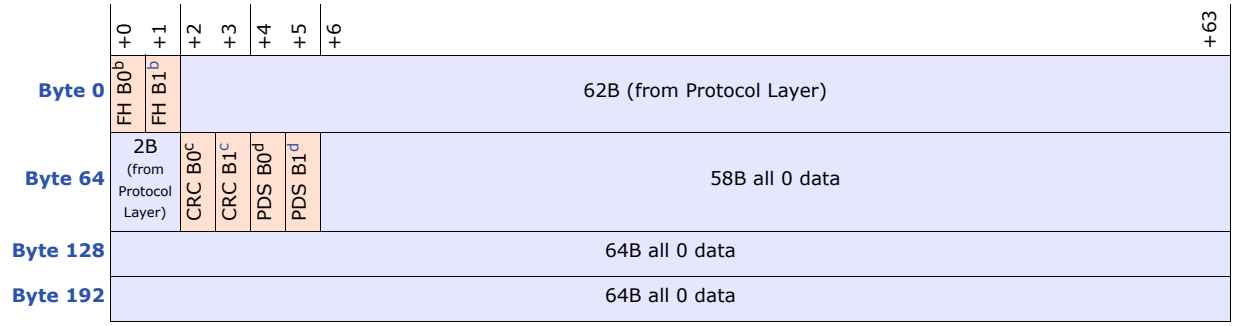
Figure 3-11 shows the 68B Flit Format. Figure 3-12 and Figure 3-13 provide examples of PDS insertion.

Figure 3-11. Format 2: 68B Flit Format[a]



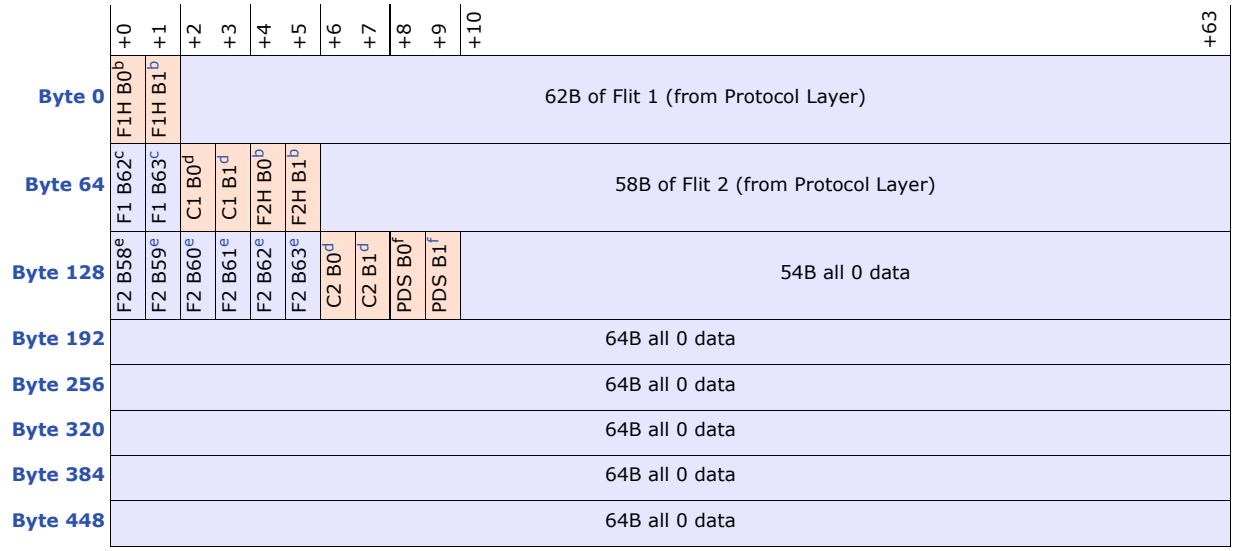
a. See Figure 2-1 for color mapping.
b. Flit 1 Header Byte 0, Flit 1 Header Byte 1, Flit 2 Header Byte 0, Flit 2 Header Byte 1, Flit 3 Header Byte 0, and Flit 3 Header Byte 1 respectively.
c. Flit 1 Byte 62 and Byte 63, respectively (from Protocol Layer).
d. Flit 1 CRC Byte 0, Flit 1 CRC Byte 1, Flit 2 CRC Byte 0, and Flit 2 CRC Byte 1, respectively.

Figure 3-12. Format 2: 68B Flit Format PDS Example 1[a]



a. See Figure 2-1 for color mapping.
b. Flit Header Byte 0 and Byte 1, respectively.
c. CRC Byte 0 and Byte 1, respectively.
d. PDS Flit Header Byte 0 and Byte 1, respectively.

Figure 3-13. Format 2: 68B Flit Format PDS Example 2 — Extra 0s Padded to Make the Data Transfer a Multiple of 256B[a]



a. See Figure 2-1 for color mapping.
b. Flit 1 Header Byte 0, Flit 1 Header Byte 1, Flit 2 Header Byte 0, and Flit 2 Header Byte 1, respectively.
c. Flit 1 Byte 62 and Byte 63, respectively (from Protocol Layer).
d. Flit 1 CRC Byte 0, Flit 1 CRC Byte 1, Flit 2 CRC Byte 0, and Flit 2 CRC Byte 1, respectively.
e. Flit 2 Bytes 58 through Byte 63, respectively (from Protocol Layer).
f. PDS Flit Header Byte 0 and Byte, respectively.

3.3.3 Standard 256B Flit Formats

These are the Standard Flit Formats defined in *PCIe Base Specification* for PCIe Flit Mode and *CXL Specification* for CXL 256B Flit Mode. These are identified as "Standard 256B End Header Flit Format" (or *Format 3*) and "Standard 256B Start Header Flit Format" (or *Format 4*), respectively. Support for this is mandatory when PCIe Flit Mode or CXL 256B Flit Mode protocols are negotiated. Standard 256B Flit Formats (Start Header or End Header) support is optional with Streaming protocols.

The Protocol Layer sends data in 256B Flits, but it drives 0 on the bytes reserved for the Adapter (shown in light orange in Figure 3-14 through Figure 3-19). The 6B of DLP defined in *PCIe Base Specification* exist in *Format 3* and *Format 4* as well for PCIe and CXL.io protocols. However, since

DLLPs are required to bypass the Tx Retry buffer in PCIe and CXL.io protocols, the DLP bytes end up being unique since they are partially filled by the Protocol Layer and partially by the Adapter. DLP0 and DLP1 are replaced with the Flit Header for UCIe and are driven by UCIe Adapter. However, if the Flit carries a Flit Marker, the Protocol Layer must populate bit 4 of Flit Header Byte 0 to 1b, as well as the relevant information in the Flit_Marker bits (these are driven as defined in *PCIe Base Specification*). Protocol Layer must also populate the Protocol Identifier bits in the Flit Header for the Flits it generates.

For Streaming protocols, Figure 3-17 shows the applicable Flit Format. Protocol Layer only populates bits [7:6] of Byte 0 of the Flit Header, and it must never set 00b for bits [7:6].

Standard 256B Start Header Flit Format is optional for PCIe Flit Mode protocol. Figure 3-18 shows the Flit Format example.

FDI provides a separate interface for DLLP transfer from the Protocol Layer to the Adapter and vice-versa. The Adapter is responsible for inserting DLLP into DLP Bytes 2:5 if a Flit Marker is not present. The credit update information is transferred as regular Update_FC DLLPs over FDI from the Protocol Layer to the Adapter. The Adapter is also responsible for formatting these updates as Optimized_Update_FC format when possible and driving them on the relevant DLP bytes. The Adapter is also responsible for adhering to all the DLLP rules defined for Flit Mode in *PCIe Base Specification*. On the receive path, the Adapter is responsible for extracting the DLLPs or Optimized_Update_FC from the Flit and driving it on the dedicated DLLP interface provided on FDI.

Two sets of CRC are computed (CRC0 and CRC1). The same 2B over 128B CRC computation as previous formats is used.

For PCIe, CXL, and Streaming:

- For *Format 3*, CRC0 is computed using Flit Bytes 0 to 127 assigned to the corresponding bytes of the CRC message input. CRC1 is computed using Flit Bytes 128 to 241 as the message input with Flit Byte 128 assigned to CRC message Byte 0, Flit Byte 129 assigned to CRC message Byte 1 and so on until Flit Byte 241 is assigned to CRC message Byte 113 (including the Flit Header bits inserted by the Adapter, which for PCIe and CXL.io, includes the DLP bytes inserted by the Adapter).
- For *Format 4*, CRC0 is computed using Flit Bytes 0 to 127 assigned to the corresponding bytes of the CRC message input (including the Flit Header bits inserted by the Adapter). CRC1 is computed using Flit Bytes 128 to 241 as the message input with Flit Byte 128 assigned to CRC message Byte 0, Flit Byte 129 assigned to CRC message Byte 1 and so on until Flit Byte 241 is assigned to CRC message Byte 113 (for PCIe and CXL.io, this includes the DLP bytes inserted by the Adapter).

If Retry is not required, the Adapter still computes and drives CRC bytes — the Receiver is strongly recommended to treat a CRC error as an Uncorrectable Internal Error (UIE) in this situation.

The Flit Header byte formats are shown in Table 3-5 when Retry is required; otherwise, it is as shown in Table 3-4.

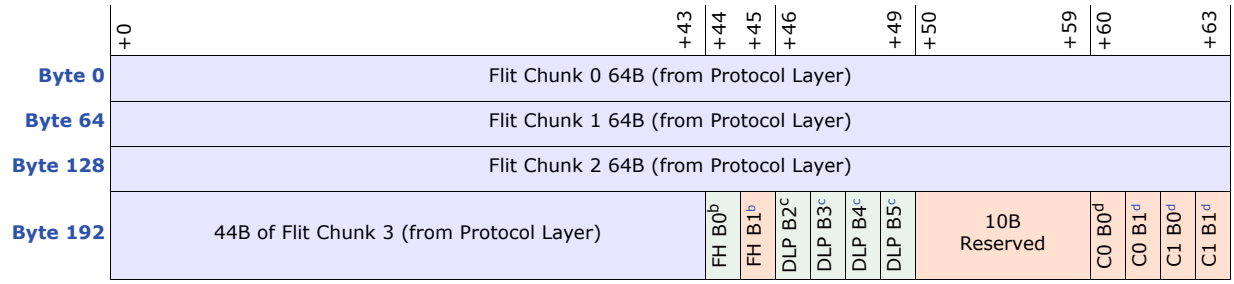
The Protocol Layer must drive bits [7:6] in Byte 1 of Flit Header to 00b for CXL/PCIe/Streaming protocol Flits and to 10b for Management Flits (when successfully negotiated).

For Management Flits, Bytes 238 to 241 are driven from the Protocol Layer with Management Transport Credit Return DWORD (CRD) Bytes 0 to 3 (see Section 8.2.5.2.2 for CRD format). Bytes 232 to 235 in *Format 3* and Bytes 234 to 237 in *Format 4* are driven from the Protocol Layer with 0s for Management Flits. See Figure 3-16 and Figure 3-19 for details of *Format 3* and *Format 4* for Management Flits, respectively.

If PCIe/CXL.io is negotiated along with Management Transport protocol on the same stack:

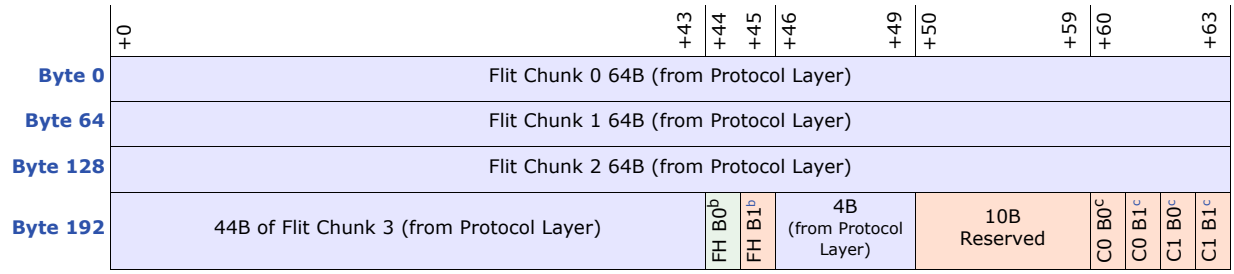
- If bits [7:6] of Byte 1 are 10b, the Adapter passes through Bytes 238 to 241 from the Protocol Layer to the Link
- If bits [7:6] of Byte 1 are 00b, Bytes 238 to 241 are treated per PCIe/CXL.io DLP rules for this flit format

Figure 3-14. Format 3: Standard 256B End Header Flit Format for PCIe[a]



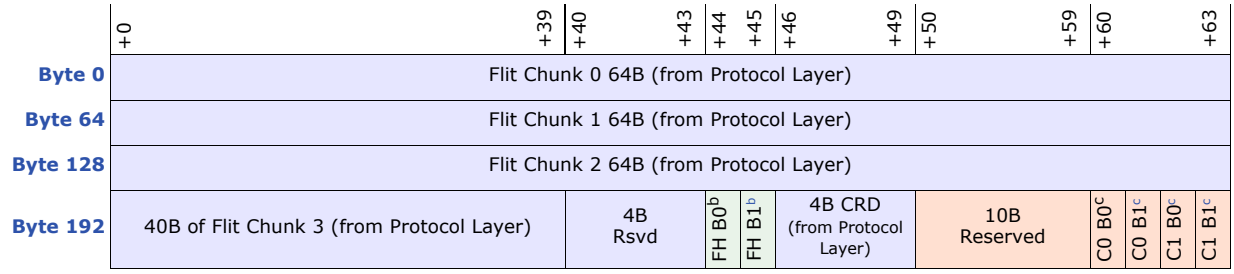
a. See Figure 2-1 for color mapping.
b. Flit Header Byte 0 and Byte 1, respectively.
c. DLP Byte 2, Byte 3, Byte 4, and Byte 5, respectively.
d. CRC0 Byte 0, CRC0 Byte 1, CRC1 Byte 0, and CRC1 Byte 1, respectively.

Figure 3-15. Format 3: Standard 256B End Header Flit Format for Streaming Protocol[a]



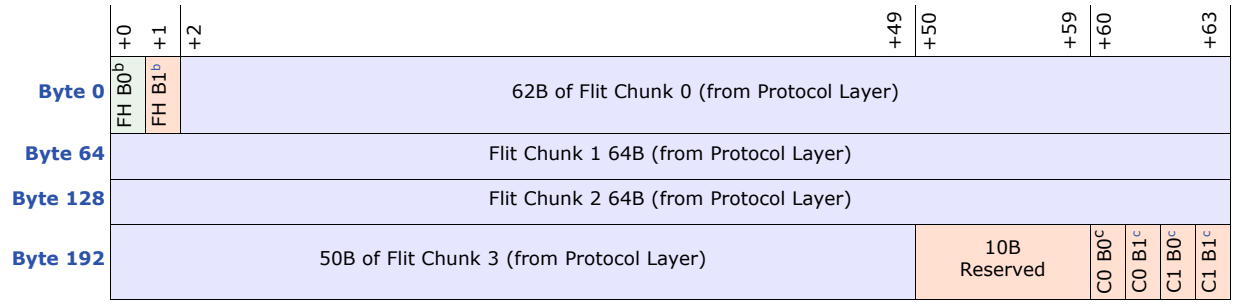
a. See Figure 2-1 for color mapping.
b. Flit Header Byte 0 and Byte 1, respectively.
c. CRC0 Byte 0, CRC0 Byte 1, CRC1 Byte 0, and CRC1 Byte 1, respectively.

Figure 3-16. Format 3: Standard 256B End Header Flit Format for Management Transport Protocol[a]



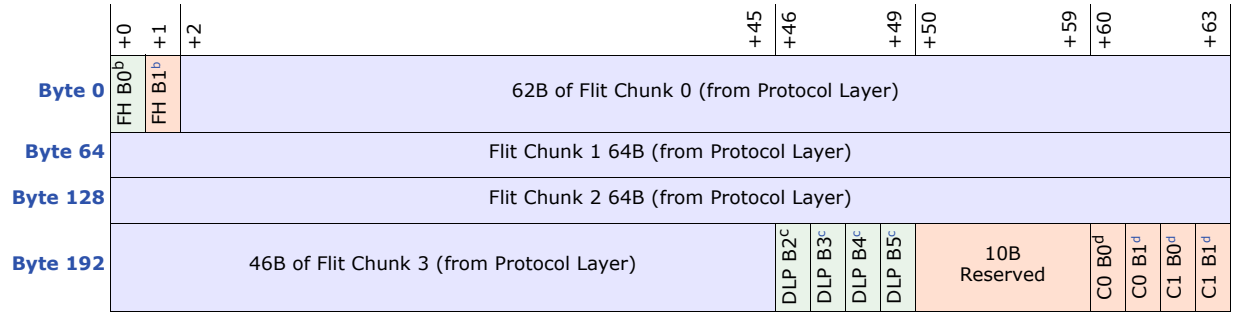
a. See Figure 2-1 for color mapping.
b. Flit Header Byte 0 and Byte 1, respectively.
c. CRC0 Byte 0, CRC0 Byte 1, CRC1 Byte 0, and CRC1 Byte 1, respectively.

Figure 3-17. Format 4: Standard 256B Start Header Flit Format for CXL.cachemem or Streaming Protocol[a]



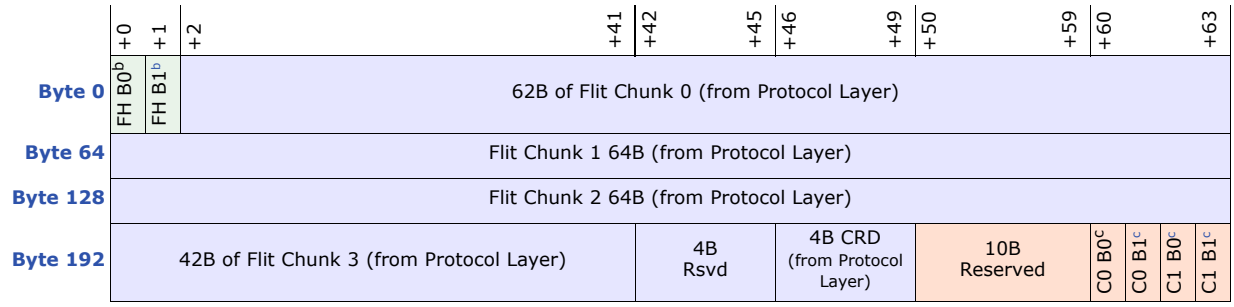
a. See Figure 2-1 for color mapping.
b. Flit Header Byte 0 and Byte 1, respectively.
c. CRC0 Byte 0, CRC0 Byte 1, CRC1 Byte 0, and CRC1 Byte 1, respectively.

Figure 3-18. Format 4: Standard 256B Start Header Flit Format for CXL.io or PCIe[a]



a. See Figure 2-1 for color mapping.
b. Flit Header Byte 0 and Byte 1, respectively.
c. DLP Byte 2, Byte 3, Byte 4, and Byte 5, respectively.
d. CRC0 Byte 0, CRC0 Byte 1, CRC1 Byte 0, and CRC1 Byte 1, respectively.

Figure 3-19. Format 4: Standard 256B Start Header Flit Format for Management Transport Protocol[a]



a. See Figure 2-1 for color mapping.
b. Flit Header Byte 0 and Byte 1, respectively.
c. CRC0 Byte 0, CRC0 Byte 1, CRC1 Byte 0, and CRC1 Byte 1, respectively.

Table 3-4. Flit Header for Format 3, Format 4, Format 5, and Format 6 without Retry

Byte	Bit	Description			
		CXL 256B Flit Mode	**PCIe Flit Mode**	**Streaming Protocol**	**Management Transport Protocol**
0	[7:6]	Protocol Identifier: 00b: D2D Adapter/CXL.io NOP Flit 01b: CXL.io Flit 10b: CXL.cachemem Flit 11b: ARB/MUX Flit	Protocol Identifier: 00b: D2D Adapter/PCIe NOP Flit 01b: PCIe Flit All other encodings are reserved.	Protocol Identifier: 00b: D2D Adapter NOP Flit Remaining encodings are permitted to be used by Protocol Layer in a vendor defined manner. Protocol Layer must never set this to 00b for Flits sent across FDI.	Protocol Identifier: 00b: D2D Adapter NOP Flit 01b: Management Flit All other encodings are reserved.
	[5]	Stack Identifier: 0: Stack 0 1: Stack 1			
	[4]	Reserved for CXL.cachemem For CXL.io or PCIe Flit Mode: 0: DLLP Payload in DLP 2..5 1: Optimized_Update_FC or Flit_Marker in DLP2..5		Reserved	
	[3:0]	Reserved			
1	[7:6]	Flit Type: 00b: CXL/PCIe/Streaming Flit/D2D Adapter NOP Flit 01b: Test Flit (see Section 11.2 for details) 10b: Management Flit 11b: Reserved			
	[5:0]	Reserved			

Table 3-5. Flit Header for Format 3, Format 4, Format 5, and Format 6 with Retry

Byte	Bit	Description			
		CXL 256B Flit Mode	**PCIe Flit Mode**	**Streaming Protocol**	**Management Transport Protocol**
0	[7:6]	Protocol Identifier: 00b: D2D Adapter/CXL.io NOP Flit 01b: CXL.io Flit 10b: CXL.cachemem Flit 11b: ARB/MUX Flit	Protocol Identifier: 00b: D2D Adapter/PCIe NOP Flit 01b: PCIe Flit All other encodings are reserved	Protocol Identifier: 00b: D2D Adapter NOP Flit Remaining encodings are permitted to be used by Protocol Layer in a vendor defined manner. Protocol Layer must never set this to 00b for Flits sent across FDI.	Protocol Identifier: 00b: D2D Adapter NOP Flit 01b: Management Flit All other encodings are reserved.
	[5]	Stack Identifier: 0: Stack 0 1: Stack 1			
	[4]	Reserved for CXL.cachemem For CXL.io or PCIe Flit Mode: 0: DLLP Payload in DLP 2..5 1: Optimized_Update_FC or Flit_Marker in DLP2..5		Reserved	
	[3:0]	The upper four bits of Sequence number "S" (i.e., S[7:4])			
1	[7:6]	Flit Type: 00b: CXL/PCIe/Streaming Flit/D2D Adapter NOP Flit 01b: Test Flit (see Section 11.2 for details) 10b: Management Flit 11b: Reserved			
	[5:4]	Ack or Nak Information: 00b: Explicit Sequence number "S" of the current Flit is present. 01b: Ack. The sequence number "S" carries the Ack'ed sequence number. 10b: Nak. The sequence number "S" carries 255 if N=1; otherwise, it carries N-1; where N is the Nak'ed sequence number. 11b: Reserved			
	[3:0]	The lower four bits of Sequence number "S" (i.e., S[3:0]). Sequence number 0 is reserved and if present, it implies no Ack or Nak is sent.			

3.3.4 Latency-Optimized 256B Flit Formats

Two Latency-Optimized 256B Flit Formats are defined: *Format 5* and *Format 6*. It is strongly recommended that UCIe implementations support *Format 6* for CXL 256B Flit Mode protocol to get the best latency benefits.

Both formats look the same from the Adapter perspective, the only difference is whether the Protocol Layer is filling in the optional bytes of protocol information . The Latency-Optimized 256B without Optional bytes Flit Format (or *Format 5*) is when the Protocol Layer is not filling in the optional bytes, whereas the Latency-Optimized 256B with Optional bytes Flit Format (or *Format 6*) is when the Protocol Layer is filling in the optional bytes.

Latency-Optimized 256B Flit Formats (with Optional bytes or without Optional bytes) support is optional with Streaming protocols. Protocol Layer only populates bits [7:6] of the Flit Header, and it must never set 00b for bits [7:6].

Latency-Optimized Flit with Optional Bytes Flit Format is optional for PCIe Flit Mode protocol. Figure 3-23 shows the Flit Format example.

Two sets of CRC are computed. CRC0 is computed using Flit Bytes 0 to 125 assigned to the corresponding bytes of the CRC message input (including the Flit Header bits and if applicable, the DLP bits inserted by the Adapter). CRC1 is computed using Flit Bytes 128 to 253 as the message input with Flit Byte 128 assigned to CRC message Byte 0, Flit Byte 129 assigned to CRC message Byte 1 and so on until Flit Byte 253 assigned to CRC message Byte 125. If Retry is not required, the Adapter still computes and drives CRC bytes — the Receiver is strongly recommended to treat a CRC error as UIE in this situation.

For Management Flits (when successfully negotiated), the Protocol Layer must drive bits [7:6] in Byte 1 of Flit Header to 00b for Protocol Flit and to 10b.

For Management Flits using *Format 5*, Bytes 240 to 243 are driven from the Protocol Layer with Management Transport Credit Return DWORD (CRD) Bytes 0 to 3 (see Section 8.2.5.2.2 for CRD format). See Figure 3-22 for details.

If CXL.io is negotiated along with Management Transport protocol on the same stack for *Format 5*:

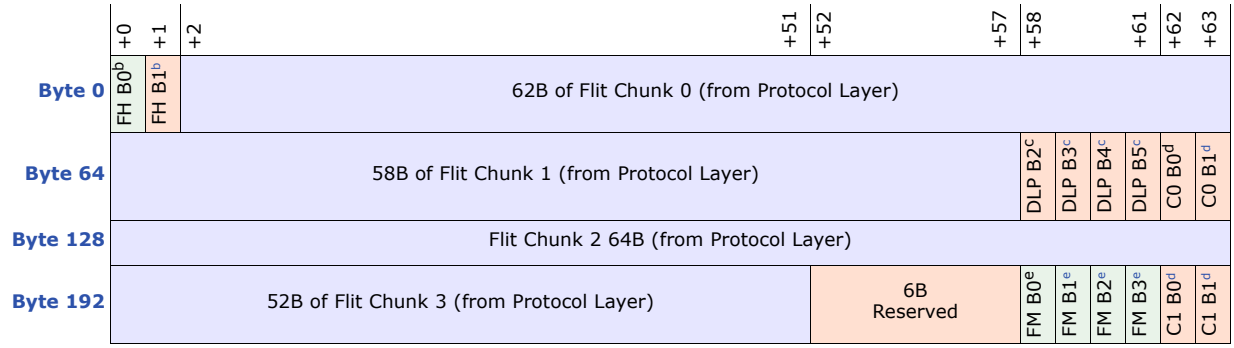
- If bits [7:6] of Byte 1 are 10b, the Adapter drives 0 on Bytes 122 to 125 and 244 to 253
- If bits [7:6] of Byte 1 are 00b, then Bytes 122 to 125 are treated per the CXL.io DLP rules of this flit format and Bytes 250 to 253 are treated per the CXL.io FM rules of this flit format

For Management Flits using *Format 6*, Bytes 250 to 253 are driven from the Protocol Layer with Management Transport Credit Return DWORD (CRD) Bytes 0 to 3 (see Section 8.2.5.2.2 for CRD format). Similarly, Bytes 244 to 249 are driven from the Protocol Layer as 0. See Figure 3-26 for details.

If PCIe/CXL.io is negotiated along with Management Transport protocol on the same stack for *Format 6*:

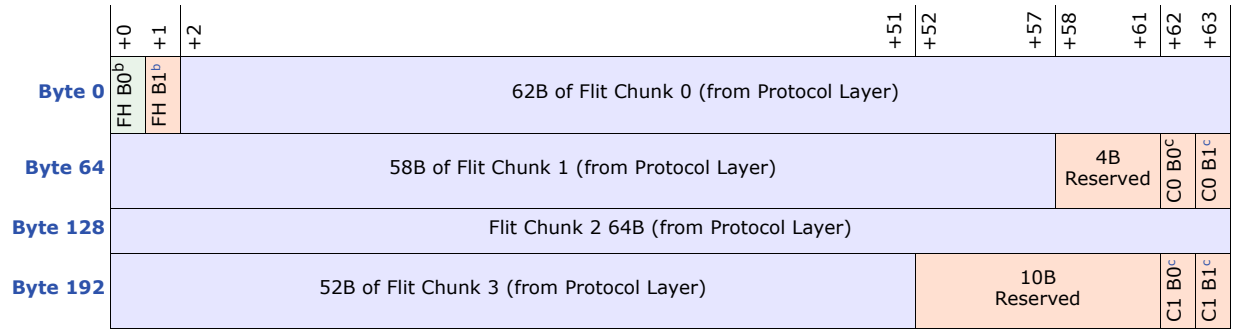
- If bits [7:6] of Byte 1 are 10b, the Adapter passes through Bytes 122 to 125 and 248 to 253
- If bits [7:6] of Byte 1 are 00b, then Bytes 122 to 125 are treated per the PCIe/CXL.io DLP rules of this flit format, Bytes 250 to 253 are treated per the PCIe/CXL.io FM rules of this flit format, and the Adapter drives 0 on Bytes 248 and 249

Figure 3-20. Format 5: Latency-Optimized 256B without Optional Bytes Flit Format for CXL.io[a]



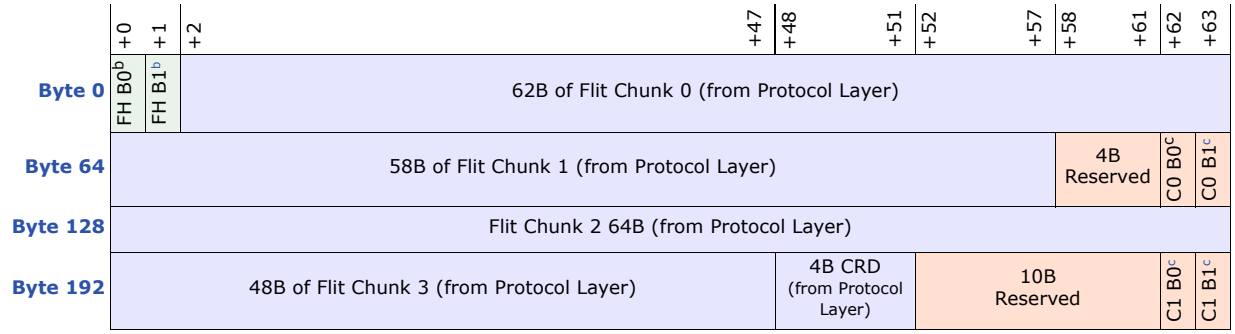
a. See Figure 2-1 for color mapping.
b. Flit Header Byte 0 and Byte 1, respectively.
c. DLP Byte 2, Byte 3, Byte 4, and Byte 5, respectively.
d. CRC0 Byte 0, CRC0 Byte 1, CRC1 Byte 0, and CRC1 Byte 1, respectively.
e. Flit_Marker or Optimized_Update_FC Byte 0, Byte 1, Byte 2, and Byte 3, respectively.

Figure 3-21. Format 5: Latency-Optimized 256B without Optional Bytes Flit Format for CXL.cachemem and Streaming Protocol[a]



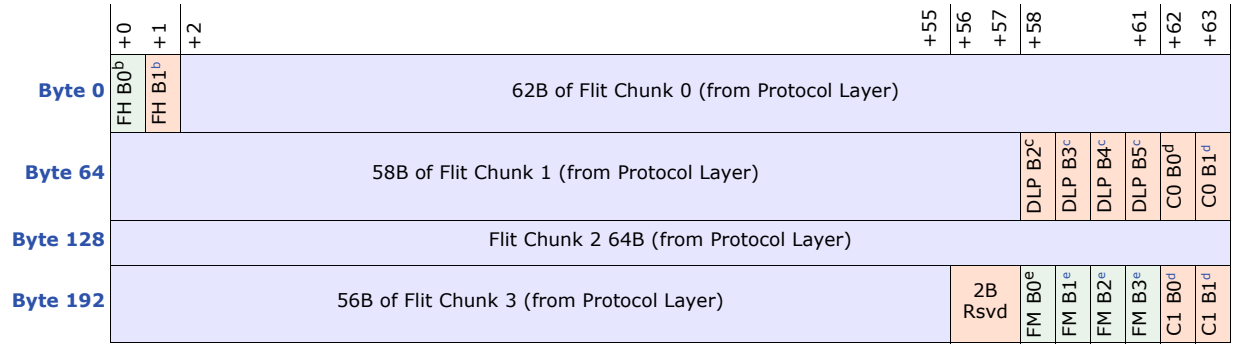
a. See Figure 2-1 for color mapping.
b. Flit Header Byte 0 and Byte 1, respectively.
c. CRC0 Byte 0, CRC0 Byte 1, CRC1 Byte 0, and CRC1 Byte 1, respectively.

Figure 3-22. Format 5: Latency-Optimized 256B without Optional Bytes Flit Format for Management Transport Protocol[a]



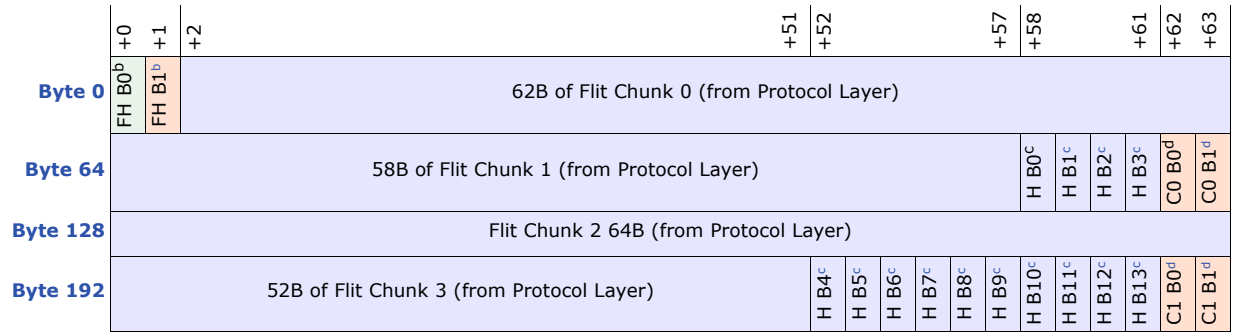
a. See Figure 2-1 for color mapping.
b. Flit Header Byte 0 and Byte 1, respectively.
c. CRC0 Byte 0, CRC0 Byte 1, CRC1 Byte 0, and CRC1 Byte 1, respectively.

Figure 3-23. Format 6: Latency-Optimized 256B with Optional Bytes Flit Format for CXL.io or PCIe[a]



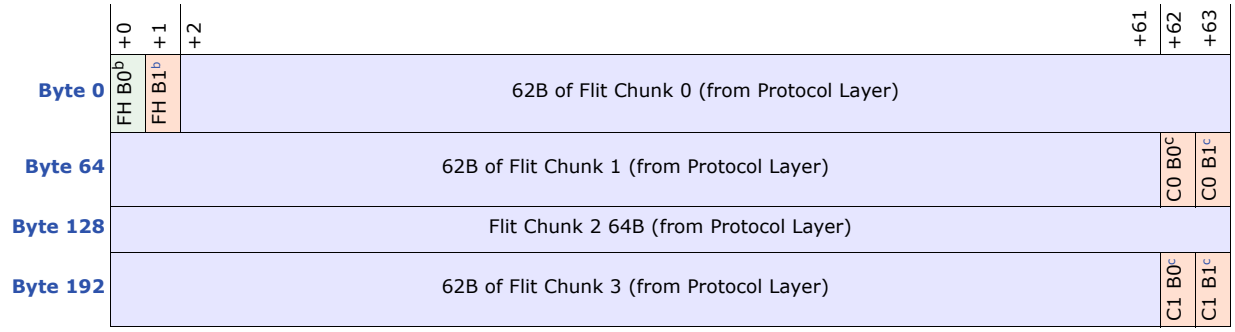
a. See Figure 2-1 for color mapping.
b. Flit Header Byte 0 and Byte 1, respectively.
c. DLP Byte 2, Byte 3, Byte 4, and Byte 5, respectively.
d. CRC0 Byte 0, CRC0 Byte 1, CRC1 Byte 0, and CRC1 Byte 1, respectively.
e. Flit_Marker Byte 0, Byte 1, Byte 2, and Byte 3, respectively.

Figure 3-24. Format 6: Latency-Optimized 256B with Optional Bytes Flit Format for CXL.cachemem[a]



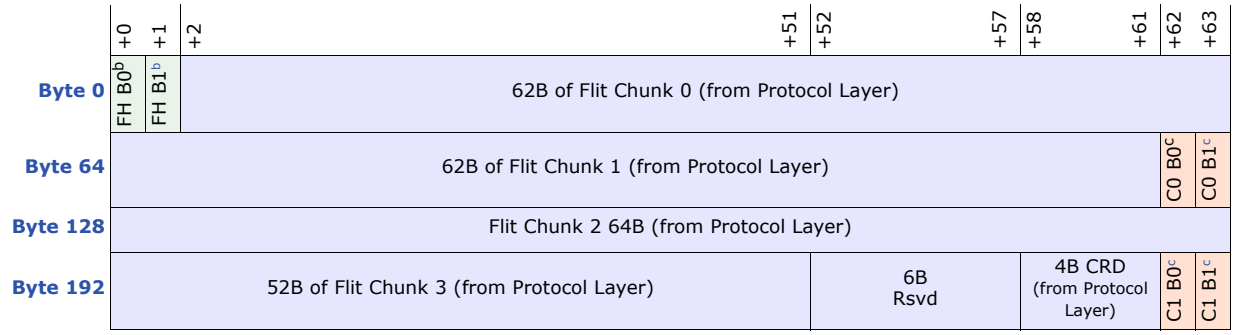
a. See Figure 2-1 for color mapping.
b. Flit Header Byte 0 and Byte 1, respectively.
c. H-slot Byte 0 through Byte 13, respectively (from Protocol Layer).
d. CRC0 Byte 0, CRC0 Byte 1, CRC1 Byte 0, and CRC1 Byte 1, respectively.

Figure 3-25. Format 6: Latency-Optimized 256B with Optional Bytes Flit Format for Streaming Protocol[a]



a. See Figure 2-1 for color mapping.
b. Flit Header Byte 0 and Byte 1, respectively.
c. CRC0 Byte 0, CRC0 Byte 1, CRC1 Byte 0, and CRC1 Byte 1, respectively.

Figure 3-26. Format 6: Latency-Optimized 256B with Optional Bytes Flit Format for Management Transport Protocol[a]



a. See Figure 2-1 for color mapping.
b. Flit Header Byte 0 and Byte 1, respectively.
c. CRC0 Byte 0, CRC0 Byte 1, CRC1 Byte 0, and CRC1 Byte 1, respectively.

The Flit Header byte formats are the same as Table 3-5 when Retry is required; otherwise, they are the same as Table 3-4. The DLP rules are also the same as defined in Section 3.3.3 for CXL protocol, except that Flit_Marker/Optimized_Update_FC has dedicated space in the Flit (i.e., bit [4] of Byte 0 corresponds to the Flit_Marker bytes, and not the DLP bytes). If Optimized_Update_FC is sent, the DLP Bytes 2:5 shown in Figure 3-20 must be reserved. If bit [4] of Byte 0 in the Flit Header is 0b, then the Flit_Marker bytes are reserved.

3.3.5 Flit Format-related Implementation Requirements for Protocol Layer and Adapter

Table 3-6 lists the different Flit Formats supported in UCIe.

Table 3-6. Summary of Flit Formats

Format Number	Name	Notes	For Details, See Also
Format 1	Raw	Protocol Layer populates all the bytes on FDI. Adapter passes to RDI without modifications or additions.	• Section 3.3.1 • Figure 3-10
Format 2	68B Flit	Protocol Layer transmits 64B per Flit on FDI. Adapter inserts two bytes of Flit header and two bytes of CRC and performs the required barrel shifting of bytes before transmitting on RDI. On the RX, Adapter strips out the Flit header and CRC only sending the 64B per Flit to the Protocol Layer on FDI.	• Section 3.3.2 • Figure 3-11 • Figure 3-12
Format 3	Standard 256B End Header Flit	Protocol Layer transmits 256B of Flit on FDI, while driving 0b on the bits reserved for the Adapter. Adapter fills in the relevant Flit header and CRC information before transmitting on RDI. On the Rx, Adapter forwards the Flit received from the Link to the Protocol Layer without modifying any bits applicable to the Protocol Layer, and the Protocol Layer must ignore any bits not applicable for it. Flit Header is located on Byte 236 and Byte 237 of the Flit.	• Section 3.3.3 • Figure 3-14 • Figure 3-15
Format 4	Standard 256B Start Header Flit	Protocol Layer transmits 256B of Flit on FDI, while driving 0b on the bits reserved for the Adapter. Adapter fills in the relevant Flit header and CRC information before transmitting on RDI. On the Rx, Adapter forwards the Flit received from the Link to the Protocol Layer without modifying any bits applicable to the Protocol Layer, and the Protocol Layer must ignore any bits not applicable for it. Flit Header is located on Byte 0 and Byte 1 of the Flit.	• Section 3.3.3 • Figure 3-17 • Figure 3-18
Format 5	Latency-Optimized 256B without Optional Bytes Flit	Protocol Layer transmits 256B of Flit on FDI, while driving 0b on the bits reserved for the Adapter. Adapter fills in the relevant Flit header and CRC information before transmitting on RDI. On the Rx, Adapter forwards the Flit received from the Link to the Protocol Layer without modifying any bits applicable to the Protocol Layer, and the Protocol Layer must ignore any bits not applicable for it. CRC bytes sent with each 128B of the Flit. The optional Protocol Layer bytes are reserved in this format and not used by the Protocol Layer.	• Section 3.3.4 • Figure 3-20 • Figure 3-21
Format 6	Latency-Optimized 256B with Optional Bytes Flit	Protocol Layer transmits 256B of Flit on FDI, while driving 0b on the bits reserved for the Adapter. Adapter fills in the relevant Flit header and CRC information before transmitting on RDI. On the Rx, Adapter forwards the Flit received from the Link to the Protocol Layer without modifying any bits applicable to the Protocol Layer, and the Protocol Layer must ignore any bits not applicable for it. CRC bytes sent with each 128B of the Flit, and optional bytes are used by the Protocol Layer.	• Section 3.3.4 • Figure 3-23 • Figure 3-24 • Figure 3-25

Table 3-7 gives the implementation requirements and Protocol Mapping for the different Flit Formats. For PCIe and CXL protocols, the implementation requirements must be followed by the Protocol Layer as well as the Adapter implementations. For Streaming protocols, the implementation requirements are for the Adapter only; Protocol Layer interoperability and implementation requirements are vendor specific.

Table 3-7. Protocol Mapping and Implementation Requirements

Format Number	Flit Format Name	PCIe Non-Flit Mode	PCIe Flit Mode	CXL 68B Flit Mode	CXL 256B Flit Mode	Streaming Protocol	Management Transport Protocol
1	Raw	Optional	Optional	Optional	Optional	Mandatory	Optional
2	68B	Mandatory	N/A	Mandatory	N/A	Optional[a]	N/A
3	Standard 256B End Header	N/A	Mandatory	N/A	N/A	Optional[a]	Optional
4	Standard 256B Start Header	N/A	Optional[b]	N/A	Mandatory	Optional[a]	Optional
5	Latency-Optimized 256B without Optional Bytes	N/A	N/A	N/A	Optional	Optional[a]	Optional
6	Latency-Optimized 256B with Optional Bytes	N/A	Strongly Recommended[c]	N/A	Strongly Recommended	Strongly Recommended[a]	Optional

a. If Streaming Flit Format capability is supported, else it is N/A.
b. If Standard Start Header for PCIe protocol capability is supported, else it is N/A.
c. If Latency-Optimized Flit with Optional Bytes for PCIe protocol capability is supported, else it is N/A.
If Enhanced Multi-Protocol capability is supported where at least one of the stacks supports PCIe, this format and the corresponding capability are strongly recommended.

3.4 Decision Table for Flit Format and Protocol

Table 3-8 shows the Truth Table for determining Protocol. Once the protocol and Flit Format have been negotiated during initial Link bring up, they cannot be changed until the UCIe Physical Layer transitions to Reset state.

If a valid Protocol and Flit Format are not negotiated, then the Adapter takes the Link down and reports the error if applicable.

Table 3-8. Truth Table for Determining Protocol[a]

{FinCap.Adapter} bits or {MultiProtFinCap.Adapter} bits[b]					{FinCap.CXL} bits		Protocol
68B Flit Mode	CXL 256B Flit Mode	PCIe Flit Mode	Streaming Protocol	Management Transport Protocol	PCIe	CXL.io	
1	0	0	x	x	0	1	CXL[c] without Management Transport protocol
1	1	1	x	0	0	1[d]	CXL[c] without Management Transport protocol
1	1	1	x	1	0	1[d]	CXL[c] with Management Transport protocol
1	0	0	x	x	1	0	PCIe[e] without Management Transport protocol
1	0	1	x	0	1[f]	0	PCIe without Management Transport protocol
1	0	1	x	1	1[f]	0	PCIe with Management Transport protocol
N/A	N/A	N/A	N/A	N/A	N/A	N/A	Streaming[g] with or without Management Transport protocol
N/A	N/A	N/A	N/A	N/A	N/A	N/A	Management Transport protocol[h]

a. x indicates don't care in this Table.
b. If Enhanced_Multi-Protocol capability is negotiated then {MultiProt*.Adapter} messages are used to determine the protocol for Stack 1. Stack 0 protocol is determined using the {FinCap.*} messages.
c. For CXL protocol, the specific combination of Single Protocol vs Type 1 vs. Type 2 vs. Type 3 is determined using the CXL.cache and CXL.mem capable/enable bits in addition to the CXL.io capable/enable bit in {FinCap.CXL}. The rules for that follow *CXL Specification*. When CXL is the protocol, if CXL 256B Flit mode is 1, then the protocol follows CXL 256B Flit mode rules; otherwise, the protocol follows CXL 68B Flit mode rules.
d. CXL.io capable/enable must be 1 if CXL 256B Flit mode is negotiated.
e. For PCIe protocol, if PCIe Flit mode is 1, then the protocol follows PCIe Flit mode rules; otherwise, the protocol follows PCIe Non-Flit mode rules.
f. PCIe capable/enable must be 1 if PCIe Flit mode is 1 but CXL 256B Flit mode is 0.
g. No {FinCap.*} message is sent for Streaming protocol negotiation, Streaming is the negotiated protocol if PCIe or CXL are not advertised, but Streaming protocol is advertised. If Management Transport protocol was also advertised along with Streaming protocol, then Management Transport protocol is enabled along with Streaming protocol.
h. No {FinCap.*} message is sent for Management Transport protocol negotiation, Management Transport is the negotiated protocol if PCIe or CXL or Streaming are not advertised, but Management Transport protocol is advertised.

IMPLEMENTATION NOTE

The "68B Flit Mode" parameter is advertised as set to 1 for both the CXL and PCIe protocols in {AdvCap.Adapter} sideband messages. As seen in Table 3-8, this parameter is set to 1 in {FinCap.Adapter} sideband messages whenever the CXL OR PCIe protocols are negotiated.

The "CXL.io" and "PCIe" bits in the {AdvCap.CXL} sideband message disambiguate between CXL support vs. PCIe support. It is permitted to set both to 1 in {AdvCap.CXL} sideband messages. However, as seen in Table 3-8, only one of these must be set in the {FinCap.CXL} sideband message to reflect the final negotiated protocol for the corresponding stack. For example:

- If the DP and UP both support CXL and PCIe protocols, then both "CXL.io" and "PCIe" will be set to 1 in the {AdvCap.CXL} sideband message
- If the DP decides to operate in CXL, the DP will set "CXL.io" to 1 and clear "PCIe" to 0 in the {FinCap.CXL} sideband message, in which case the remaining CXL-related bits in the {FinCap.CXL} sideband message are also applicable and are assigned as per the negotiation

Table 3-9 (Truth Table 1) shows the truth table for deciding the Flit format in which to operate if PCIe or CXL protocols are negotiated (with or without Management Transport protocol), and none of the following are negotiated:

- Enhanced Multi_Protocol_Enable
- Standard 256B Start Header for PCIe protocol capability
- Latency-Optimized Flit with Optional Bytes for PCIe protocol capability

Table 3-10 (Truth Table 2) provides the Truth Table for determining the Flit Format for Streaming protocols if Streaming Flit Format capability is negotiated or if Management Transport protocol is negotiated without CXL or PCIe or Streaming protocols on the same stack. Note that for Streaming protocol negotiation or for Management Transport protocol negotiation without CXL or PCIe protocol multiplexed on the same stack, there are no {FinCap.*} messages exchanged. Each side of the UCIe Link advertises its own capabilities in the {AdvCap.Adapter} message it sends. The bits in Table 3-10 represent the logical AND of the corresponding bits in the sent and received {AdvCap.Adapter} messages. Truth Table 2 must be followed for determining the Flit Format if both sides of the Link have any of the following capabilities are supported and enabled for both sides of the Link:

- Enhanced Multi-Protocol Capability
- Standard Start Header Flit for PCIe protocol capability
- Latency-Optimized Flit with Optional Bytes for PCIe protocol capability

For situations where {FinCap.Adapter} messages are sent, the bits in the truth table represent the bits set in the {FinCap.Adapter} message.

It is permitted for the Adapter OR the Protocol Layer to take the Link down to LinkError if the desired Flit Format is not negotiated or the negotiated Flit format and protocol combination is illegal (e.g., 68B Flit *Format 2* and Management Transport protocol combination).

Table 3-9. Truth Table 1

{FinCap.Adapter} bits[a]						Flit Format
Raw Format	68B Flit Mode	CXL 256B Flit Mode	PCIe Flit Mode	CXL_LatOpt_ Fmt5	CXL_LatOpt_ Fmt6	
1	x	x	x	x	x	*Format 1*: Raw Format
0	x	1	x	0	0	*Format 4*: Standard 256B Start Header Flit Format for CXL
0	x	1	x	x	1	*Format 6*: Latency-Optimized 256B with Optional Bytes Flit Format for CXL
0	x	1	x	1	0	*Format 5*: Latency-Optimized 256B without Optional Bytes Flit Format for CXL
0	x	0	1	x	x	*Format 3*: Standard 256B End Header Flit Format for PCIe
0	1	0	0	x	x	*Format 2*: 68B Flit Format

a. x indicates don't care.

Table 3-10. Truth Table 2

Logical AND of Corresponding Bits in the Sent and Received {AdvCap.Adapter} Message OR the Bits Sent in the {FinCap.Adapter} Message[c]						Final Negotiated Flit Format[a]
Raw Format[b]	68B Flit Format[c]	Standard 256B End Header Flit Format	Standard 256B Start Header Flit Format	Latency-Optimized 256B without Optional Bytes Flit Format	Latency-Optimized 256B with Optional Bytes Flit Format	
1	x	x	x	x	x	*Format 1*: Raw Format
0	1	0	0	x	0	*Format 2*: 68B Flit Format
0	x	1	0	x	0	*Format 3*: Standard 256B End Header Flit Format
0	x	x	1	x	0	*Format 4*: Standard 256B Start Header Flit Format
0	0	0	0	1	0	*Format 5*: Latency-Optimized 256B without Optional Bytes Flit Format
0	x	x	x	x	1	*Format 6*: Latency-Optimized 256B with Optional Bytes Flit Format

a. *Format 6* is the highest priority format when Raw Format is not advertised because it has the best performance characteristics. Between *Format 4* and *Format 3*, *Format 4* is higher priority because it enables lower latency through the D2D Adapter when multiplexing different protocols. *Format 5* has the highest overhead and therefore has the lowest priority relative to other formats.
b. Raw Format is always explicitly enabled through UCIe Link Control register and advertised only when it is the required format of operation to ensure interoperability, and therefore appears as a higher priority in the decision table.
c. x indicates don't care.

3.5 State Machine Hierarchy

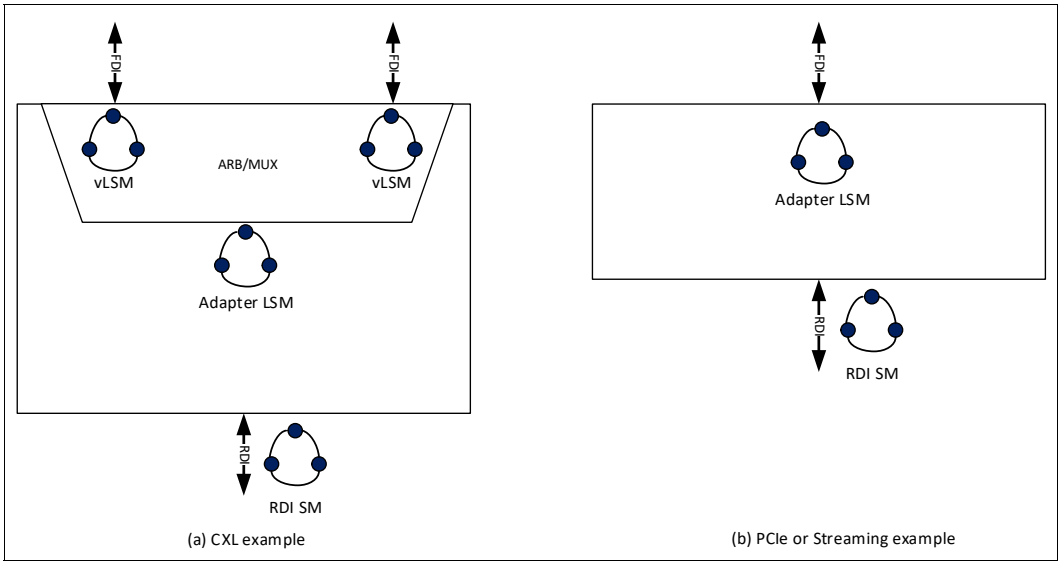
UCIe has a hierarchical approach to Link state management in order to have well-defined functionality partitioning between the different layers and also enabling common state transitions or sequencing at FDI and RDI.

Figure 3-27 shows examples of state machine hierarchy for different configurations. For CXL, the ARB/MUX vLSMs are exposed on FDI **`pl_state_sts`**. The Adapter LSM is used to coordinate Link states with remote Link Partner and is required for all configurations. Each protocol stack has its corresponding Adapter LSM. For PCIe or Streaming protocols, the Adapter LSM is exposed on FDI **`pl_state_sts`**.

The RDI state machine (SM) is used to abstract the Physical Layer states for the upper layers. The Adapter data path and RDI data width can be extended for multi-module configurations; however, there is a single RDI state machine for this configuration. The Multi-module PHY Logic creates the abstraction and coordinates between the RDI state and individual modules. The following rules apply:

- vLSM state transitions are coordinated with remote Link partner using ALMPs on mainband data path. The rules for state transitions follow the CXL 256B Flit Mode rules in the *CXL Specification*.
- Adapter LSM state transitions are coordinated with remote Link partner using {LinkMgmt.Adapter*} sideband messages. These messages are originated and received by the D2D Adapter.
- RDI SM state transitions are coordinated with the remote Link partner using {LinkMgmt.RDI*} sideband messages. These messages are originated and received by the Physical Layer.

Figure 3-27. State Machine Hierarchy Examples



General rules for State transition hierarchy are captured below. For specific sequencing, see the rules outlined in Chapter 10.0.

- Active State transitions: RDI SM must be in Active before Adapter LSM can begin negotiation to transition to Active. Adapter LSM must be in Active before vLSMs can begin negotiations to transition to Active.
- Retrain State transitions: RDI SM must be in Retrain before propagating Retrain to Adapter LSMs. If RDI SM is in Retrain, Retrain must be propagated to all Adapter LSMs that are in Active state.

Adapter must not request Retrain exit on RDI before all the relevant Adapter LSMs have transitioned to Retrain.

- PM State transitions (both L1 and L2): Both CXL.io and CXL.cachemem vLSMs (if CXL), must transition to PM before the corresponding Adapter LSM can transition to PM. All Adapter LSMs (if multiple stacks are enabled on the same Adapter) must be in PM before RDI SM is transitioned to PM.
- LinkError State transitions: RDI SM must be in LinkError before Adapter LSM can transition to LinkError. RDI SMs coordinate LinkError transition with remote Link partner using sideband, and each RDI SM propagates LinkError to all enabled Adapter LSMs. Adapter LSM must be in LinkError before propagating LinkError to both vLSMs if CXL. LinkError transition takes priority over LinkReset or Disabled transitions. Adapter must not request LinkError exit on RDI before all the relevant Adapter LSMs and CXL vLSMs have transitioned to LinkError.
- LinkReset or Disabled State transitions: Adapter LSM negotiates LinkReset or Disabled transition with its remote Link partner using sideband messages. LinkReset or Disabled is propagated to RDI SM only if all the Adapter LSMs associated with it transition to LinkReset or Disabled. Disabled transition takes priority over LinkReset transition. If RDI SM moves to LinkReset or Disabled, it must be propagated to all Adapter LSMs. If Adapter LSM moves to LinkReset or Disabled, it must propagate it to both vLSMs for CXL protocol.

For UCIe Retimers, it is the responsibility of the Retimer die to negotiate state transitions with the remote Retimer partner and make sure the different UCIe Die are in sync and do not time out waiting for a response. As an example, referring to Figure 1-18, if UCIe Die 0 sends an Active Request message for the Adapter LSM to UCIe Retimer 0, UCIe Retimer 0 must resolve with UCIe Retimer 1 that an Active Request message has been forwarded to UCIe Die 1 and that UCIe Die 1 has responded with an Active Status message before responding to UCIe Die 0 with an Active Status message. The Off Package Interconnect cannot be taken to a low power state unless all the relevant states on UCIe Die 0 AND UCIe Die 1 have reached the low power state. UCIe Retimers must respond with "Stall" encoding every 4ms while completing resolution with the remote Retimer partner.

3.6 Power Management Link States

Power management states are mandatory for PCIe and CXL protocols. FDI supports L1 and L2 power states which follow the handshake rules and state transitions of CXL 256B Flit Mode. RDI supports L1 and L2 on the interfaces for Physical Layer to perform power management optimizations; however, the Physical Layer is permitted to internally map both L1 and L2 to a common state. These together allow for global clock gating and enable system level flows like Package-Level Idle (C-states). Other Protocols are permitted to disable PM flows by always sending a PMNAK for a PM request from remote Link partner.

When Management Transport protocol is supported and negotiated with CXL.io/PCIe/Streaming on the same stack, L1 and L2 entry requests to the Adapter from the Management Port Gateway multiplexer (MPG mux) must comprehend L1 and L2 entry readiness of the Management Transport protocol as well as the co-located protocol stack, in an implementation specific manner. Additionally, the MPG mux must also follow the FDI semantics for PM rules of the co-located CXL.io/PCIe/Streaming protocol. Similarly, L1 and L2 exit would wake both the Management Transport protocol and as well as the co-located protocol stack, and exit flow semantics must adhere to the negotiated CXL.io/PCIe/Streaming protocol.

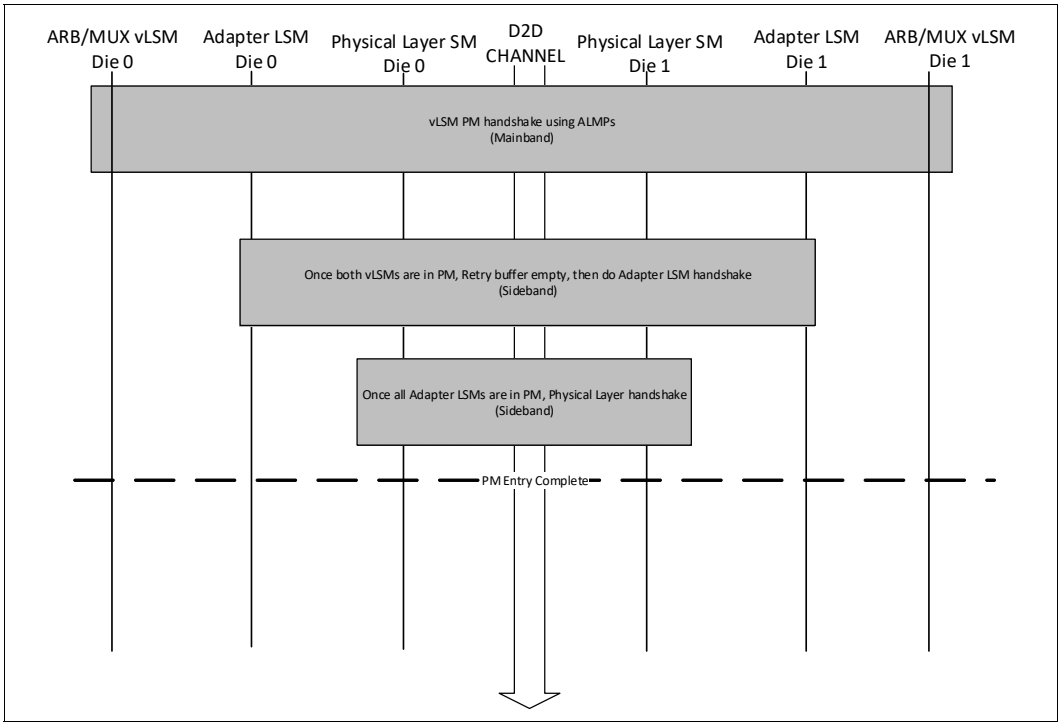
The Power management state entry sequence is as follows:

1. **Protocol Layer PM entry request**: FDI defines a common flow for PM entry request at the interface that is based on Link idle time. All protocols using UCIe must follow that flow when PM needs to be supported. For CXL protocol, D2D Adapter implements the ARB/MUX functionality and follows the handshakes defined in *CXL Specification* (corresponding to the "CXL 256B Flit Mode",

since all ALMPs also go through the Retry buffer in UCIe). Even CXL 68B Flit Mode over UCIe uses the "CXL 256B Flit Mode" ALMP formats and flows (but the Flit is truncated to 64B and two bytes of Flit header and two bytes of CRC are added by the Adapter to make a 68B Flit). For PCIe protocol in UCIe Flit Mode, PM DLLP handshakes are NOT used. Protocol Layer requests PM entry on FDI based on Link idle time. The specific algorithm and hysteresis for determining Link idle time is implementation specific.

2. **Adapter Link State Machine PM entry**: The PM transition for this is coordinated over sideband with remote Link partner. In scenarios where the Adapter is multiplexing between two protocol stacks, each stack's Link State Machine must transition to PM independently.
3. **PM entry on RDI**: Once all the Adapter's LSMs are in a PM state, the Adapter initiates PM entry on the RDI as defined in Section 10.2.9.
4. Physical Layer moves to a deeper PM state and takes the necessary actions for power management. Note that the sideband Link must remain active because the sideband Link is used to initiate PM exit.

Figure 3-28. Example of Hierarchical PM Entry for CXL



PM exit follows the reverse sequence of wake up as mentioned below:

1. Active request from Protocol Layer is transmitted across the FDI and RDI to the local Physical Layer.
2. The Physical Layer uses sideband to coordinate wake up and retraining of the physical Link.
3. Once the physical Link is retrained, the RDI is in Active state on both sides, and the Adapter LSM PM exit is triggered from both sides (coordinated via sideband messages between Adapters as outlined in the FDI PM flow). For PCIe or Streaming protocol scenarios, this also transitions the Protocol Layer to Active state on FDI.
4. For CXL protocol, this step is followed by ALMP exchanges to bring the required protocol to Active state and then protocol Flit transfer can begin.

3.7 CRC Computation

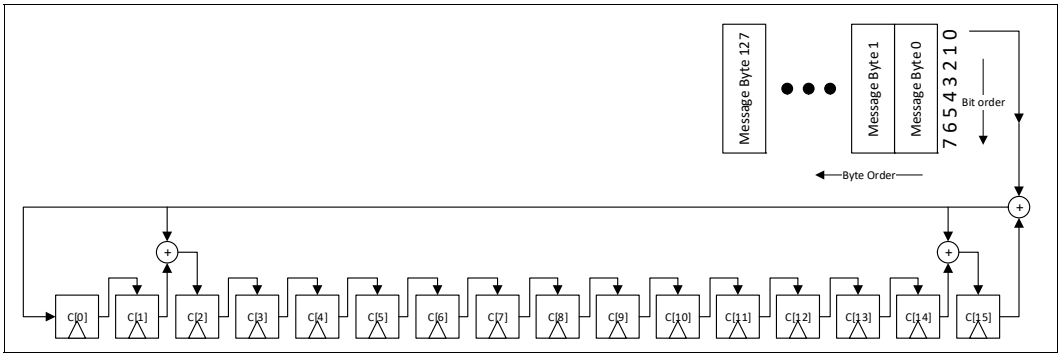
The CRC generator polynomial is $(x+1)*(x^{15} + x + 1) = x^{16} + x^{15} + x^2 + 1$. This gives a 3-bit detection guarantee for random bit errors: 2 bit detection guarantee is because of the primitive polynomial $(x^{15} + x + 1)$, and 1 additional bit error detection guarantee is provided by making it odd parity because of the (x+1) term in the polynomial.

The CRC is always computed over 128 bytes of the message. For smaller messages, the message is zero extended in the MSB. Any bytes which are part of the 128B CRC message but are not transmitted over the Link are assigned to 0b. Whenever non-CRC bytes of the Flit populated by the Adapter are included for CRC computation (e.g., the Flit Header or DLP bytes), CRC is computed after the Adapter has assigned those bytes the values that will be sent over the UCIe Link. Any reserved bits which are part of the Flit are assigned 0b for the purpose of CRC computation.

The initial value of CRC bits for CRC LFSR computation is 0000h. The CRC calculation starts with bit 0 of byte 0 of the message, and proceeds from bit 0 to bit 7 of each byte as shown in Figure 3-29. In the figure, C[15] is bit 7 of CRC Byte 1, C[14] is bit 6 of CRC Byte 1 and so on; C[7] is bit 7 of CRC Byte 0, C[6] is bit 6 of CRC Byte 0 and so on.

The Verilog code for CRC code generation is provided in crc_gen.vs (attached to the PDF copy of this Specification). This Verilog code must be used as the golden reference for implementing the CRC during encode or decode. The code is provided for the Transmit side. It takes 1024 bits (bit 1023 is bit 7 of message Byte 127, 1022 is bit 6 of message Byte 127 and so on; bit 1015 is bit 7 of message Byte 126 and so on until bit 0 is bit 0 of message Byte 0) as an input message and outputs 16 bits of CRC. On the Receiver, the CRC is computed using the received Flit bytes with appropriate zero padding in the MSB to form a 128B message. If the received CRC does not match the computed CRC, the flit is declared Invalid and a replay must be requested.

Figure 3-29. Diagram of CRC Calculation



3.8 Retry Rules

For configurations where the raw BER is higher than 1e-27, Retry must be supported in the Adapter, unless the only format of operation is Raw Format. If Retry is not supported by the Adapter, Link speeds where the raw BER is higher than 1e-27 must NOT be advertised by the Physical Layer during Link Training, unless the format of operation is Raw Format. See Table 5-26 for the raw BER characteristics of different configurations. Once Retry has been negotiated during Part 2 of Stage 3 of Link Initialization described in Section 3.2.1.2, it cannot be disabled even if Link speed degrades during runtime. Retry can only be re-negotiated at the next Link Initialization (i.e., RDI moves to Reset). For multiple stacks with a common Adapter, the Tx Retry buffer is shared between the stacks.

The Retry scheme on UCIe is a simplified version of the Retry mechanism for Flit Mode defined in *PCIe Base Specification*. The rules that differ from PCIe are as follows:

- Selective Nak and associated rules are not applicable and must not be implemented. Rx Retry Buffer-related rules are also not applicable and must not be implemented.
- Throughout the duration of Link operation, when not conflicting with PCIe rules of replay, Explicit Sequence number Flits and Ack/Nak Flits alternate. This allows for faster Ack turnaround and thus smaller Retry buffer sizes. It is permitted to send consecutive Explicit Sequence number Flits if there are no pending Ack/Nak Flits to send (see also the Implementation Note below). To meet this requirement, all Explicit Sequence Number Flit transmissions described by the PCIe rules of replay that require the condition "CONSECUTIVE_TX_EXPLICIT_SEQ_NUM_FLIT < 3" to be met require "CONSECUTIVE_TX_EXPLICIT_SEQ_NUM_FLIT < 1" to be met instead, and it is not required to send three consecutive Flits with Explicit Sequence Number.
- All 10-bit retry related counters are replaced with 8-bit counters, and the maximum-permitted sequence number is 255 (hence 1023 in all calculations is replaced with 255 and any variables defined in the "Flit Sequence Number and Retry Mechanism" section of *PCIe Base Specification* which had an initial value of 1023 instead have an initial value of 255).
- REPLAY_TIMEOUT_FLIT_COUNT is a 9-bit counter that saturates at 1FFh.
  - In addition to incrementing REPLAY_TIMEOUT_FLIT_COUNT as described in *PCIe Base Specification*, the count must also be incremented when in Active state and a Flit Time (Number of Adapter clock cycles (**lclk**) that are required to transfer 256B of data at the current Link speed and width) has elapsed since the last flit was sent and neither a Payload Flit nor a NOP flit was transmitted. The counter must be incremented for every Flit Time in which a flit was not sent (this could lead to it being incremented several times in-between flits or prior to the limit being met). The added requirement compensates for the noncontinuous transfer of NOP flits. For 68B Flit Format, data transfers are also in 256B granularity (including the PDS bytes), and thus this counter increments every time 256B of data are transmitted, OR during idle conditions in Active state, it must be incremented according to the time that is required to transfer 256B of data at the current Link speed and width.
  - Replay Schedule Rule 0 of *PCIe Base Specification* must check for REPLAY_TIMEOUT_FLIT_COUNT ≥ 375. Replay Timer Timeout error is logged in the Correctable Internal Error in the Adapter for UCIe.
- For the FLIT_REPLAY_NUM counter, it is strongly recommended to follow the rules provided in *PCIe Base Specification* for speeds ≤ 32.0 GT/s. This counter tracks the number of times that a Replay has occurred without making forward progress. Given the significantly lower probability of Replay for UCIe Links, the rules associated with ≤ 32.0 GT/s PCIe speeds are sufficient for UCIe.
- NAK_WITHDRAWAL_ALLOWED is always cleared to 0. Note that this requires implementations to set the flag NAK_SCHEDULED=1 in the "Nak Schedule 0" set of rules.
- IDLE Flit Handshake Phase is not applicable. This is because the transition to Link Active (equivalent to LTSSM being in L0 for PCIe) is managed via handshakes on sideband, and there is no requirement for IDLE Flits to be exchanged. As per PCIe rules, any Flits received with all 0s in

the Flit Header bytes are discarded by the Adapter. Any variables that are initialized during the IDLE Flit Handshake Phase in *PCIe Base Specification* are initialized to the corresponding value whenever the RDI is in Reset state or Retrain state. Similarly, PCIe rules that indicate relation to "last entry to IDLE Flit Handshake Phase" would instead apply for UCIe to "last exit from Reset or Retrain state on RDI".

- Variables applicable to Flit Sequence number and Retry mechanism that are initialized during DL_Inactive, as with PCIe, would be initialized to their corresponding values when RDI is in Reset state for UCIe.
- Sequence Number Handshake Phase must be performed on every entry of the RDI to Active state from Reset state or Retrain state (after Flit transfers are permitted). Sequence Number Handshake Phase timeout and exit to Link Retrain is 128 Flits transmitted without exiting Sequence Number Handshake Phase. As with PCIe, both NOP flits or Payload flits are permitted to be used to complete the Sequence Number Handshake Phase. If there are no Payload flits to send, the Adapter must generate NOP flits to complete the Sequence Number Handshake Phase.
- The variable "Prior Flit was Payload" is always set to 1. This bit does not exist in the Flit Header, and thus from the Retry perspective, implementations must assume that it is always set to 1.
- MAX_UNACKNOWLEDGED_FLITS is set to the lesser of:
  - Number of Flits that can be stored in the Tx Retry Buffer, or
  - 127
- Flit Discard 2 rule from PCIe does not result in a Data Link Protocol Error condition in UCIe. Receiving an invalid Flit Sequence number in a received Ack or Nak flit (see the corresponding conditions in *PCIe Base Specification* with the adjusted variable widths and values) OR a Payload Flit with an Explicit Sequence number of 0 results in an Uncorrectable Internal Error in UCIe (instead of a Data Link Protocol Error).
- Conditions from the "Flit Sequence Number and Retry Mechanism" section in *PCIe Base Specification* that led to Recovery for the Port must result in the Adapter initiating Retrain on the RDI for UCIe.

> **IMPLEMENTATION NOTE**
>
> In UCIe, to encourage power savings through dynamic clock gating, it is not required to continuously transmit NOP flits during periods in which there are no Payload flits or any Ack/Nak pending. Consider an example in which an Adapter's Tx Retry Buffer is empty and it transmitted a NOP flit with an Ack as the last flit before it stopped sending additional flits to the Physical Layer. Let's say this flit had a CRC error and hence the remote Link partner never receives this Ack. Moreover, because the remote Link partner received a flit with a CRC error, it would transmit a Nak to original sender. If the Ack is never re-sent and the remote Link partner has a corresponding Payload flit in its Tx Retry Buffer, eventually a Replay Timeout will trigger from the remote Link partner and resolve this scenario. However, rather than always relying on Replay Timeout for these kind of scenarios, it is recommended for implementations to ensure they have transmitted at least two flits with an Ack (these need not be consecutive Ack flits) before stopping flit transfer whenever a Nak is received and the transmitter has completed all the requirements of received Nak processing, including any Replay related transfers. If no new Payload Flits were received from the remote Link partner, as per PCIe rules, it is permitted to re-send the last transmitted Ack on a NOP flit as well to meet this condition.

3.9 Runtime Link Testing using Parity

UCIe defines a mechanism to detect Link health during runtime by periodically injecting parity bytes in the middle of the data stream when this mechanism is enabled. The receiver checks and logs parity errors for the inserted parity bytes.

When this mechanism is enabled, the Adapter inserts 64*N Bytes every 256*256*N Bytes of data, where N is obtained from the Error and Link Testing Control register (Field name: Number of 64 Byte Inserts). Software must set N=4 when this feature is enabled during regular Link operation for UCIe Flit mode because that makes the parity bytes also a multiple of 256B and is more consistent with the granularity of data transfer. Only bit 0 of the inserted byte has the parity information which is computed as follows:

ParityByte X, bit 0 = ^((DataByte [X]) ^ (DataByte [X + 64*N]) ^(DataByte [X + 128*N])^....^(DataByte [X + (256*256*N - 64*N)]))

The remaining 7 bits of the inserted byte are Reserved.

The Transmitter and Receiver in the Adapter must independently keep track of the number of data bytes elapsed to compute or check the parity information. If the RDI state moves away from Active state, the data count and parity is reset, and both sides must renegotiate the enabling of the Parity insertion before next entry to Active from Retrain (if the mechanism is still enabled in the Error and Link Testing Control register). When entering Active state with Parity insertion enabled, the number of data bytes elapsed begins counting from 0. On the transmitter, following the insertion of the parity information, the counter for the number of bytes elapsed to compute the parity information is reset. On the Receiver, following the receipt and check of parity bytes, the counter for the number of bytes elapsed to check the parity information is reset.

This mechanism is enabled by Software writing 1b to the enable bit in the register located in both Adapters across a UCIe Link (see Section 9.5.3.9 for register details). Software must trigger UCIe Link Retrain after writing to the enable bit on both the Adapters. Support for this feature in Raw Format is beyond the scope of this specification and is implementation-dependent. The Adapters exchange sideband messages while the Adapter LSMs are in Retrain to ensure the remote Link partner's receiver is prepared to receive the extra parity bytes in the data stream once the states transition to Active. The Adapter must not request Retrain exit to local RDI until the Parity Feature exchanges are completed. It is permitted to enable it during Initial Link bring up, by using sideband to access the remote Link partner's registers or other implementation specific means; however software must trigger Link Retrain for the feature to take effect.

Adapter sends a {ParityFeature.Req} sideband message to remote Link Partner if its Transmitter is enabled to send parity bytes ("Runtime Link Testing Tx Enable" bit in Section 9.5.3.9). Remote Adapter responds with a {ParityFeature.Ack} sideband message if its receiver is enabled and ready to accept parity bytes ("Runtime Link Testing Rx Enable" bit in Section 9.5.3.9). Figure 3-30 shows an example of a successful negotiation. If Die 0 Adapter Transmitter is enabled to insert parity bytes, it must send a {ParityFeature.Req} from Die 0 to Die 1.

Adapter responds with a {ParityFeature.Nak} if it is not ready to accept parity bytes, or if the feature has not been enabled for it yet. The requesting Adapter must log the Nak in a status register so that Software can determine that a Nak had occurred. Figure 3-31 shows an example of an unsuccessful negotiation.

Note: The Adapters are permitted to transition to a higher latency data path if the Parity Feature is enabled. The explicit Ack/Nak handshake is provided to ensure both sides have sufficient time to transition to alternate data path for this mechanism.

The Parity bytes do not consume Retimer receiver buffer credits. The Retimer receiver must not write the Parity bytes into its receiver buffer or forward these to remote Retimer partner over the Off Package Interconnect. This mechanism is to help characterize local UCIe Links only.

Figure 3-30. Successful Parity Feature negotiation between Die 1 Tx and Die 0 Rx

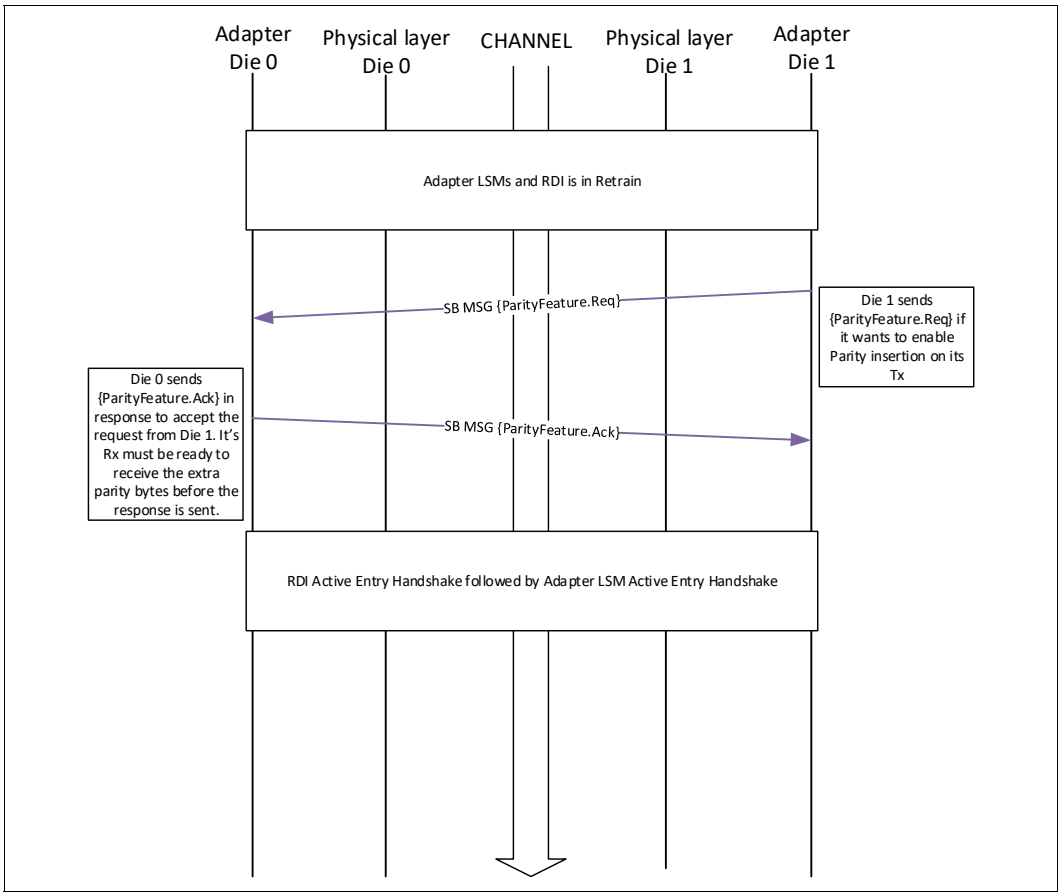
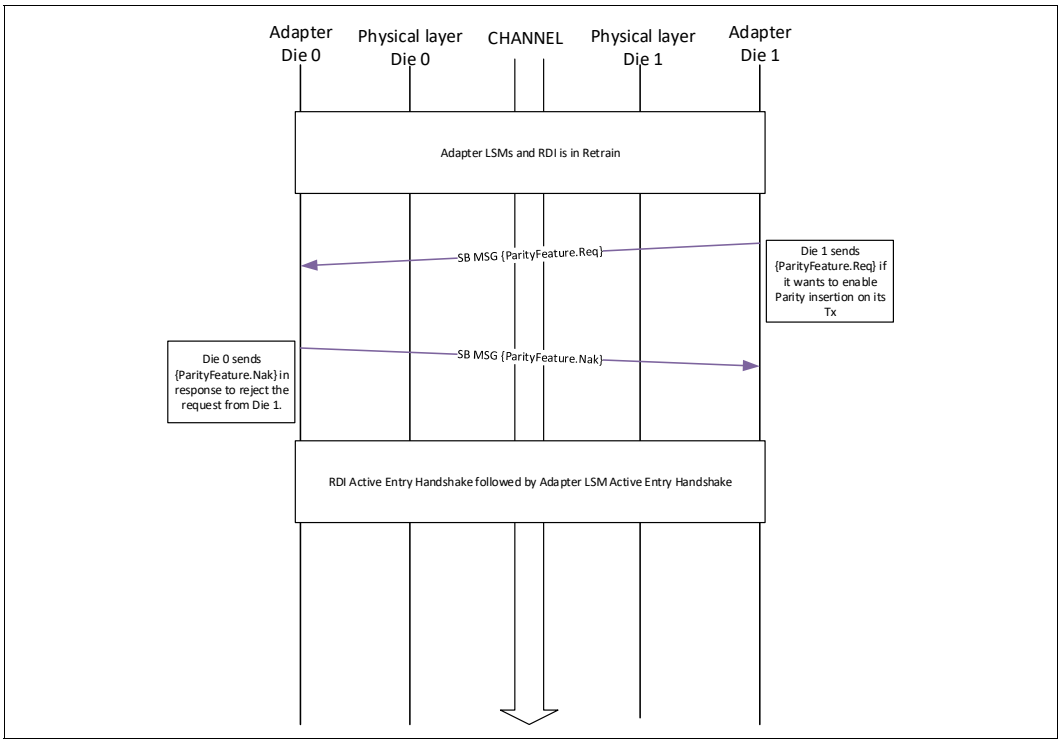


Figure 3-31. Unsuccessful Parity Feature negotiation between Die 1 Tx and Die 0 Rx



If a parity error is detected by a chiplet, the error is treated as a Correctable error and reported via the correctable error reporting mechanism. By enabling interrupt on correctable errors, SW can implement a BER counter in SW, if so desired.

When a Pause Data Stream occurs the Pause Data Stream and corresponding padding bytes are included in the number of bytes elapsed before parity injection as well as parity computation.

§ §

4.0 Logical Physical Layer

The Logical PHY comprehends the following functions:

- Link initialization, training and power management states
- Byte to Lane mapping for data transmission over Lanes
- Interconnect redundancy remapping (when required)
- Transmitting and receiving sideband messages
- Scrambling and training pattern generation
- Lane reversal
- Width degradation (when applicable)

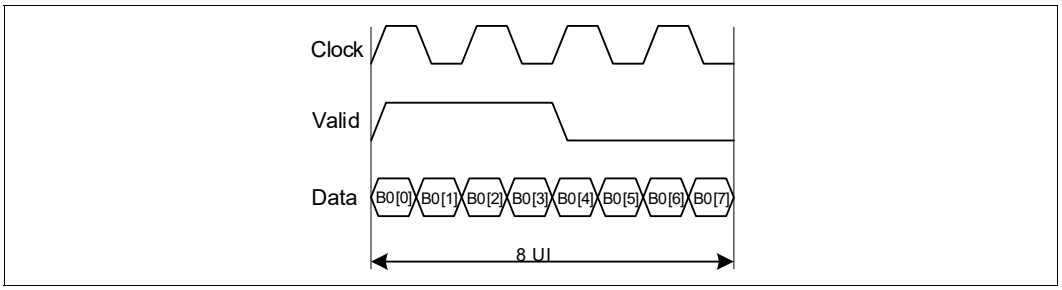
4.1 Data and Sideband Transmission Flow

This specification defines clock, valid and Data to send and receive data over the physical Lanes. The transmitted data is framed by the valid signal.

4.1.1 Byte to Lane Mapping

Data packets are transmitted in Bytes. Within each Byte, bit [0] is transmitted first. Figure 4-1 shows an example of bit arrangement within one byte transmission over Lane 0.

Figure 4-1. Bit arrangement within a byte transfer



Each Byte is transmitted on a separate Lane. Byte 0 (B0) is transmitted on Lane 0, Byte 1 is transmitted on Lane 1 and so on.

Figure 4-2 shows an example of a 256B Flit transmitted over a x64 interface (one x64 Advanced Package module or two x32 Advanced Package modules or four Standard Package modules). If the I/O width changes to x32 or x16 interface (Standard Package), transmission of one Byte per Lane is preserved as shown in Figure 4-3 and Figure 4-4 respectively.

Figure 4-5 shows an example for a width degraded Standard Package module.

Figure 4-2. Byte map for x64 interface

UI \ LANE	0	1	2	3	4	5	6	7	...	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
0 - 7	B00	B01	B02	B03	B04	B05	B06	B07	...	B48	B49	B50	B51	B52	B53	B54	B55	B56	B57	B58	B59	B60	B61	B62	B63
8 - 15	B64	B65	B66	B67	B68	B69	B70	B71	...	B112	B113	B114	B115	B116	B117	B118	B119	B120	B121	B122	B123	B124	B125	B126	B127
16 - 23	B128	B129	B130	B131	B132	B133	B134	B135	...	B176	B177	B178	B179	B180	B181	B182	B183	B184	B185	B186	B187	B188	B189	B190	B191
24 - 31	B192	B193	B194	B195	B196	B197	B198	B199	...	B240	B241	B242	B243	B244	B245	B246	B247	B248	B249	B250	B251	B252	B253	B254	B255

Figure 4-3. Byte map for x32 interface

UI \ Lane	0	1	2	3	4	5	6	7	...	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0 - 7	B0	B1	B2	B3	B4	B5	B6	B7	...	B16	B17	B18	B19	B20	B21	B22	B23	B24	B25	B26	B27	B28	B29	B30	B31
8 - 15	B32	B33	B34	B35	B36	B37	B38	B39	...	B48	B49	B50	B51	B52	B53	B54	B55	B56	B57	B58	B59	B60	B61	B62	B63
16 - 23	B64	B65	B66	B67	B68	B69	B70	B71	...	B80	B81	B82	B83	B84	B85	B86	B87	B88	B89	B90	B91	B92	B93	B94	B95
24 - 31	B96	B97	B98	B99	B100	B101	B102	B103	...	B112	B113	B114	B115	B116	B117	B118	B119	B120	B121	B122	B123	B124	B125	B126	B127
32 - 39	B128	B129	B130	B131	B132	B133	B134	B135	...	B144	B145	B146	B147	B148	B149	B150	B151	B152	B153	B154	B155	B156	B157	B158	B159
40 - 47	B160	B161	B162	B163	B164	B165	B166	B167	...	B176	B177	B178	B179	B180	B181	B182	B183	B184	B185	B186	B187	B188	B189	B190	B191
48 - 55	B192	B193	B194	B195	B196	B197	B198	B199	...	B208	B209	B210	B211	B212	B213	B214	B215	B216	B217	B218	B219	B220	B221	B222	B223
56 - 63	B224	B225	B226	B227	B228	B229	B230	B231	...	B240	B241	B242	B243	B244	B245	B246	B247	B248	B249	B250	B251	B252	B253	B254	B255

Figure 4-4. Byte map for x16 interface

UI \ Lane	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0 - 7	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15
8 - 15	B16	B17	B18	B19	B20	B21	B22	B23	B24	B25	B26	B27	B28	B29	B30	B31
16 - 23	B32	B33	B34	B35	B36	B37	B38	B39	B40	B41	B42	B43	B44	B45	B46	B47
24 - 31	B48	B49	B50	B51	B52	B53	B54	B55	B56	B57	B58	B59	B60	B61	B62	B63
32 - 39	B64	B65	B66	B67	B68	B69	B70	B71	B72	B73	B74	B75	B76	B77	B78	B79
40 - 47	B80	B81	B82	B83	B84	B85	B86	B87	B88	B89	B90	B91	B92	B93	B94	B95
48 - 55	B96	B97	B98	B99	B100	B101	B102	B103	B104	B105	B106	B107	B108	B109	B110	B111
56 - 63	B112	B113	B114	B115	B116	B117	B118	B119	B120	B121	B122	B123	B124	B125	B126	B127
64 - 71	B128	B129	B130	B131	B132	B133	B134	B135	B136	B137	B138	B139	B140	B141	B142	B143
72 - 79	B144	B145	B146	B147	B148	B149	B150	B151	B152	B153	B154	B155	B156	B157	B158	B159
80 - 87	B160	B161	B162	B163	B164	B165	B166	B167	B168	B169	B170	B171	B172	B173	B174	B175
88 - 95	B176	B177	B178	B179	B180	B181	B182	B183	B184	B185	B186	B187	B188	B189	B190	B191
96 - 103	B192	B193	B194	B195	B196	B197	B198	B199	B200	B201	B202	B203	B204	B205	B206	B207
104 - 111	B208	B209	B210	B211	B212	B213	B214	B215	B216	B217	B218	B219	B220	B221	B222	B223
112 - 119	B224	B225	B226	B227	B228	B229	B230	B231	B232	B233	B234	B235	B236	B237	B238	B239
120 - 127	B240	B241	B242	B243	B244	B245	B246	B247	B248	B249	B250	B251	B252	B253	B254	B255

Figure 4-5. Byte to Lane mapping for Standard package x16 degraded to x8

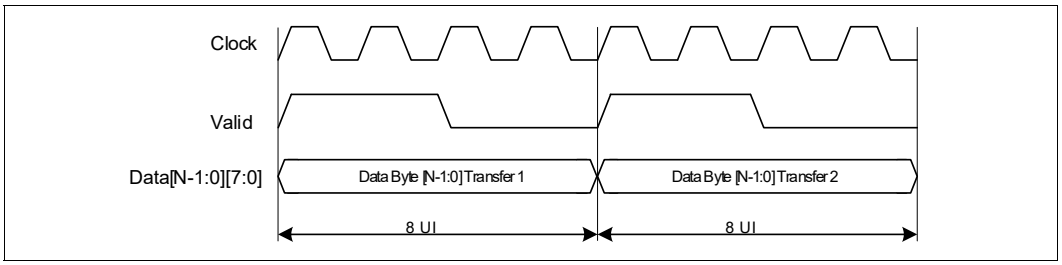
Lane	8	9	10	11	12	13	14	15
	or							
Lane / UI	0	1	2	3	4	5	6	7
0 - 7	B0	B1	B2	B3	B4	B5	B6	B7
8 - 15	B8	B9	B10	B11	B12	B13	B14	B15
16 - 23	B16	B17	B18	B19	B20	B21	B22	B23
24 - 31	B24	B25	B26	B27	B28	B29	B30	B31
32 - 39	B32	B33	B34	B35	B36	B37	B38	B39
40 - 47	B40	B41	B42	B43	B44	B45	B46	B47
48 - 55	B48	B49	B50	B51	B52	B53	B54	B55
56 - 63	B56	B57	B58	B59	B60	B61	B62	B63
64 - 71	B64	B65	B66	B67	B68	B69	B70	B71
72 - 79	B72	B73	B74	B75	B76	B77	B78	B79
80 - 87	B80	B81	B82	B83	B84	B85	B86	B87
88 - 95	B88	B89	B90	B91	B92	B93	B94	B95
96 - 103	B96	B97	B98	B99	B100	B101	B102	B103
104 - 111	B104	B105	B106	B107	B108	B109	B110	B111
112 - 119	B112	B113	B114	B115	B116	B117	B118	B119
120 - 127	B120	B121	B122	B123	B124	B125	B126	B127
128-135	B128	B129	B130	B131	B132	B133	B134	B135
136-143	B136	B137	B138	B139	B140	B141	B142	B143
144-151	B144	B145	B146	B147	B148	B149	B150	B151
152-159	B152	B153	B154	B155	B156	B157	B158	B159
160-167	B160	B161	B162	B163	B164	B165	B166	B167
168-175	B168	B169	B170	B171	B172	B173	B174	B175
176-183	B176	B177	B178	B179	B180	B181	B182	B183
184-191	B184	B185	B186	B187	B188	B189	B190	B191
192-199	B192	B193	B194	B195	B196	B197	B198	B199
200-207	B200	B201	B202	B203	B204	B205	B206	B207
208-215	B208	B209	B210	B211	B212	B213	B214	B215
216-223	B216	B217	B218	B219	B220	B221	B222	B223
224-231	B224	B225	B226	B227	B228	B229	B230	B231
232-239	B232	B233	B234	B235	B236	B237	B238	B239
240-247	B240	B241	B242	B243	B244	B245	B246	B247
248-255	B248	B249	B250	B251	B252	B253	B254	B255

4.1.2 Valid Framing

Valid signal is used to frame the transmitted data. For each 8-bit data packet, valid is asserted for the first 4 UI and de-asserted for 4 UI. This will allow data transfer in Raw Format or various Flit Formats as described in Chapter 3.0 using one or multiple valid frames. An example is shown in Figure 4-6 where Transfer 1 and Transfer 2 can be from the same Flit or different Flits.

Note: An 8-UI block assertion is enforced by the Transmitter and tracked by the Receiver during Active state. This means that following the first valid transfer of data over mainband in Active state, each subsequent transfer is after an integer multiple of 8 UI from the rising edge of Valid of the first transfer. Note that for Retimers, this means that the first transfer after entering the Active state cannot be a 'No Flit data transfer + 1 credit release' encoding; this is acceptable because the Retimer-advertised credits are replenished or readvertised whenever the state moves away from Active.

Figure 4-6. Valid framing example



4.1.2.1 Valid Framing for Retimers

The UCIe Retimer releases credits to its local UCIe die using the Valid wire, as described in Table 4-1. Each credit tracks 256 Bytes of data (including any FEC, CRC, etc). The Valid Framing encodings ensure triple bit flip detection guarantee. It is permitted for receiver implementations to trigger Retrain on any bit error (using triple bit flip detection guarantee). It is also permitted for receiver implementations to use the encodings below to correct single bit errors (with the understanding that three bit error detection is lost, and its contribution to overall FIT is negligible).

Table 4-1. Valid framing for Retimers

8-UI Valid[a]	Encoding
00000000b	No Flit data transfer + no credit release
00001111b	Flit data transfer valid + no credit release
11110000b	No Flit data transfer + 1 credit release
11111111b	Flit data transfer valid + 1 Credit release

a. Note that the bits above are transmitted on the Link in order from right to left (i.e., bit 0 is transmitted on the Link first, followed by bit 1 and so on until bit 7).

4.1.3 Clock Gating

Forwarded mainband clocks on the UCIe Link must be gated when Valid signal is low after providing fixed 16 UI (8 cycles) of postamble clock for half-rate clocking and 32 UI (8 cycles) of postamble clock for quarter-rate clocking, unless free running clock mode is negotiated or Runtime Recalibration has been requested by the remote Link partner. Data and Clock signal parking levels when clocks are gated are described in Section 5.11.

Note that the clock postamble is required any time that the clock can toggle with Valid assertion, and the clock needs to stop toggling, regardless of LTSM state.

Figure 4-7. Clock gating