



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.

© 2022–2024 UNIVERSAL CHIPLET INTERCONNECT EXPRESS, INC. ALL RIGHTS RESERVED.

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

Terminology	25
Reference Documents	34
Revision History	34
1.0 Introduction	35
1.1 UCIe Components.....	40
1.1.1 Protocol Layer.....	40
1.1.2 Die-to-Die (D2D) Adapter.....	41
1.1.3 Physical Layer.....	41
1.1.4 Interfaces	42
1.2 UCIe Configurations.....	42
1.2.1 Single Module Configuration.....	42
1.2.2 Multi-module Configurations	43
1.2.3 Sideband-only Configurations.....	44
1.3 UCIe Retimers.....	45
1.4 UCIe Key Performance Targets	47
1.5 Interoperability	48
2.0 Protocol Layer	49
2.1 PCIe	50
2.1.1 Raw Format.....	51
2.1.2 Standard 256B End Header Flit Format.....	51
2.1.3 68B Flit Format	51
2.1.4 Standard 256B Start Header Flit Format	51
2.1.5 Latency-Optimized 256B with Optional Bytes Flit Format.....	52
2.2 CXL 256B Flit Mode.....	52
2.2.1 Raw Format.....	52
2.2.2 Latency-Optimized 256B Flit Formats	52
2.2.3 Standard 256B Start Header Flit Format	53
2.3 CXL 68B Flit Mode	53
2.3.1 Raw Format.....	53
2.3.2 68B Flit Format	53
2.4 Streaming Protocol	54
2.4.1 Raw Format.....	54
2.4.2 68B Flit Format	54
2.4.3 Standard 256B Flit Formats	54
2.4.4 Latency-Optimized 256B Flit Formats	55
2.5 Management Transport Protocol	55
3.0 Die-to-Die Adapter.....	56
3.1 Stack Multiplexing	57
3.2 Link Initialization	60
3.2.1 Stage 3 of Link Initialization: Adapter Initialization	60
3.2.1.1 Part 1: Determine Local Capabilities	60
3.2.1.2 Part 2: Parameter Exchange with Remote Link Partner	61
3.2.1.3 FDI bring up	69
3.3 Operation Formats.....	70
3.3.1 Raw Format for All Protocols	70
3.3.2 68B Flit Format	70
3.3.2.1 68B Flit Format Alignment and Padding Rules	72
3.3.3 Standard 256B Flit Formats	74
3.3.4 Latency-Optimized 256B Flit Formats	79

3.3.5	Flit Format-related Implementation Requirements for Protocol Layer and Adapter.....	83
3.4	Decision Table for Flit Format and Protocol.....	84
3.5	State Machine Hierarchy	88
3.6	Power Management Link States	89
3.7	CRC Computation	91
3.8	Retry Rules.....	92
3.9	Runtime Link Testing using Parity	94
4.0	Logical Physical Layer.....	97
4.1	Data and Sideband Transmission Flow	97
4.1.1	Byte to Lane Mapping	97
4.1.2	Valid Framing	99
4.1.2.1	Valid Framing for Retimers	100
4.1.3	Clock Gating	100
4.1.4	Free Running Clock Mode	101
4.1.5	Sideband transmission	101
4.1.5.1	Sideband Performant Mode Operation (PMO)	101
4.2	Lane Reversal	102
4.2.1	Lane ID	103
4.3	Interconnect redundancy remapping	105
4.3.1	Data Lane repair	105
4.3.2	Data Lane repair with Lane reversal	107
4.3.3	Data Lane repair implementation.....	107
4.3.3.1	Single Lane repair	107
4.3.3.2	Two Lane repair.....	109
4.3.3.3	Single Lane repair with Lane reversal	111
4.3.3.3.1	x64 Advanced Package Pseudo Codes	111
4.3.3.3.2	x32 Advanced Package Pseudo Codes	111
4.3.3.4	Two Lane repair with Lane reversal	112
4.3.3.4.1	x64 Advanced Package Pseudo Codes	112
4.3.3.4.2	x32 Advanced Package Pseudo Codes	113
4.3.4	Clock and Track Lane remapping	114
4.3.5	Clock and Track Lane repair implementation	115
4.3.6	Valid Repair and implementation	116
4.3.7	Width Degrade in Standard Package Interfaces	117
4.4	Data to Clock Training and Test Modes	117
4.4.1	Scrambling and training pattern generation	119
4.5	Link Initialization and Training.....	122
4.5.1	Link Training Basic Operations	123
4.5.1.1	Transmitter initiated Data to Clock Point Test.....	123
4.5.1.2	Transmitter initiated Data to Clock Eye Width Sweep	125
4.5.1.3	Receiver initiated Data to Clock point test	126
4.5.1.4	Receiver initiated Data to Clock Eye Width Sweep.....	127
4.5.2	Link Training with Retimer	128
4.5.3	Link Training State Machine	129
4.5.3.1	RESET	131
4.5.3.2	Sideband Initialization (SBINIT)	132
4.5.3.3	MBINIT.....	135
4.5.3.3.1	MBINIT.PARAM	136
4.5.3.3.2	MBINIT.CAL	145
4.5.3.3.3	MBINIT.REPAIRCLK	145
4.5.3.3.4	MBINIT.REPAIRVAL	148
4.5.3.3.5	MBINIT.REVERSALMB	150
4.5.3.3.6	MBINIT.REPAIRMB	152
4.5.3.4	MBTRAIN	154

4.5.3.4.1	MBTRAIN.VALVREF	155
4.5.3.4.2	MBTRAIN.DATAVREF	156
4.5.3.4.3	MBTRAIN.SPEEDIDLE	157
4.5.3.4.4	MBTRAIN.TXSELFCAL	157
4.5.3.4.5	MBTRAIN.RXCLKCAL	157
4.5.3.4.6	MBTRAIN.VALTRAINCENTER	158
4.5.3.4.7	MBTRAIN.VALTRAINVREF	158
4.5.3.4.8	MBTRAIN.DATATRAINCENTER1	159
4.5.3.4.9	MBTRAIN.DATATRAINVREF	160
4.5.3.4.10	MBTRAIN.RXDESKEW	160
4.5.3.4.11	MBTRAIN.DATATRAINCENTER2	161
4.5.3.4.12	MBTRAIN.LINKSPEED	162
4.5.3.4.13	MBTRAIN.REPAIR	165
4.5.3.5	LINKINIT	166
4.5.3.6	ACTIVE	166
4.5.3.7	PHYRETRAIN	166
4.5.3.7.1	Adapter initiated PHY retrain	168
4.5.3.7.2	PHY initiated PHY retrain	168
4.5.3.7.3	Remote Die requested PHY retrain	169
4.5.3.7.4	PHY retrain from LINKSPEED	169
4.5.3.8	TRAINERROR	169
4.5.3.9	L1/L2	170
4.6	Runtime Recalibration	170
4.7	Multi-module Link	170
4.7.1	Multi-module initialization	171
4.7.1.1	Sideband Assignment and Retimer Credits for Multi-module Configurations	174
4.7.1.2	Examples of MMPL Synchronization	174
4.7.1.2.1	Example 1: MMPL Resolution results in a Width Degrade per Module	175
4.7.1.2.2	Example 2: MMPL Resolution results in a Speed Degrade	176
4.7.1.2.3	Example 3: MMPL Resolution results in a Module Disable	176
4.7.2	Multi-module Interoperability between x64 and x32 Advanced Packages	177
4.8	Sideband PHY Arbitration between MPMs and Link Management Packets	178
5.0	Electrical Layer (2D and 2.5D)	180
5.1	Interoperability	180
5.1.1	Data rates	180
5.1.2	Reference Clock (REFCLK)	181
5.2	Overview	182
5.2.1	Interface Overview	182
5.2.2	Electrical summary	184
5.3	Transmitter Specification	185
5.3.1	Driver Topology	185
5.3.2	Transmitter Electrical parameters	186
5.3.3	24 GT/s and 32 GT/s Transmitter Equalization	187
5.4	Receiver Specification	189
5.4.1	Receiver Electrical Parameters	190
5.4.2	Rx Termination	190
5.4.3	24 and 32 GT/s Receiver Equalization	193
5.5	Clocking	193
5.5.1	Track	194
5.6	Supply noise and clock skew	196
5.7	Ball-out and Channel Specification	197

5.7.1	Voltage Transfer Function	199
5.7.2	Advanced Package.....	200
5.7.2.1	Loss and Crosstalk Mask	201
5.7.2.2	x64 Advanced Package Module Bump Map.....	201
5.7.2.3	x32 Advanced Package Module Bump Map.....	210
5.7.2.4	x64 and x32 Advanced Package Module Interoperability	216
5.7.2.5	Module Naming of Advanced Package Modules	219
5.7.3	Standard Package	223
5.7.3.1	x16 Standard Package Module Bump Map.....	224
5.7.3.2	x8 Standard Package Module Bump Map	227
5.7.3.3	x16 and x8 Standard Package Module Interoperability.....	227
5.7.3.4	Module Naming of Standard Package Modules	228
5.7.3.4.1	Module Degrade Rules	232
5.7.4	UCIE-S Sideband-only Port	234
5.8	Tightly Coupled Mode.....	235
5.9	Interconnect redundancy Remapping	235
5.9.1	Advanced Package Lane Remapping.....	235
5.9.2	Standard Package Lane remapping	236
5.10	BER requirements, CRC and retry	236
5.11	Valid and Clock Gating	237
5.12	Electrical Idle	239
5.13	Sideband signaling.....	239
5.13.1	Sideband Electrical Parameters	240
5.13.2	Auxiliary Clock (AUXCLK)	241
6.0	UCIE-3D.....	242
6.1	Introduction.....	242
6.2	UCIE-3D Features and Summary	242
6.3	UCIE-3D Tx, Rx, and Clocking	244
6.4	Electrical Specification.....	245
6.4.1	Timing Budget	245
6.4.2	ESD and Energy Efficiency	247
6.4.3	UCIE-3D Module and Bump Map	248
6.4.4	Repair Strategy.....	249
6.4.5	Channel and Data Rate Extension	251
7.0	Sideband	252
7.1	Protocol Specification	252
7.1.1	Packet Types	253
7.1.2	Packet Formats	256
7.1.2.1	Register Access Packets	256
7.1.2.2	Messages without Data	259
7.1.2.3	Messages with data payloads.....	265
7.1.2.4	Management Port Message (MPM) with Data	274
7.1.2.4.1	Common Fields in MPM Header of MPM with Data Messages on Sideband	274
7.1.2.4.2	Encapsulated MTP Message	275
7.1.2.4.3	Vendor-defined Management Port Gateway Message	276
7.1.2.5	MPMs without Data.....	276
7.1.2.5.1	Common Header Fields of MPM without Data Messages on Sideband.....	277
7.1.2.5.2	Management Port Gateway Capabilities Message....	277
7.1.2.5.3	Credit Return Message	278
7.1.2.5.4	Init Done Message	279
7.1.2.5.5	PM Message	279

7.1.2.5.6	Vendor-defined Management Port Gateway Message.....	279
7.1.3	Flow Control and Data Integrity.....	280
7.1.3.1	Flow Control and Data Integrity over FDI and RDI	280
7.1.3.2	Flow Control and Data Integrity over UCIE sideband Link between dies	281
7.1.3.3	End-to-End flow control and forward progress for UCIE Link sideband.....	281
7.1.4	Operation on RDI and FDI	284
8.0	System Architecture	285
8.1	UCIE Manageability	285
8.1.1	Overview	285
8.1.2	Theory of Operation.....	286
8.1.3	UCIE Management Transport	289
8.1.3.1	UCIE Management Transport Packet	289
8.1.3.1.1	Traffic Class and Packet Ordering Requirements....	291
8.1.3.1.2	Packet Length	293
8.1.3.1.3	Management Protocol.....	293
8.1.3.2	Management Network ID	294
8.1.3.3	Routing	295
8.1.3.3.1	Routing of a Packet from a Management Entity within the Chiplet	295
8.1.3.3.2	Routing of a Packet Received on a Management Port	297
8.1.3.4	Packet Integrity Protection	297
8.1.3.4.1	CRC Integrity Protection	297
8.1.3.5	Access Control.....	298
8.1.3.5.1	Standard Asset Classes	300
8.1.3.5.2	Security Director	302
8.1.3.6	Initialization and Configuration	302
8.1.3.6.1	Management Capability Directory.....	304
8.1.3.6.2	Chiplet Capability Structure.....	306
8.1.3.6.3	Access Control Capability Structure	315
8.1.3.6.4	Security Clearance Group Capability Structure	322
8.1.4	UCIE Memory Access Protocol	323
8.1.4.1	UCIE Memory Access Protocol Packets	324
8.1.4.1.1	UCIE Memory Request Packet.....	324
8.1.4.1.2	UCIE Memory Access Response Packet	326
8.1.4.2	UCIE Memory Access Protocol Capability Structure	328
8.2	Management Transport Packet (MTP) Encapsulation	331
8.2.1	MTP Encapsulation Architecture Overview	331
8.2.2	Management Port Messages.....	335
8.2.2.1	Sideband	335
8.2.2.2	Mainband.....	336
8.2.2.2.1	MPMs with Data.....	336
8.2.2.2.2	MPMs without Data	338
8.2.3	Management Transport Path Setup	341
8.2.3.1	Sideband	341
8.2.3.1.1	Negotiation Phase Steps	341
8.2.3.1.2	Initialization Phase Steps	341
8.2.3.1.3	Other Sideband Management Transport Path Rules	346
8.2.3.2	Mainband.....	346
8.2.3.2.1	Negotiation Phase Steps	346
8.2.3.2.2	Initialization Phase Steps	346

8.2.3.2.3	Other Mainband Management Transport Path Rules	349
8.2.4	Common Rules for Management Transport over Sideband and Mainband	349
8.2.4.1	Management Packet Flow Control	349
8.2.4.2	Segmentation.....	351
8.2.4.3	Interleaving and Multi-module Sideband and Multi-stack Mainband Ordering.....	352
	8.2.4.3.1 Transmitter Rules	352
	8.2.4.3.2 Receiver Rules	353
8.2.4.4	'Init Done' Timeout Flow	355
8.2.5	Other Management Transport Details	355
8.2.5.1	Sideband	355
	8.2.5.1.1 Management Port Gateway Flow Control over RDI..	355
	8.2.5.1.2 MPMs with Data Length Rules.....	355
	8.2.5.1.3 Sideband Runtime Management Transport Path Monitoring — Heartbeat Mechanism	356
	8.2.5.1.4 Sideband Management Path Power Management Rules	357
	8.2.5.1.5 Management Port Gateway Mux Arbitration	358
8.2.5.2	Mainband.....	358
	8.2.5.2.1 NOP Message	358
	8.2.5.2.2 Credit Return DWORD Format	358
	8.2.5.2.3 Management Flit Formats	359
	8.2.5.2.4 L1/L2 Link States and Management Transport	362
	8.2.5.2.5 Link Reset/Link Disable and Management Transport.....	362
8.2.6	Retimers and Management Transport.....	363
8.3	UCIE Debug and Test Architecture (UDA).....	363
8.3.1	Overview	363
8.3.1.1	DFx Management Hub (DMH)	364
8.3.1.2	DFx Management Spoke (DMS).....	364
8.3.1.3	Supported Protocols	365
	8.3.1.3.1 UCIE Memory Access Protocol (UMAP)	365
	8.3.1.3.2 Vendor-defined Test and Debug Protocol.....	365
8.3.1.4	UDM and UCIE Memory Access Protocol Message Encapsulation over UCIE	366
8.3.1.5	UCIE Test Port Options and Other Considerations	366
	8.3.1.5.1 Determinism Considerations.....	366
8.3.1.6	DFx Security	366
8.3.2	Sort/Pre-bond Chiplet Testing with UDA	367
8.3.3	SiP-level Chiplet Testing with UDA	368
8.3.4	System Debug with UDA	369
8.3.5	DMH/DMS Registers	369
8.3.5.1	DMH/DMS Register Address Space and Access Mechanism	369
8.3.5.2	DMH Registers.....	370
	8.3.5.2.1 Ver (Offset 00h)	371
	8.3.5.2.2 Capability ID (Offset 02h)	371
	8.3.5.2.3 DBG_CAP — Debug Capabilities (Offset 04h)	372
	8.3.5.2.4 DBG_CTL — Debug Control (Offset 08h)	372
	8.3.5.2.5 DBG_STS — Debug Status (Offset Ah).....	372
	8.3.5.2.6 DMH_Length_Low — DMH Register Space Length Low (Offset 10h).....	372
	8.3.5.2.7 DMH_Length_High — DMH Register Space Length High (Offset 14h)	372

8.3.5.2.8	DMH_Ext_Cap_Low — DMH Extended Capability Pointer Low (Offset 18h).....	373
8.3.5.2.9	DMH_Ext_Cap_High — DMH Extended Capability Pointer High (Offset 1Ch).....	373
8.3.5.2.10	DMS_Start_Low — DMS Starting Address Low (Offset 20h)	373
8.3.5.2.11	DMS_Start_High — DMS Starting Address High (Offset 24h)	373
8.3.5.3	DMS Registers	373
8.3.5.3.1	"Empty" Spoke Registers	374
8.3.5.3.2	Common DMS Registers for All Non-empty Spokes	374
8.3.5.3.3	DMS Registers of UCIE Spoke Types ('Spoke Type' = 0 through 2)	379
8.3.5.3.4	DMS Registers of Vendor-defined Spoke Types ('Spoke Type' = 128 through 255)	383
8.3.5.3.5	DMS Register Implementation in UCIE Adapter and in UCIE Physical Layer.....	384
9.0 Configuration and Parameters		385
9.1	High-level Software View of UCIE.....	385
9.2	SW Discovery of UCIE Links	386
9.3	Register Location Details and Access Mechanism.....	386
9.4	Software view Examples.....	388
9.5	UCIE Registers	391
9.5.1	UCIE Link DVSEC	391
9.5.1.1	PCI Express Extended Capability Header (Offset 0h)	393
9.5.1.2	Designated Vendor Specific Header 1, 2 (Offsets 4h and 8h)	393
9.5.1.3	Capability Descriptor (Offset Ah)	394
9.5.1.4	UCIE Link DVSEC - UCIE Link Capability (Offset Ch).....	395
9.5.1.5	UCIE Link DVSEC - UCIE Link Control (Offset 10h).....	396
9.5.1.6	UCIE Link DVSEC - UCIE Link Status (Offset 14h).....	398
9.5.1.7	UCIE Link DVSEC - Link Event Notification Control (Offset 18h) ..	401
9.5.1.8	UCIE Link DVSEC - Error Notification Control (Offset 1Ah)	401
9.5.1.9	UCIE Link DVSEC - Register Locator 0, 1, 2, 3 Low (Offset 1Ch and when Register Locators 1, 2, 3 are present Offsets 24h, 2Ch, and 34h respectively)	405
9.5.1.10	UCIE Link DVSEC - Register Locator 0, 1, 2, 3 High (Offset 20h and when Register Locators 1, 2, 3 Are Present Offsets 28h, 30h, and 38h respectively)	406
9.5.1.11	UCIE Link DVSEC - Sideband Mailbox Index Low (Offset is design dependent).....	406
9.5.1.12	UCIE Link DVSEC - Sideband Mailbox Index High (Offset is design dependent).....	407
9.5.1.13	UCIE Link DVSEC - Sideband Mailbox Data Low (Offset is design dependent).....	407
9.5.1.14	UCIE Link DVSEC - Sideband Mailbox Data High (Offset is design dependent).....	407
9.5.1.15	UCIE Link DVSEC - Sideband Mailbox Control (Offset is design dependent).....	408
9.5.1.16	UCIE Link DVSEC - Sideband Mailbox Status (Offset is design dependent).....	408
9.5.1.17	UCIE Link DVSEC - Requester ID (Offset is design dependent) ...	408
9.5.1.18	UCIE Link DVSEC - Associated Port Numbers (Offset is design dependent).....	409
9.5.1.19	Examples of setting the Length field in DVSEC for various Scenarios	409
9.5.2	UCIE Switch Register Block (UiSRB) DVSEC Capability	409

9.5.2.1	PCI Express Extended Capability Header (Offset 0h)	409
9.5.2.2	Designated Vendor Specific Header 1, 2 (Offsets 4h and 8h)	409
9.5.2.3	PCIe Switch Register Block (UiSRB) Base Address (Offset Ch)....	410
9.5.3	D2D/PHY Register Block	411
9.5.3.1	PCIe Register Block Header.....	411
9.5.3.2	Uncorrectable Error Status Register (Offset 10h)	411
9.5.3.3	Uncorrectable Error Mask Register (Offset 14h).....	412
9.5.3.4	Uncorrectable Error Severity Register (Offset 18h)	413
9.5.3.5	Correctable Error Status Register (Offset 1Ch)	413
9.5.3.6	Correctable Error Mask Register (Offset 20h)	414
9.5.3.7	Header Log 1 Register (Offset 24h)	414
9.5.3.8	Header Log 2 Register (Offset 2Ch)	415
9.5.3.9	Error and Link Testing Control Register (Offset 30h).	417
9.5.3.10	Runtime Link Testing Parity Log 0 (Offset 34h)	418
9.5.3.11	Runtime Link Testing Parity Log 1 (Offset 3Ch)	418
9.5.3.12	Runtime Link Testing Parity Log 2 (Offset 44h)	418
9.5.3.13	Runtime Link Testing Parity Log 3 (Offset 4Ch)	419
9.5.3.14	Advertised Adapter Capability Log (Offset 54h)	419
9.5.3.15	Finalized Adapter Capability Log (Offset 5Ch).....	419
9.5.3.16	Advertised CXL Capability Log (Offset 64h).....	419
9.5.3.17	Finalized CXL Capability Log (Offset 6Ch)	419
9.5.3.18	Advertised Multi-Protocol Capability Log Register (Offset 78h)....	420
9.5.3.19	Finalized Multi-Protocol Capability Log Register (Offset 80h)	420
9.5.3.20	Advertised CXL Capability Log Register for Stack 1 (Offset 88h) .	420
9.5.3.21	Finalized CXL Capability Log Register for Stack 1 (Offset 90h)....	420
9.5.3.22	PHY Capability (Offset 1000h).....	421
9.5.3.23	PHY Control (Offset 1004h)	422
9.5.3.24	PHY Status (Offset 1008h)	423
9.5.3.25	PHY Initialization and Debug (Offset 100Ch)	424
9.5.3.26	Training Setup 1 (Offset 1010h).....	425
9.5.3.27	Training Setup 2 (Offset 1020h).....	425
9.5.3.28	Training Setup 3 (Offset 1030h).....	426
9.5.3.29	Training Setup 4 (Offset 1050h).....	426
9.5.3.30	Current Lane Map Module 0 (Offset 1060h)	427
9.5.3.31	Current Lane Map Module 1 (Offset 1068h)	427
9.5.3.32	Current Lane Map Module 2 (Offset 1070h)	427
9.5.3.33	Current Lane Map Module 3 (Offset 1078h)	427
9.5.3.34	Error Log 0 (Offset 1080h)	428
9.5.3.35	Error Log 1 (Offset 1090h)	429
9.5.3.36	Runtime Link Test Control (Offset 1100h).....	429
9.5.3.37	Runtime Link Test Status (Offset 1108h).....	431
9.5.3.38	Mainband Data Repair (Offset 110Ch)	431
9.5.3.39	Clock, Track, Valid and Sideband Repair (Offset 1134h)	433
9.5.3.40	PCIe Link Health Monitor (UHM) DVSEC	434
9.5.3.40.1	UHM Status (Offset Eh)	435
9.5.3.40.2	Eye Margin (Starting Offset 18h)	435
9.5.4	Test/Compliance Register Block.....	436
9.5.4.1	PCIe Register Block Header.....	437
9.5.4.2	D2D Adapter Test/Compliance Register Block Offset	437
9.5.4.3	PHY Test/Compliance Register Block Offset.....	437
9.5.4.4	D2D Adapter Test/Compliance Register Block.....	438
9.5.4.4.1	Adapter Compliance Control	438
9.5.4.4.2	Flit Tx Injection Control	439
9.5.4.4.3	Adapter Test Status (Offset 30h from D2DOFF).....	440
9.5.4.4.4	Link State Injection Control Stack 0 (Offset 34h from D2DOFF)	441
9.5.4.4.5	Link State Injection Control Stack 1 (Offset 38h from D2DOFF)	442
9.5.4.4.6	Retry Injection Control (Offset 40h from D2DOFF)..	442

9.5.4.5	PHY Test/Compliance Register Block	444
9.5.4.5.1	Physical Layer Compliance Control 1 (Offsets 000h, 400h, 800h, and C00h from PHYOFF)	444
9.5.4.5.2	Physical Layer Compliance Control 2 (Offsets 008h, 408h, 808h, and C08h from PHYOFF)	446
9.5.4.5.3	Physical Layer Compliance Status 1 (Offsets 010h, 410h, 810h, and C10h from PHYOFF)	447
9.5.4.5.4	Physical Layer Compliance Status 2 (Offsets 018h, 418h, 818h, and C18h from PHYOFF)	447
9.5.4.5.5	Physical Layer Compliance Status 3 (Offsets 020h, 420h, 820h, and C20h from PHYOFF)	448
9.5.5	Implementation Specific Register Blocks.....	448
9.6	PCIe Link Registers in Streaming Mode and System SW/FW Implications	449
9.7	MSI and MSI-X Capability in Hosts/Switches for PCIe Interrupt.....	450
9.8	PCIe Early Discovery Table (UEDT)	450
10.0	Interface Definitions.....	452
10.1	Raw Die-to-Die Interface (RDI)	452
10.1.1	Interface reset requirements.....	460
10.1.2	Interface clocking requirements	460
10.1.3	Dynamic clock gating.....	461
10.1.3.1	Rules and description for lp_wake_req/pl_wake_ack handshake	461
10.1.3.2	Rules and description for pl_clk_req/lp_clk_ack handshake.....	461
10.1.4	Data Transfer	463
10.1.5	RDI State Machine.....	465
10.1.6	RDI bring up flow	466
10.1.7	RDI PM flow.....	467
10.2	Flit-Aware Die-to-Die Interface (FDI).....	471
10.2.1	Interface reset requirements.....	479
10.2.2	Interface clocking requirements	479
10.2.3	Dynamic clock gating.....	479
10.2.3.1	Rules and description for lp_wake_req/pl_wake_ack handshake	480
10.2.3.2	Rules and description for pl_clk_req/lp_clk_ack handshake.....	480
10.2.4	Data Transfer	481
10.2.4.1	DLLP transfer rules for 256B Flit Mode	482
10.2.5	Examples of pl_flit_cancel Timing Relationships.....	483
10.2.6	FDI State Machine.....	484
10.2.7	Rx_active_req/Sts Handshake.....	484
10.2.8	FDI Bring up flow	485
10.2.9	FDI PM Flow	487
10.3	Common rules for FDI and RDI.....	491
10.3.1	Byte Mapping for FDI and RDI.....	491
10.3.2	Stallreq/Ack Mechanism	495
10.3.3	State Request and Status	496
10.3.3.1	Reset State rules	497
10.3.3.2	Active State rules.....	498
10.3.3.3	PM Entry/Exit Rules.....	498
10.3.3.4	Retrain State Rules	498
10.3.3.5	LinkReset State Rules	499
10.3.3.6	Disabled State Rules.....	500
10.3.3.7	LinkError State Rules.....	500
10.3.3.8	Common State Rules	501
10.3.4	Example Flow Diagrams	504
10.3.4.1	LinkReset Entry and Exit	504

10.3.4.2	LinkError	505
10.3.4.3	Example of L2 Cross Product with Retrain on RDI	506
10.3.4.4	L2 Exit Example for RDI	507
11.0	Compliance.....	508
11.1	Protocol Layer Compliance	509
11.2	Adapter Compliance.....	509
11.3	PHY Compliance	510
A	CXL/PCIe Register applicability to UCIe	511
A.1	CXL Registers applicability to UCIe	511
A.2	PCIe Register applicability to UCIe	511
A.3	PCIe/CXL registers that need to be part of D2D	513
B	AIB Interoperability	514
B.1	AIB Signal Mapping	514
B.1.1	Data path.....	514
B.1.2	Always high Valid	514
B.1.3	Sideband.....	514
B.1.4	Raw Die-to-Die interface	515
B.2	Initialization.....	516
B.3	Bump Map	516

Figures

1-1	A Package Composed of CPU Dies, Accelerator Die(s), and I/O Tile Die Connected through UCIe	36
1-2	UCIe enabling long-reach connectivity at Rack/Pod Level	37
1-3	Standard Package interface	37
1-4	Advanced Package interface: Example 1	38
1-5	Advanced Package interface: Example 2	38
1-6	Advanced Package interface: Example 3	38
1-7	Example of UCIe-3D	39
1-8	UCIe Layers and functionalities	40
1-9	Physical Layer components	41
1-10	Single module configuration: Advanced Package	42
1-11	Single module configuration: Standard Package	43
1-12	Two-module configuration for Standard Package	43
1-13	Four-module configuration for Standard Package	43
1-14	Example of a Two-module Configuration for Advanced Package	44
1-15	One-port Sideband-only Link	44
1-16	Two-port Sideband-only Link	44
1-17	Four-port Sideband-only Link.....	45
1-18	Block Diagram for UCIe Retimer Connection	45
2-1	Color-coding Convention in Flit Format Byte Map Figures	50
2-2	68B Flit Format on FDI	54
3-1	Functionalities in the Die-to-Die Adapter	56
3-2	Example Configurations	59
3-3	Stages of UCIe Link Initialization	60
3-4	Parameter Exchange for CXL or PCIe (i.e., "68B Flit Mode" or "CXL 256B Flit Mode" is 1 in {FinCap.Adapter})	64
3-5	Both Stacks are CXL or PCIe.....	66
3-6	Stack 0 is PCIe, Stack 1 is Streaming	67
3-7	Stack 0 is Streaming, Stack 1 is PCIe	67
3-8	Both Stacks are Streaming	68
3-9	Stack 0 is Streaming, Stack 1 is CXL	69
3-10	Format 1: Raw Format	70
3-11	Format 2: 68B Flit Format	73
3-12	Format 2: 68B Flit Format PDS Example 1	74
3-13	Format 2: 68B Flit Format PDS Example 2 — Extra 0s Padded to Make the Data Transfer a Multiple of 256B	74
3-14	Format 3: Standard 256B End Header Flit Format for PCIe	76
3-15	Format 3: Standard 256B End Header Flit Format for Streaming Protocol	76
3-16	Format 3: Standard 256B End Header Flit Format for Management Transport Protocol	76
3-17	Format 4: Standard 256B Start Header Flit Format for CXL.cachemem or Streaming Protocol	77
3-18	Format 4: Standard 256B Start Header Flit Format for CXL.io or PCIe	77
3-19	Format 4: Standard 256B Start Header Flit Format for Management Transport Protocol	77
3-20	Format 5: Latency-Optimized 256B without Optional Bytes Flit Format for CXL.io	80
3-21	Format 5: Latency-Optimized 256B without Optional Bytes Flit Format for CXL.cachemem and Streaming Protocol	81
3-22	Format 5: Latency-Optimized 256B without Optional Bytes Flit Format for Management Transport Protocol	81

3-23	Format 6: Latency-Optimized 256B with Optional Bytes Flit Format for CXL.io or PCIe	81
3-24	Format 6: Latency-Optimized 256B with Optional Bytes Flit Format for CXL.cachemem	82
3-25	Format 6: Latency-Optimized 256B with Optional Bytes Flit Format for Streaming Protocol	82
3-26	Format 6: Latency-Optimized 256B with Optional Bytes Flit Format for Management Transport Protocol	82
3-27	State Machine Hierarchy Examples	88
3-28	Example of Hierarchical PM Entry for CXL	90
3-29	Diagram of CRC Calculation	91
3-30	Successful Parity Feature negotiation between Die 1 Tx and Die 0 Rx	95
3-31	Unsuccessful Parity Feature negotiation between Die 1 Tx and Die 0 Rx	96
4-1	Bit arrangement within a byte transfer	97
4-2	Byte map for x64 interface	98
4-3	Byte map for x32 interface	98
4-4	Byte map for x16 interface	98
4-5	Byte to Lane mapping for Standard package x16 degraded to x8	99
4-6	Valid framing example	100
4-7	Clock gating	100
4-8	Example 64-bit Sideband Serial Packet Transfer	101
4-9	Sideband Packet Transmission: Back-to-Back	101
4-10	Example 64-bit Sideband Serial Packet Transfer in Sideband Performant Mode	102
4-11	Sideband Packet Transmission: Back-to-Back in Sideband Performant Mode	102
4-12	Data Lane remapping possibilities to fix potential defects	105
4-13	Data Lane remapping: Mux chain	106
4-14	Example of Single Lane failure remapping	108
4-15	Example of Single Lane remapping implementation	108
4-16	Example of Two Lane failure remapping	110
4-17	Example of Two Lane remapping implementation	110
4-18	Clock and Track repair	114
4-19	Clock and track repair: Differential Rx	114
4-20	Clock and track repair: Pseudo Differential Rx	114
4-21	Clock and Track Lane repair scheme	115
4-22	Clock and track repair: CKP repair	115
4-23	Clock and track repair: CKN repair	116
4-24	Clock and track repair: Track repair	116
4-25	Valid repair: Normal Path	116
4-26	Valid Repair: Repair Path	117
4-27	Test and training logic	118
4-28	Lane failure detection	118
4-29	All Lane error detection	119
4-30	LFSR implementation	120
4-31	LFSR alternate implementation	121
4-32	Example Retimer bring up when performing speed/width match	128
4-33	Link Training State Machine	130
4-34	MBINIT: Mainband Initialization and Repair Flow	136
4-35	Example Sideband Management Transport Protocol Negotiation – Single-module Scenario	138
4-36	Example Sideband Management Transport Protocol Negotiation – Two-module Scenario	139
4-37	Example Sideband MPM Logical Flow with Two Modules and No Module Reversal ...	141
4-38	Example Sideband MPM Logical Flow with Two Modules and Module Reversal	141
4-39	MBINIT "Stall" Example 1	143
4-40	MBINIT "Stall" Example 2	144

4-41	Mainband Training.....	155
4-42	Example of Byte Mapping for Matching Module IDs	171
4-43	Example of Byte Mapping for Differing Module IDs	171
4-44	Example of Width Degradation with Byte Mapping for Differing Module IDs	172
4-45	Example of Byte Mapping with Module Disable	172
4-46	Decision Flow Chart for Multi-module Advanced Package	173
4-47	Decision Flow Chart for Multi-module Standard Package	174
4-48	Implementation Example Showing Two Different Operating Modes of the Same Hardware Implementation	177
4-49	RDI Byte-to-Module Assignment Example for x64 Interop with x32	178
4-50	Example of Encapsulated MTPs Transmitted on Sideband Link without Sideband PMO	179
4-51	Example of a Large Management Packet Split into Two Encapsulated MTPs, with No Segmentation, No Sideband PMO, and with Two Link Management Packets between the Two Encapsulated MTPs	179
5-1	Example Common Reference Clock	181
5-2	x64 or x32 Advanced Package Module	183
5-3	x16 or x8 Standard Package Module	183
5-4	Transmitter	185
5-5	Transmitter driver example circuit	186
5-6	Transmitter de-emphasis	188
5-7	Transmitter de-emphasis waveform.....	188
5-8	Receiver topology	189
5-9	Receiver Termination Map for Table 5-6 (Tx Swing = 0.4 V)	191
5-10	Receiver termination	191
5-11	Receiver termination map for Table 5-7 (TX Swing = 0.85 V)	192
5-12	Example CTLE	193
5-13	Clocking architecture	194
5-14	Track Usage Example	195
5-15	Example Eye diagram	197
5-16	Example Eye Simulation Setup.....	198
5-17	Circuit for VTF calculation	199
5-18	Loss and Crosstalk Mask	201
5-19	Viewer Orientation Looking at the Defined UCIe Bump Matrix	201
5-20	10-column x64 Advanced Package Bump Map	203
5-21	16-column x64 Advanced Package Bump Map	204
5-22	8-column x64 Advanced Package Bump Map	205
5-23	10-column x64 Advanced Package Bump map: Signal exit order	206
5-24	10-column x64 Advanced Package Bump Map Example for 32 GT/s Implementation	207
5-25	16-column x64 Advanced Package Bump Map Example for 16 GT/s Implementation	208
5-26	8-column x64 Advanced Package Bump Map Example for 32 GT/s Implementation	209
5-27	10-column x32 Advanced Package Bump Map	211
5-28	16-column x32 Advanced Package Bump Map	211
5-29	8-column x32 Advanced Package Bump Map	212
5-30	10-column x32 Advanced Package Bump Map: Signal Exit Order	212
5-31	10-column x32 Advanced Package Bump Map Example for 32 GT/s Implementation	213
5-32	16-column x32 Advanced Package Bump Map Example for 16 GT/s Implementation	214
5-33	8-column x32 Advanced Package Bump Map Example for 32 GT/s Implementation	215

5-34	Example of Normal and Mirrored x64-to-x32 Advanced Package Module Connection	218
5-35	Example of Normal and Mirrored x32-to-x32 Advanced Package Module Connection	219
5-36	Naming Convention for One-, Two-, and Four-module Advanced Package Paired with "Standard Die Rotate" Configurations	220
5-37	Naming Convention for One-, Two-, and Four-module Advanced Package Paired with "Mirrored Die Rotate" Configurations	220
5-38	Examples for Advanced Package Configurations Paired with "Standard Die Rotate" Counterparts, with a Different Number of Modules	221
5-39	Examples for Advanced Package Configurations Paired with "Mirrored Die Rotate" Counterparts, with a Different Number of Modules	222
5-40	Standard Package Bump Map: x16 interface	225
5-41	Standard Package x16 interface: Signal exit order	225
5-42	Standard Package Bump Map: x32 interface	225
5-45	Standard Package reference configuration	226
5-43	Standard Package x32 interface: Signal exit routing	226
5-44	Standard Package cross section for stacked module	226
5-46	Standard Package Bump Map: x8 Interface	227
5-47	Naming Convention for One-, Two-, and Four-module Standard Package Paired with "Standard Die Rotate" Configurations	228
5-48	Naming Convention for One-, Two-, and Four-module Standard Package Paired with "Mirrored Die Rotate" Configurations	229
5-49	Examples for Standard Package Configurations Paired with "Standard Die Rotate" Counterparts, with a Different Number of Modules	230
5-50	Examples for Standard Package Configurations Paired with "Mirrored Die Rotate" Counterparts, with a Different Number of Modules	231
5-51	Additional Examples for Standard Package Configurations Paired with "Mirrored Die Rotate" Counterparts, with a Different Number of Modules	231
5-52	Example of a Configuration for Standard Package, with Some Modules Disabled ..	233
5-53	UCIE-S Sideband-only Port Bump Map	234
5-54	UCIE-S Sideband-only Port Supported Configurations	234
5-55	Data Lane repair resources	236
5-56	Data Lane repair	236
5-57	Valid Framing	237
5-58	Data, Clock, Valid Levels for Half-rate Clocking: Clock-gated Unterminated Link	238
5-59	Data, Clock, Valid Levels for Quarter-rate Clocking: Clock-gated Unterminated Link	238
5-60	Data, Clock, Valid Levels for Half-rate Clocking: Continuous Clock Unterminated Link	238
5-61	Data, Clock, Valid Gated Levels for Half-rate Clocking: Terminated Link	239
5-62	Data, Clock, Valid Gated Levels for Quarter-rate Clocking: Terminated Link	239
5-63	Data, Clock, Valid Gated Levels for Half-rate Clocking: Continuous Clock Terminated Link	239
5-64	Sideband signaling	240
6-1	Example of 3D Die Stacking.....	242
6-2	UCIE-3D Illustration	243
6-3	UCIE-3D PHY.....	244
6-4	Start Edge and Sample Edge	245
6-5	Dtx and Drx Spec Range for 4 GT/s	247
6-6	UCIE-3D Module Bump Map.....	248
6-7	x70 Module	249
6-8	Bundle Repair	250

7-1	Format for Register Access Request	257
7-2	Format for Register Access Completions	258
7-3	Format for Messages without Data	259
7-4	Format for Messages with data payloads	266
7-5	Common Fields in MPM Header of all MPM with Data Messages on Sideband	274
7-6	Encapsulated MTP on Sideband	275
7-7	Vendor-defined Management Port Gateway Message with Data on Sideband	276
7-8	Common Fields in MPM Header of all MPM without Data Messages on Sideband	277
7-9	Management Port Gateway Capabilities MPM on Sideband	277
7-10	Credit Return MPM on Sideband	278
7-11	Init Done MPM on Sideband	279
7-12	PM MPM on Sideband	279
7-13	Vendor-defined Management Port Gateway MPM without Data on Sideband	280
7-14	Example Flow for Remote Register Access Request (Local FDI/RDI Credit Checks Are Not Explicitly Shown)	282
8-1	Example UCIe Chiplet that Supports Manageability	286
8-2	Example SiP that Supports Manageability	287
8-3	UCIe Manageability Protocol Hierarchy	288
8-4	Relationship Between the Various Types of Management Entities	289
8-5	UCIe Management Transport Packet	290
8-6	Management Network ID Format	294
8-7	Access Control Determination in a Responder Management Entity	300
8-8	Memory Map for Management Entities	303
8-9	Management Capability Structure Organization	304
8-10	Vendor Defined Management Capability Structure Organization	304
8-11	Management Capability Directory	305
8-12	Capability Pointer	305
8-13	Chiplet Capability Structure Organization	307
8-14	Chiplet Capability Structure	307
8-15	Management Port Structure	310
8-16	Route Entry	314
8-17	Access Control Capability Structure	316
8-18	Standard Asset Class Access Table	318
8-19	Vendor Defined Asset Class Access Table	319
8-20	Security Clearance Group Capability Structure	322
8-21	Security Clearance Group Context	323
8-22	UCIe Memory Access Request Packet Format	324
8-23	UCIe Memory Access Response Packet	328
8-24	UCIe Memory Access Protocol Capability Structure	329
8-25	UCIe Sideband Management Path Architecture	332
8-26	UCIe Mainband Management Path Architecture	333
8-27	Supported Configurations for Management Port Gateway Connectivity to D2D Adapter on Mainband	335
8-28	Common Fields in MPM Header of all MPM with Data Messages on Mainband	336
8-29	Encapsulated MTP on Mainband	337
8-30	Vendor-defined Management Port Gateway Message with Data on Mainband	338
8-31	Common Fields in MPM Header of all MPM without Data Messages on Mainband	339
8-32	Management Port Gateway Capabilities MPM on Mainband	339
8-33	Init Done MPM on Mainband	340
8-34	Vendor-defined Management Port Gateway Message without Data on Mainband	340
8-35	Sideband Management Transport Initialization Phase Example with RxQ-ID=0 and One VC (VC0)	343
8-36	Sideband Management Transport Initialization Phase Example with RxQ-ID=0, 1 and One VC (VC0)	344

8-37	Sideband Management Transport Initialization Phase Example with RxQ-ID=0 and Two VCs (VC0, VC1)	345
8-38	Mainband Management Transport Initialization Phase Example with RxQ-ID=0 and One VC (VC0)	348
8-39	Mainband Management Transport Initialization Phase Example with RxQ-ID=0, 1 and One VC (VC0)	348
8-40	Mainband Management Transport Initialization Phase Example with RxQ-ID=0 and Two VCs (VC0, VC1)	349
8-41	Example Illustration of a Large MTP Transmitted over Multiple RxQ-IDs on Sideband with Segmentation	352
8-42	Conceptual Illustration of Sideband Multi-module Ordering with Three RxQs	354
8-43	Example Illustration of a Large MTP Split into Multiple Smaller Encapsulated-MTPs for Transport over Sideband, without Segmentation	356
8-44	Management Flit NOP Message on Mainband	358
8-45	Management Transport Credit Return DWORD (CRD) Format on Mainband	358
8-46	Valid MPM Header Start Locations for Various Flit Formats	360
8-47	Example Mapping of MPMs and NOPs in Flit of Format 3	361
8-48	Example Mapping of MPMs and NOPs in Flit of Format 5	361
8-49	Example MPM Mapping to Management Flit for Format 3 with MPM Rollover to Next Flit	362
8-50	UDA Overview in Each Chiplet – Illustration	364
8-51	Vendor-defined Test and Debug UDM	366
8-52	PCIe-based Chiplet Testing/Debugging at Sort	367
8-53	PCIe-based Testing of Chiplets in an SiP	368
8-54	PCIe-based System Testing/Debug	369
8-55	DMH/DMS Address Mapping	370
8-56	DMH Capability Register Map	371
8-57	Empty Spoke Register Map	374
8-58	Common DMS Registers for All Non-empty Spokes Register Map	374
8-59	DMS Register Map for PCIe Spoke Types	379
8-60	DMS Register Map for Vendor-defined Spoke Types	383
9-1	Software view Example with Root Ports and Endpoints	388
9-2	Software view Example with Switch and Endpoints	389
9-3	Software view Example of PCIe Endpoint	390
9-4	PCIe Link DVSEC	392
9-5	PCIe Link Health Monitor (UHM) DVSEC	434
9-6	PCIe Test/Compliance Register Block	436
10-1	Example configurations using RDI	453
10-2	Example Waveform Showing Handling of Level Transition	462
10-3	Data Transfer from Adapter to Physical Layer	463
10-4	lp_irdy asserting two cycles before lp_valid	464
10-5	lp_irdy asserting at the same cycle as lp_valid	464
10-6	RDI State Machine	465
10-7	Example flow of Link bring up on RDI	466
10-8	Successful PM entry flow	469
10-9	PM Abort flow	469
10-10	PM Exit flow	470
10-11	RDI PM Exit Example Showing Interactions with LTSI	470
10-12	Example configurations using FDI	471
10-13	Example Waveform Showing Handling of Level Transition	481
10-14	Data Transfer from Protocol Layer to Adapter	482
10-15	Example for pl_flit_cancel for Latency-Optimized Flits and CRC Error on First Flit Half	483
10-16	Example for pl_flit_cancel for Latency-Optimized Flits and CRC Error on Second Flit Half	483

10-17	Example for pl_flit_cancel for Latency-Optimized Flits and CRC Error on Second Flit Half, Alternate Implementation Example	483
10-18	Example for pl_flit_cancel for Standard 256B Flits	484
10-19	FDI State Machine	484
10-20	FDI Bring up flow	486
10-21	PM Entry example for CXL or PCIe protocols	489
10-22	PM Entry example for symmetric protocol	489
10-23	PM Abort Example	490
10-24	PM Exit Example	490
10-25	CXL.io Standard 256B Start Header Flit Format Example	491
10-26	FDI (or RDI) Byte Mapping for 64B Datapath to 256B Flits.....	492
10-27	FDI (or RDI) Byte Mapping for 128B Datapath to 256B Flits	492
10-28	FDI Byte Mapping for 128B Datapath for 68B Flit Format.....	492
10-29	RDI Byte Mapping for 128B Datapath for 68B Flit Format.....	493
10-30	LinkReset Example	504
10-31	LinkError example	505
10-32	Example of L2 Cross Product with Retrain on RDI	506
10-33	L2 Exit Example for RDI	507
11-1	Examples of Standard and Advanced Package setups for DUT and Golden Die Compliance Testing	509
B-1	AIB interoperability	514

Tables

1	Terms and Definitions.....	25
2	Unit of Measure Symbols.....	33
3	Reference Documents	34
4	Revision History	34
1-1	Characteristics of UCIe on Standard Package	37
1-2	Characteristics of UCIe on Advanced Package	38
1-3	Characteristics of UCIe-3D	39
1-4	UCIe 2D and 2.5D Key Performance Targets	47
1-5	UCIe-3D Key Performance Targets	47
1-6	Groups for different bump pitches	48
2-1	Specification to protocol mode requirements.....	50
3-1	Capabilities that Must Be Negotiated between Link Partners	61
3-2	Flit Header for Format 2 without Retry	71
3-3	Flit Header for Format 2 with Retry	71
3-4	Flit Header for Format 3, Format 4, Format 5, and Format 6 without Retry	78
3-5	Flit Header for Format 3, Format 4, Format 5, and Format 6 with Retry	79
3-6	Summary of Flit Formats	83
3-7	Protocol Mapping and Implementation Requirements	84
3-8	Truth Table for Determining Protocol	85
3-9	Truth Table 1.....	87
3-10	Truth Table 2	87
4-1	Valid framing for Retimers	100
4-2	Lane ID: Advanced Package module	103
4-3	Lane ID: Standard Package Module	104
4-4	LFSR seed values	119
4-5	Data-to-Clock Training Parameters	124
4-6	State Definitions for Initialization	129
4-7	Per Lane ID Pattern	150
4-8	Per Lane ID Pattern Examples	151
4-9	Standard Package Logical Lane Map	154
4-10	Runtime Link Test Status Register based Retrain encoding	167
4-11	Retrain encoding	167
4-12	Retrain exit state resolution	167
4-13	Messages exchanged that are used to determine resolution for Example 1	175
4-14	Messages exchanged that are used to determine resolution for Example 2	176
4-15	Messages exchanged that are used to determine resolution for Example 3	176
4-16	Capability Register and Link Training Parameter Values for RDI Byte-to-Module Assignment Example for x64 Interop with x32	178
5-1	REFCLK Frequency PPMs and SSC PPMs	181
5-2	Electrical summary	184
5-3	Transmitter Electrical Parameters	186
5-4	Transmitter de-emphasis values	188
5-5	Receiver Electrical parameters	190
5-6	Maximum channel reach for unterminated Receiver (Tx Swing = 0.4 V)	191
5-7	Maximum Channel reach for unterminated Receiver (TX swing = 0.85V)	192
5-8	Forwarded clock frequency and phase	194
5-9	I/O Noise and Clock Skew	196
5-10	Eye requirements	197
5-11	Channel Characteristics.....	200

5-12	x64 Advanced Package Module Signal List	200
5-13	Bump Map Options and the Recommended Bump Pitch Range and Max Speed	206
5-14	Maximum Systematic Lane-to-lane Length Mismatch in um between the Reference Bump Maps in the Implementation Note	216
5-15	x64 and x32 Advanced Package Connectivity Matrix	217
5-16	Summary of Advanced Package Module Connection Combinations with Same Number of Modules on Both Sides	221
5-17	Summary of Advanced Package Module Connection Combinations with Different Number of Modules on Both Sides	222
5-18	IL and Crosstalk for Standard Package: With Receiver Termination Enabled	223
5-19	IL and Crosstalk for Standard Package: No Rx Termination	223
5-20	Standard Package Module Signal List	223
5-21	Summary of Standard Package Module Connection Combinations with Same Number of Modules on Both Sides	229
5-22	Summary of Standard Package Module Connection Combinations with Different Number of Modules on Both Sides	232
5-23	Summary of Degraded Links when Standard Package Module-pairs Fail	233
5-24	Tightly Coupled Mode: Eye Mask	235
5-25	Tightly Coupled Mode Channel for Advanced Package	235
5-26	Raw BER requirements	237
5-27	Sideband Parameters summary	240
5-28	AUXCLK Frequency Parameters	241
6-1	UCIE-3D Key Performance Indicators	243
6-2	Timing and Mismatch Specification	245
6-3	ESD Specification for ≤ 10 um Bump Pitch	248
6-4	Energy Efficiency Target	248
7-1	Opcode Encodings Mapped to Packet Types	253
7-2	FDI sideband: srcid and dstid encodings on FDI	254
7-3	RDI sideband: srcid and dstid encodings on RDI	254
7-4	UCIE Link sideband: srcid and dstid encodings for UCIE Link	255
7-5	Field descriptions for Register Access Requests	256
7-6	Mapping of Addr[23:0] for Different Requests	257
7-7	Field Descriptions for a Completion	258
7-8	Message Encodings for Messages without Data	259
7-9	Link Training State Machine related Message encodings for messages without data	262
7-10	Message encodings for Messages with Data	266
7-11	Link Training State Machine related encodings	268
7-12	Supported MPM with Data Messages on Sideband	274
7-13	Common Fields in MPM Header of all MPM with Data Messages on Sideband	274
7-14	Encapsulated MTP on Sideband Fields	275
7-15	Vendor-defined Management Port Gateway Message with Data on Sideband Fields	276
7-16	Supported MPM without Data Messages on Sideband	276
7-17	Common Fields in MPM Header of all MPM without Data Messages on Sideband	277
7-18	Management Port Gateway Capabilities MPM Header Fields on Sideband	277
7-19	Credit Return MPM Header Fields on Sideband	278
7-20	Init Done MPM Header Fields on Sideband	279
7-21	PM MPM Header Fields on Sideband	279
7-22	MPM Header Vendor-defined Management Port Gateway Message without Data on Sideband	280

8-1	UCIE Management Transport Packet Fields	290
8-2	Traffic Class Characteristics	292
8-3	Management Protocol IDs	294
8-4	Management Protocol use of Access Control Mechanism	298
8-5	Asset Types	301
8-6	Asset Contexts	301
8-7	Standard Security Asset Classes	301
8-8	UCIE-defined Management Capability IDs	304
8-9	Management Capability Directory Fields	305
8-10	Capability Pointer Fields	306
8-11	Chiplet Capability Structure Fields	308
8-12	Management Port Structure Fields	310
8-13	Route Entry Fields	314
8-14	Access Control Capability Structure Fields	316
8-15	Read Access Control (RAC) Structure Field Description	320
8-16	Write Access Control (WAC) Structure Field Description	321
8-17	Security Clearance Group Capability Structure Fields	322
8-18	Security Clearance Group Context Fields	323
8-19	UCIE Memory Access Request Packet Fields	325
8-20	UCIE Memory Access Response Packet Fields	328
8-21	UCIE Memory Access Protocol Capability Structure Fields	329
8-22	MPM Opcodes on Mainband	336
8-23	Supported MPM with Data Messages on Mainband	336
8-24	Common Fields in MPM Header of all MPM with Data Messages on Mainband	336
8-25	Encapsulated MTP on Mainband Fields.....	337
8-26	Vendor-defined Management Port Gateway Message with Data on Mainband Fields	338
8-27	Supported MPM without Data Messages on Mainband	338
8-28	Common Fields in MPM Header of all MPM without Data Messages on Mainband....	339
8-29	Management Port Gateway Capabilities MPM Header Fields on Mainband	339
8-30	Init Done MPM Header Fields on Mainband.....	340
8-31	MPM Header Vendor-defined Management Port Gateway Message without Data on Mainband.....	341
8-32	Version.....	371
8-33	Capability ID, Ver.....	371
8-34	Debug Capability.....	372
8-35	Debug Control	372
8-36	Debug Status	372
8-37	DMH_Length_Low	372
8-38	DMH_Length_High	372
8-39	DMH Extended Capability Pointer Low	373
8-40	DMH Extended Capability Pointer High	373
8-41	DMS_Start_Low.....	373
8-42	DMS_Start_High	373
8-43	DMS_Next_Low Address	374
8-44	DMS_Next_High Address.....	374
8-45	Spoke Vendor ID.....	375
8-46	Spoke Device ID	375
8-47	DMS_Next_Low Address	376
8-48	DMS_Next_High Address.....	376
8-49	Spoke Revision ID	376
8-50	DMS-ID	377
8-51	Associated DMS-ID[0, 1, 2]	377
8-52	Spoke Capability	377

8-53	Spoke Control.....	377
8-54	Spoke Status.....	378
8-55	DMS Register Space Length Low	378
8-56	DMS Register Space Length High.....	378
8-57	DMS Extended Capability Pointer Low	378
8-58	DMS Extended Capability Pointer High	378
8-59	Port ID	380
8-60	Adapter_Physical_Layer_Ptr_Low	380
8-61	Adapter_Physical_Layer_Ptr_High	380
8-62	Compliance_Test_Ptr_Low.....	381
8-63	Compliance_Test_Ptr_High	381
8-64	Impl_Spec_Adapter_Ptr_Low	382
8-65	Impl_Spec_Adapter_Ptr_High	382
8-66	Impl_Spec_Physical_Layer_Ptr_Low	382
8-67	Impl_Spec_Physical_Layer_Ptr_High.....	382
9-1	Software view of Upstream and Downstream Device at UCIE interface	385
9-2	SW discovery of UCIE Links	386
9-3	Summary of location of various UCIE Link related registers	387
9-4	Register Attributes	391
9-5	UCIE Link DVSEC - PCI Express Extended Capability Header	393
9-6	UCIE Link DVSEC - Designated Vendor Specific Header 1, 2	393
9-7	UCIE Link DVSEC - Capability Descriptor	394
9-8	UCIE Link DVSEC - UCIE Link Capability	395
9-9	UCIE Link DVSEC - UCIE Link Control.....	396
9-10	UCIE Link DVSEC - UCIE Link Status	398
9-11	UCIE Link DVSEC - Link Event Notification Control	401
9-12	UCIE Link DVSEC - Error Notification Control.....	402
9-13	UCIE Link DVSEC - Register Locator 0, 1, 2, 3 Low	405
9-14	UCIE Link DVSEC - Register Locator 0, 1, 2, 3 High	406
9-15	UCIE Link DVSEC - Sideband Mailbox Index Low.....	406
9-16	UCIE Link DVSEC - Sideband Mailbox Index High.....	407
9-17	UCIE Link DVSEC - Sideband Mailbox Data Low	407
9-18	UCIE Link DVSEC - Sideband Mailbox Data High	407
9-19	UCIE Link DVSEC - Sideband Mailbox Control.....	408
9-20	UCIE Link DVSEC - Sideband Mailbox Status	408
9-21	UCIE Link DVSEC - Requester ID	408
9-22	UCIE Link DVSEC - Associated Port Numbers	409
9-23	UiSRB DVSEC - PCI Express Extended Capability Header	409
9-24	UiSRB DVSEC - Designated Vendor Specific Header 1, 2	410
9-25	UiSRB DVSEC - UiSRB Base Address	410
9-26	D2D/PHY Register Block - UCIE Register Block Header (Offset 0h)	411
9-27	Uncorrectable Error Status Register	411
9-28	Uncorrectable Error Mask Register.....	412
9-29	Uncorrectable Error Severity Register	413
9-30	Correctable Error Status Register	413
9-31	Correctable Error Mask Register	414
9-32	Header Log 1 Register	414
9-33	Header Log 2 Register	415
9-34	Error and Link Testing Control Register	417
9-35	Runtime Link Testing Parity Log 0 Register	418
9-36	Runtime Link Testing Parity Log 1 Register	418
9-37	Runtime Link Testing Parity Log 2 Register	418
9-38	Runtime Link Testing Parity Log 3 Register	419
9-39	Advertised Adapter Capability Log Register	419

9-40	Finalized Adapter Capability Log Register.....	419
9-41	Advertised CXL Capability Log Register	419
9-42	Finalized CXL Capability Log Register	419
9-43	Advertised Multi-Protocol Capability Log Register.....	420
9-44	Finalized Multi-Protocol Capability Log Register	420
9-45	Advertised CXL Capability Log Register for Stack 1	420
9-46	Finalized CXL Capability Log Register for Stack 1	420
9-47	Physical Layer Capability Register.....	421
9-48	Physical Layer Control Register	422
9-49	Physical Layer Status Register	423
9-50	Phy Init and Debug Register	424
9-51	Training Setup 1 Register.....	425
9-52	Training Setup 2 Register.....	425
9-53	Training Setup 3 Register.....	426
9-54	Training Setup 4 Register.....	426
9-55	Current Lane Map Module 0 Register.....	427
9-56	Current Lane Map Module 1 Register.....	427
9-57	Current Lane Map Module 2 Register.....	427
9-58	Current Lane Map Module 3 Register.....	427
9-59	Error Log 0 Register	428
9-60	Error Log 1 Register	429
9-61	Runtime Link Test Control	429
9-62	Runtime Link Test Status Register.....	431
9-63	Mainband Data Repair Register	431
9-64	Clock, Track, Valid and Sideband Repair Register.....	433
9-65	UHM DVSEC - Designated Vendor Specific Header 1, 2 (Offsets 04h and 08h)	434
9-66	UHM Status	435
9-67	EML_Lnx_Mody	435
9-68	EMR_Lnx_Mody	435
9-69	UCIE Register Block Header (Offset 0h).....	437
9-70	D2D Adapter Test/Compliance Register Block Offset (Offset 10h).....	437
9-71	PHY Test/Compliance Register Block Offset (Offset 14h)	437
9-72	Adapter Compliance Control (Offset 20h from D2DOFF).....	438
9-73	Flit Tx Injection Control (Offset 28h from D2DOFF)	439
9-74	Adapter Test Status.....	440
9-75	Link State Injection Control Stack 0	441
9-76	Link State Injection Control Stack 1	442
9-77	Retry Injection Control	442
9-78	Physical Layer Compliance Control 1	444
9-79	Physical Layer Compliance Control 2	446
9-80	Physical Layer Compliance Status 1	447
9-81	Physical Layer Compliance Status 2	447
9-82	Physical Layer Compliance Status 3	448
9-83	UEDT Header	450
9-84	UCIE Link Structure (UCLS)	450
10-1	RDI signal list	453
10-2	RDI Config interface extensions for Management Transport	458
10-3	FDI signal list	472
10-4	Requests Considered in Each State by Lower Layer	497
A-1	CXL Registers for UCIE devices	511
A-2	PCIe Registers for UCIE devices	512
B-1	AIB 2.0 Datapath mapping for Advanced Package	515
B-2	AIB 1.0 Datapath mapping for Advanced Package	515

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 <i>CXL Specification</i>)
APMW	Advanced Package Module Width
ARB/MUX	Arbiter/Multiplexer (as defined in <i>CXL Specification</i>)
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 <i>CXL Specification</i> .
CXL 256B Flit Mode	This term is used to reference 256B Flit Mode related Protocol features defined in <i>CXL Specification</i> .
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 <i>PCIe Base Specification</i>)
DLP	In Flit modes, the Data Link Layer Payload within a flit (as defined in <i>PCIe Base Specification</i>).
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 PCIe registers and state machines associated with a given PCIe 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 <i>CXL Specification</i>)
DVFS	Dynamic Voltage Frequency Scaling
DVSEC	Designated Vendor-Specific Extended Capability (as defined in <i>PCIe Base Specification</i>)
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 PCIe link. In the case of an MTP, the resulting packet after Encapsulation is called the Encapsulated MTP.
Endpoint EP	As defined in <i>PCIe Base Specification</i> .
eRCD	Exclusive Restricted CXL Device (as defined in <i>CXL Specification</i>)
eRCH	Exclusive Restricted CXL Host (as defined in <i>CXL Specification</i>)
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 <i>CXL Specification</i>).
Flit_Marker FM	Flit Marker (as defined in <i>PCIe Base Specification</i>)
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
LTS defense	Link Training State Machine
LTSSM	Link Training and Status State Machine (as defined in <i>PCIe Base Specification</i>)
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 <i>PCIe Base Specification</i>)
PCIe Flit Mode	This term is used to reference Flit Mode related Protocol features defined in <i>PCIe Base Specification</i> .
PCIe non-Flit Mode	This term is used to reference non-Flit Mode related Protocol features defined in <i>PCIe Base Specification</i> .
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 <i>CXL Specification</i>)
RCH	Restricted CXL Host (as defined in <i>CXL Specification</i>)
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 <i>PCIe Base Specification</i> .
Root Port RP	As defined in <i>PCIe Base Specification</i> .
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 <i>PCIe Base Specification</i>)
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 <i>CXL Specification</i>)
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
<i>PCI Express® Base Specification (PCIe Base Specification) Revision 6.2</i>	www.pcisig.com
<i>Compute Express Link Specification (CXL Specification) Revision 3.1</i>	computeexpresslink.org
<i>ACPI Specification (version 6.5 or later)</i>	www.uefi.org
<i>Industry Council on ESD Targets white papers</i>	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	<ul style="list-style-type: none"> Chapter 6.0, UCIE-3D and related support for UCIE 3D Chapter 8.0, System Architecture and related support for: <ul style="list-style-type: none"> 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	<ul style="list-style-type: none"> 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.).

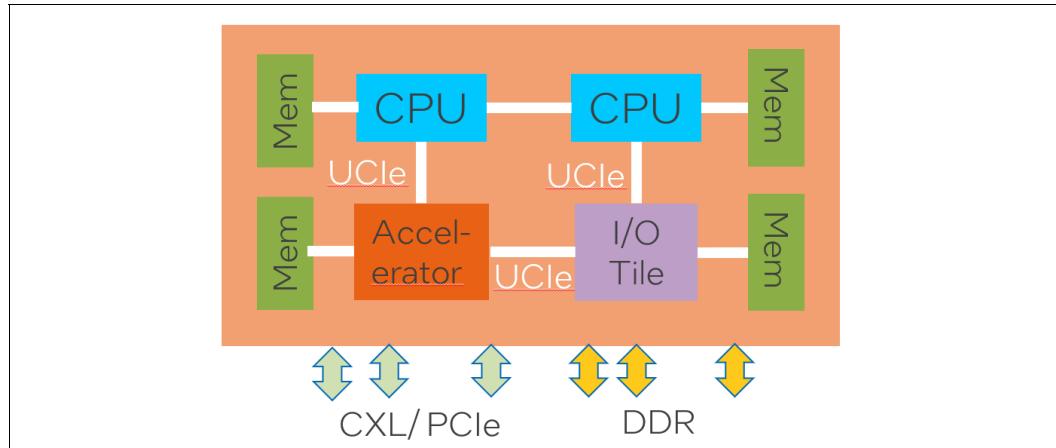
UCIE Manageability Architecture is an optional mechanism to manage a UCIE-based System-in-Package (SiP) by provisioning for a common manageability architecture and hardware/software infrastructure to be leveraged across implementations. UCIE DFx Architecture (UDA) leverages the UCIE Manageability Architecture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

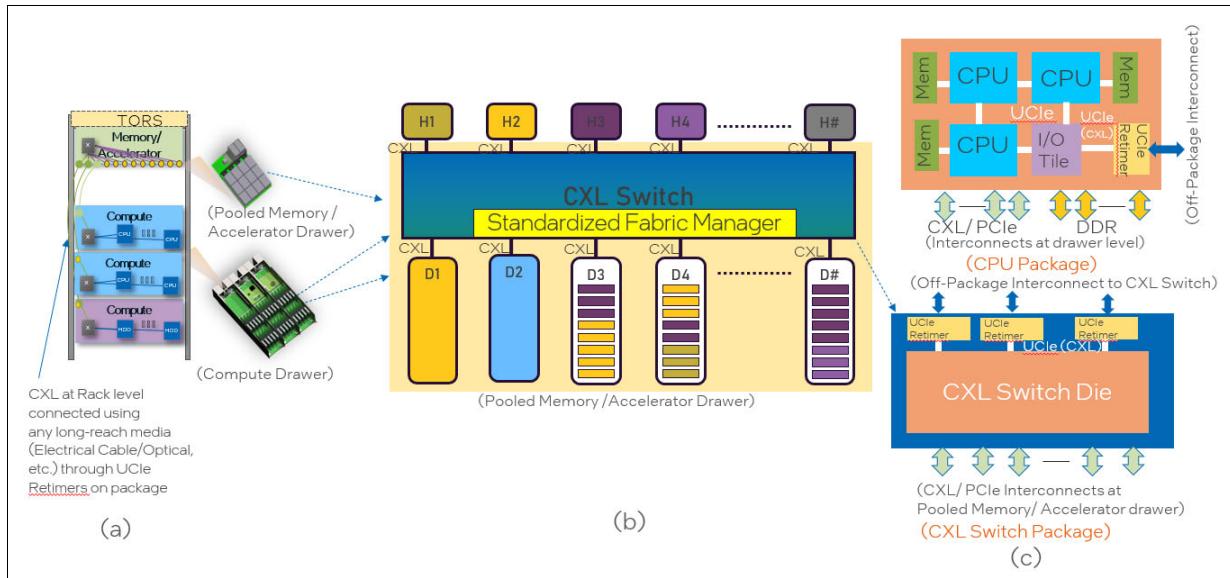
Figure 1-1 demonstrates an SoC package composed of CPU Dies, accelerator Die(s) and I/O Tile Die(s) connected through UCIE. The accelerator or I/O Tile can use CXL transactions over UCIE when connected to a CPU — leveraging the I/O, coherency, and memory protocols of CXL. The I/O tile can provide the 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



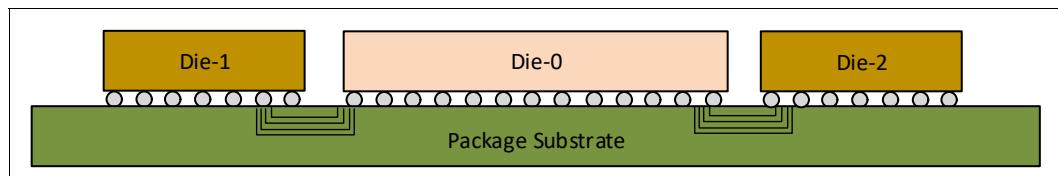
A UCIe Retimer may be used to extend the UCIe connectivity beyond the package using an Off-Package Interconnect. Examples of Off-Package Interconnect include electrical cable or optical cable or any other technology to connect packages at a Rack/Pod level as shown in [Figure 1-2](#). The UCIe specification 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

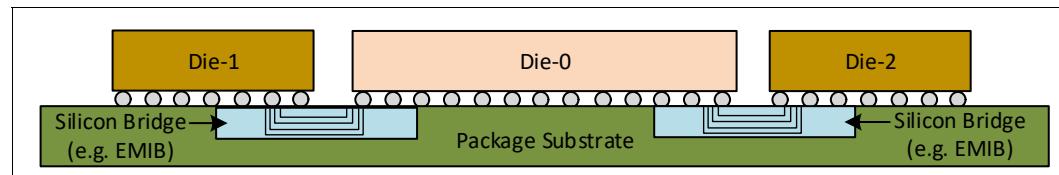


Figure 1-5. Advanced Package interface: Example 2

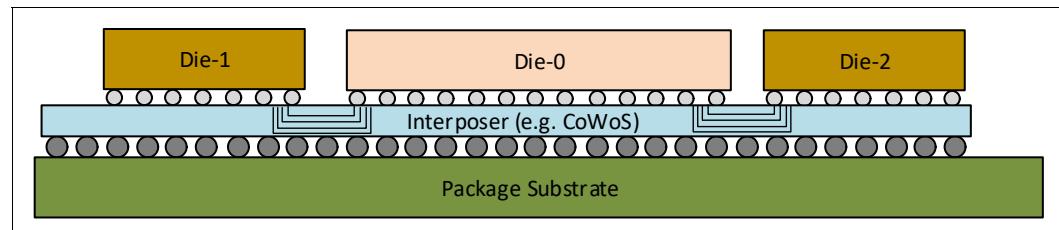
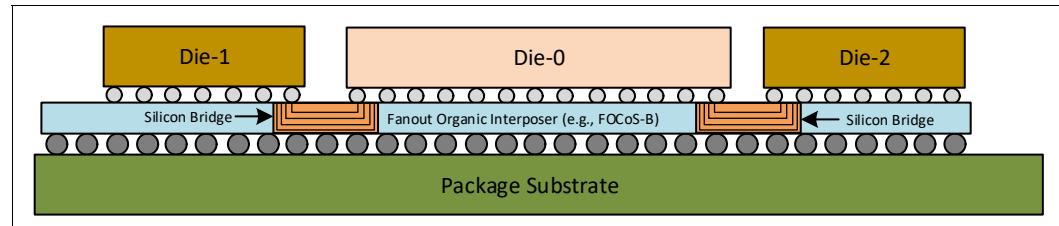


Figure 1-6. Advanced Package interface: Example 3



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.

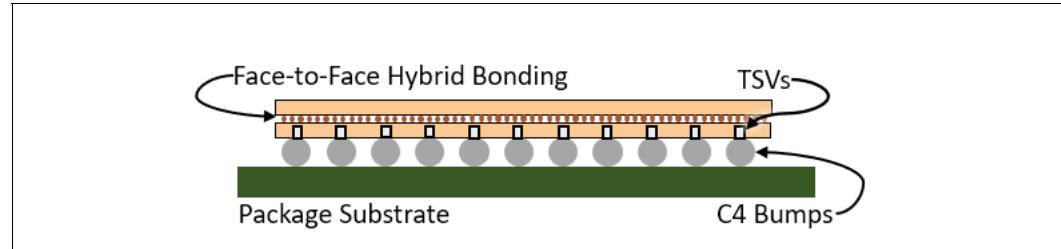
[Table 1-3](#) shows a summary of the main characteristics of UCIE-3D. [Figure 1-7](#) shows an example of UCIE-3D. See [Chapter 6.0](#) for a detailed description of UCIE-3D.

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.

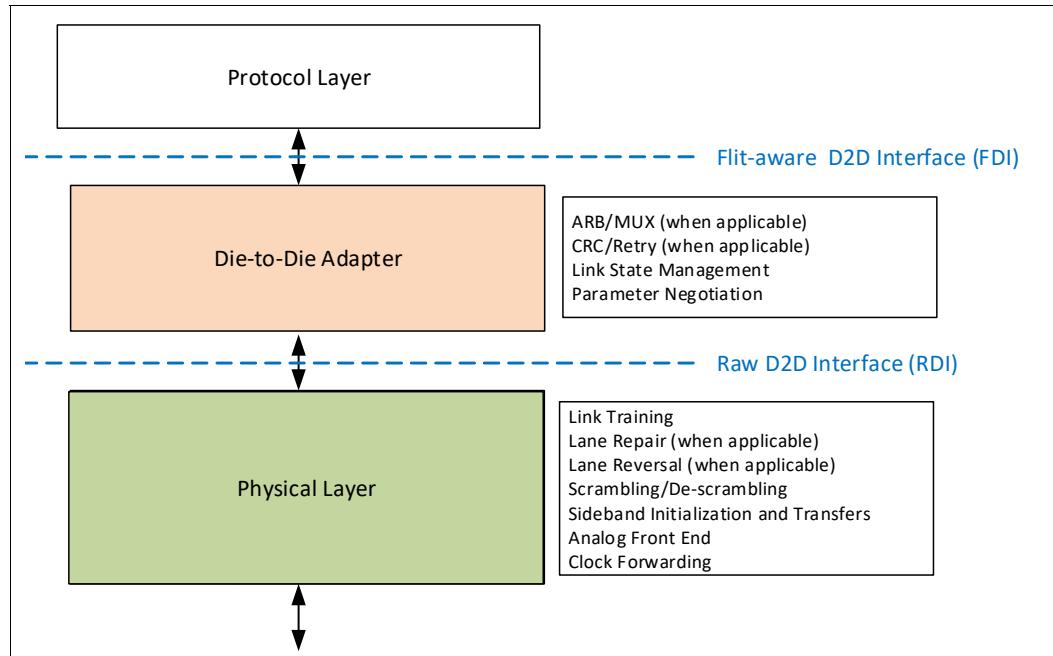
Figure 1-7. Example of UCIE-3D



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

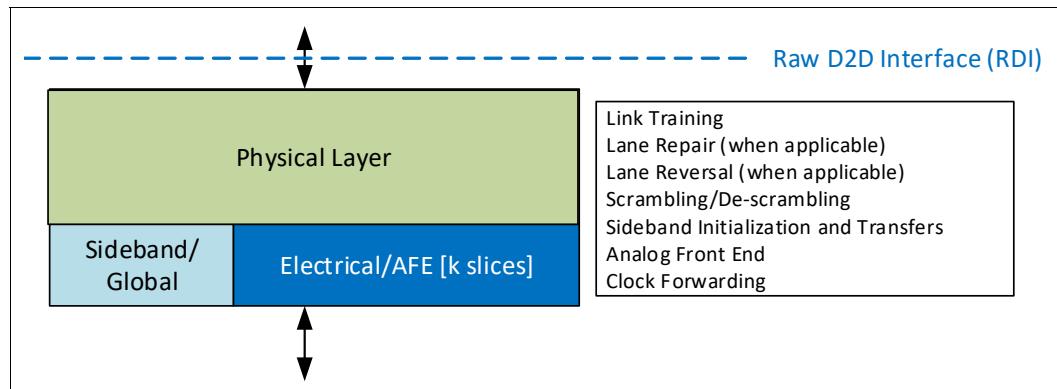
The D2D Adapter coordinates with the Protocol Layer and the Physical Layer to ensure successful data transfer across the UCIe Link. It minimizes logic on the main data path as much as possible, thus providing a low-latency, optimized data path for protocol Flits. When transporting CXL protocol, the ARB/MUX 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

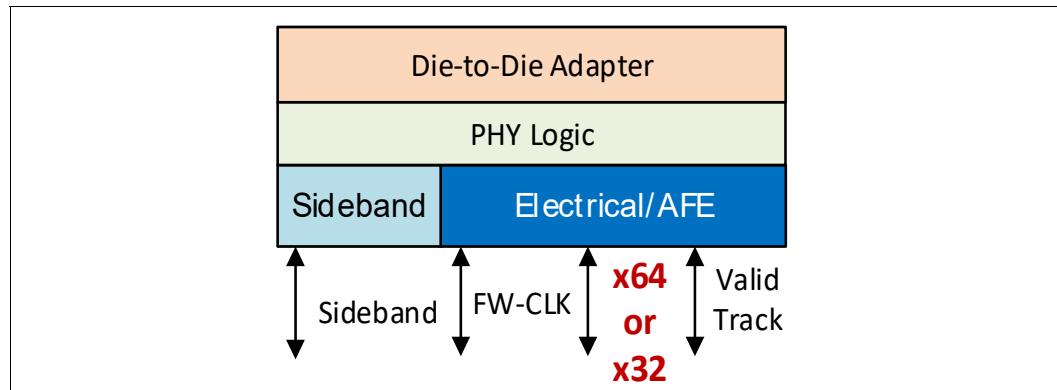
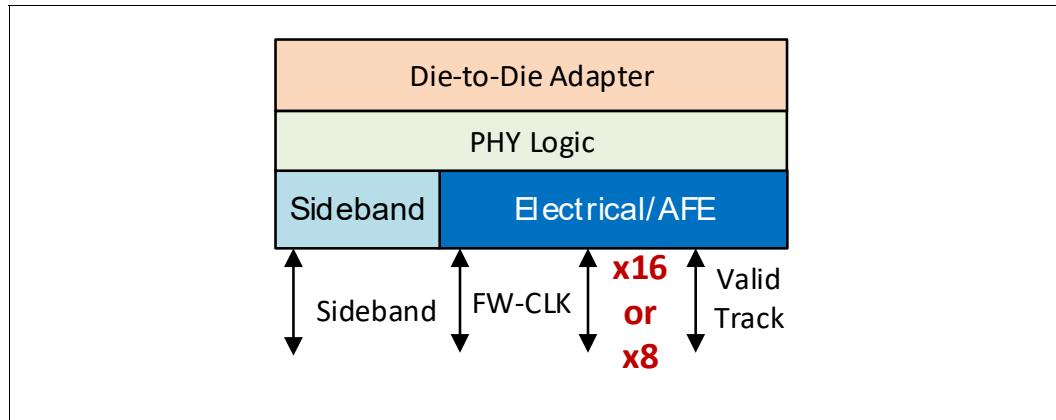


Figure 1-11. Single module configuration: Standard Package

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](#).

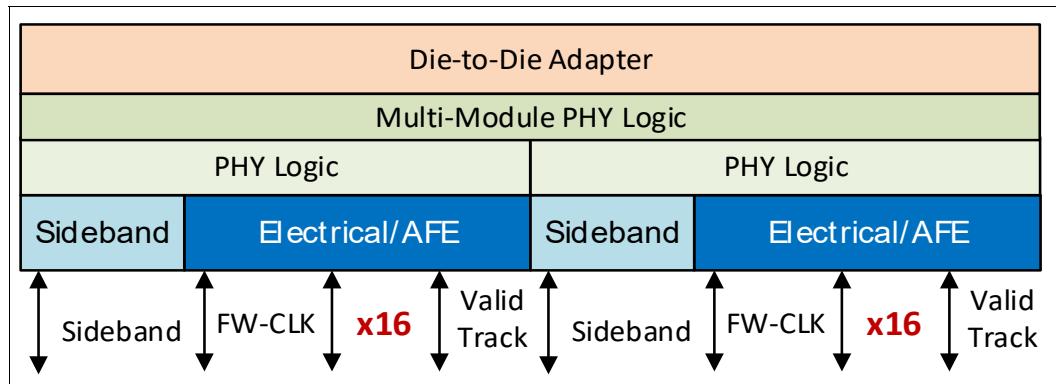
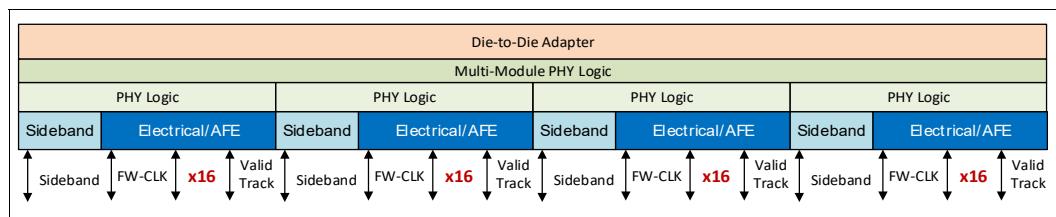
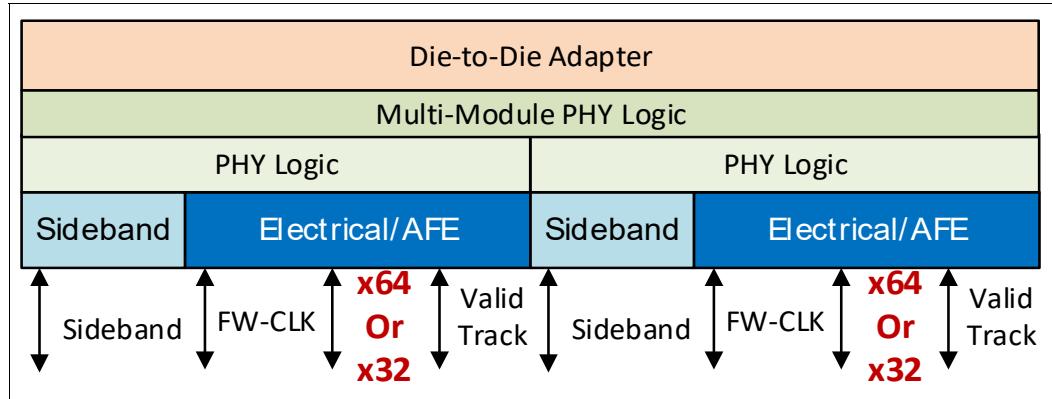
Figure 1-12. Two-module configuration for Standard Package**Figure 1-13. Four-module configuration for Standard Package**

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.

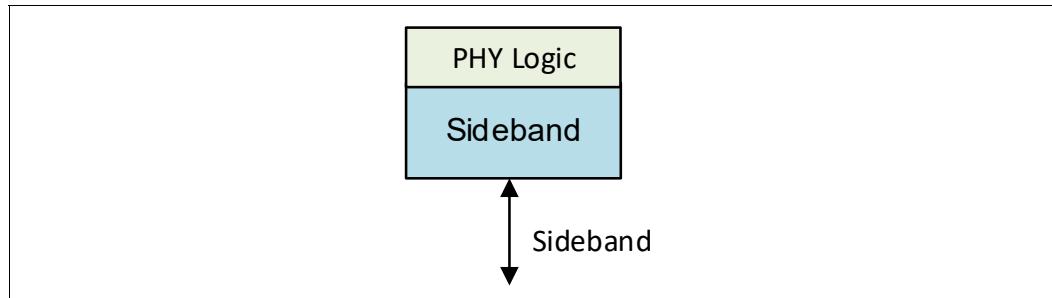
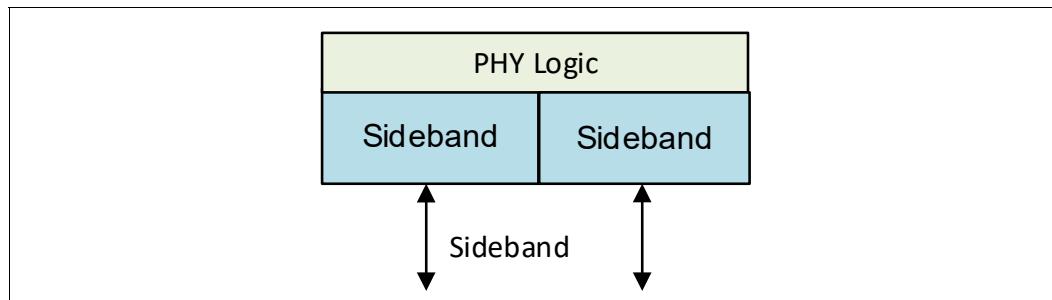
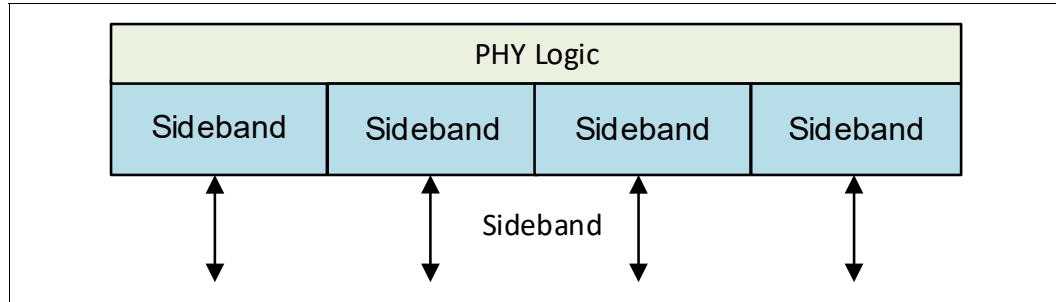
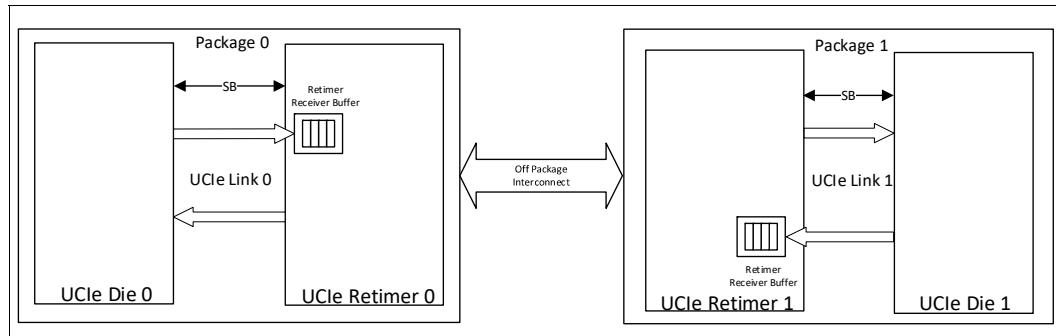
Figure 1-15. One-port Sideband-only Link**Figure 1-16.** Two-port Sideband-only Link

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 reassigned 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
Light Green	Some bits populated by the Protocol Layer, some bits populated by the Adapter.
Light Orange	All bits populated by Adapter.
Light Blue	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.

§ §

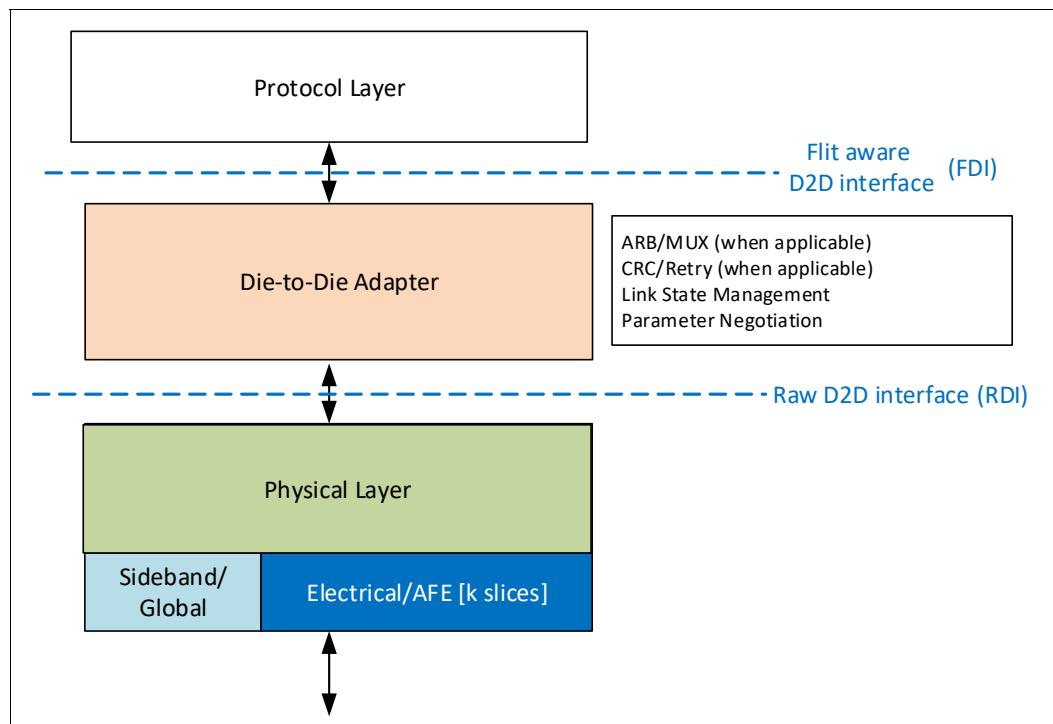
3.0 Die-to-Die Adapter

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.

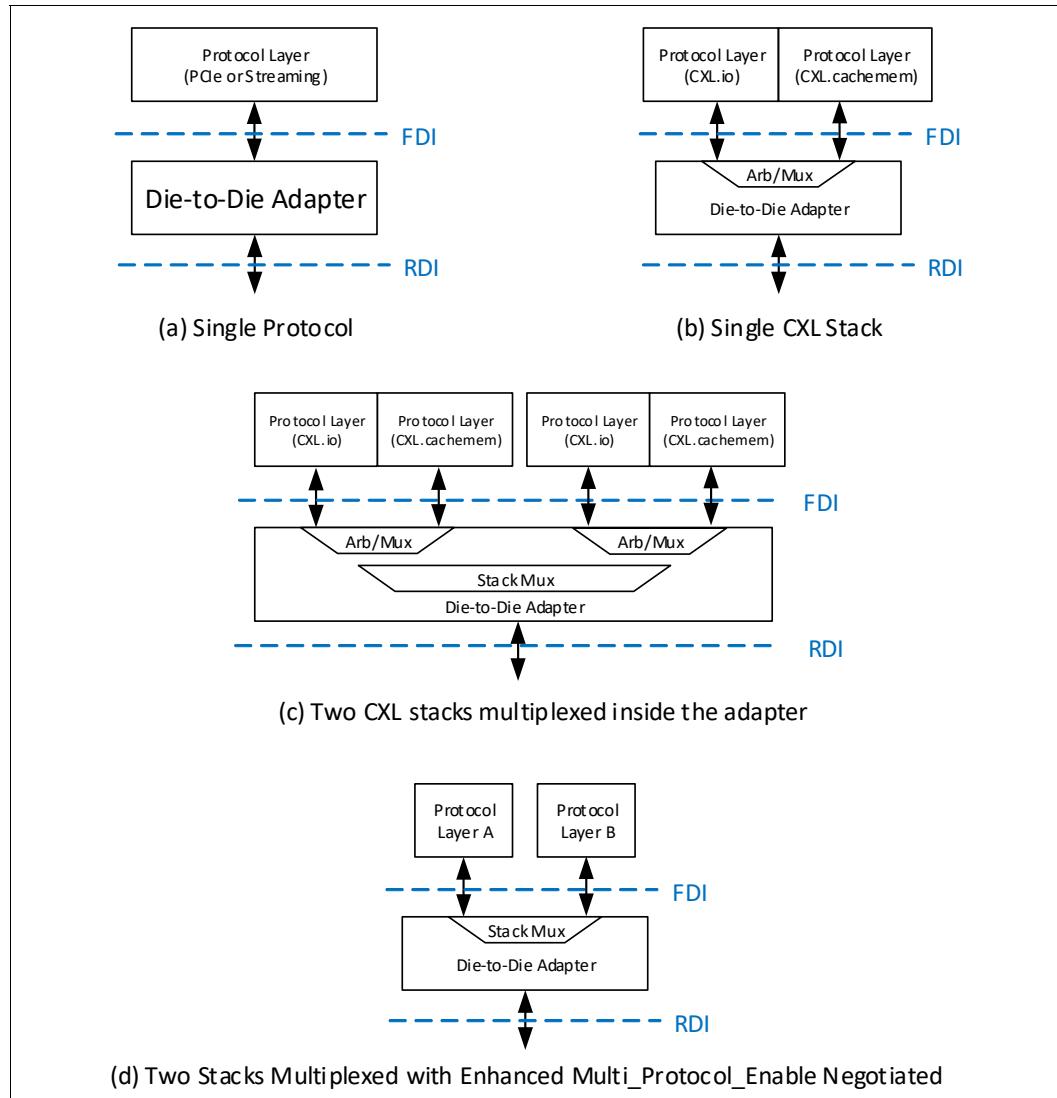
When Multi_Protocol_Enable is supported and negotiated, the Adapter must guarantee that it will not send consecutive flits from the same protocol stack on the Link. This applies in all cases including when Flits are sourced from FDI, from Retry Buffer, and when the data stream is paused and restarted. 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

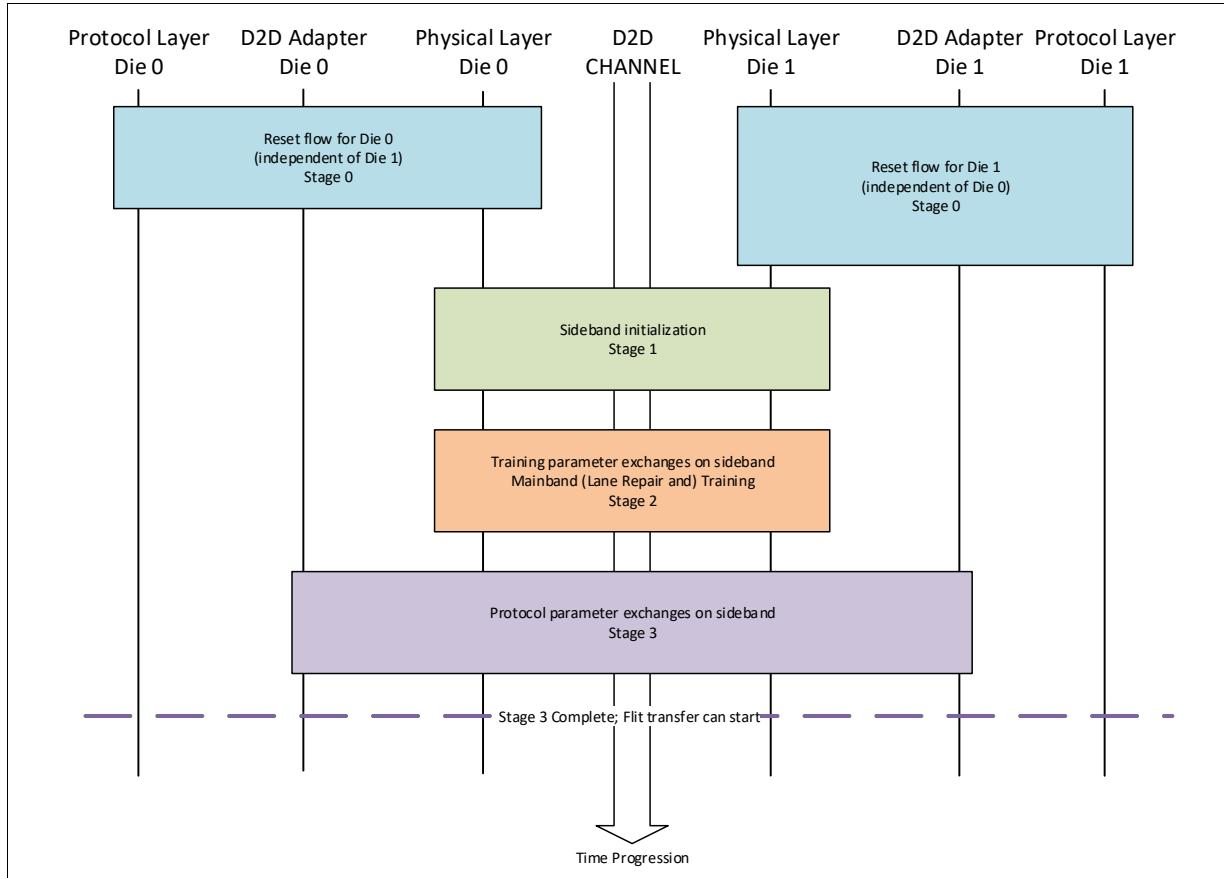
Multi_Protocol_Enable and Management Transport protocol are negotiated, each stack can have different protocols with or without MPG mux. For example, in [Figure 8-27](#), the Enhanced Multi_Protocol_Enable parameter must be negotiated for configs e, f, and h. The parameter is also negotiated for configs 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 <i>CXL Specification</i> . 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 <i>Format 5</i> 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 <i>Format 6</i> 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 PCIe 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 PCIe 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: <ul style="list-style-type: none"> 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: <ul style="list-style-type: none"> 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: <ul style="list-style-type: none"> 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: <ul style="list-style-type: none"> 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: <ul style="list-style-type: none"> 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.

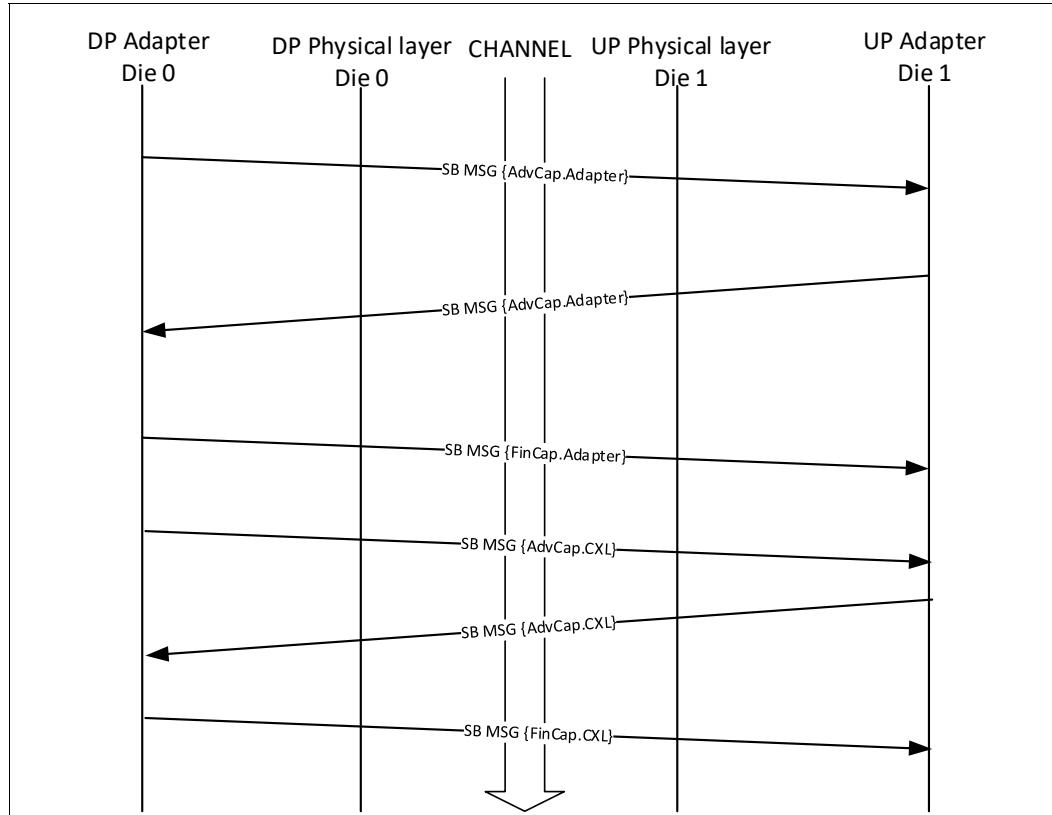
If “68B Flit Mode” or “CXL 256B Flit Mode” is set in the {FinCap.Adapter} message, there must be another handshake of Parameter Exchanges using the {AdvCap.CXL} and the {FinCap.CXL} messages to determine the details associated with this mode. Note that because CXL 68B Flit Mode protocol is mandatory 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.

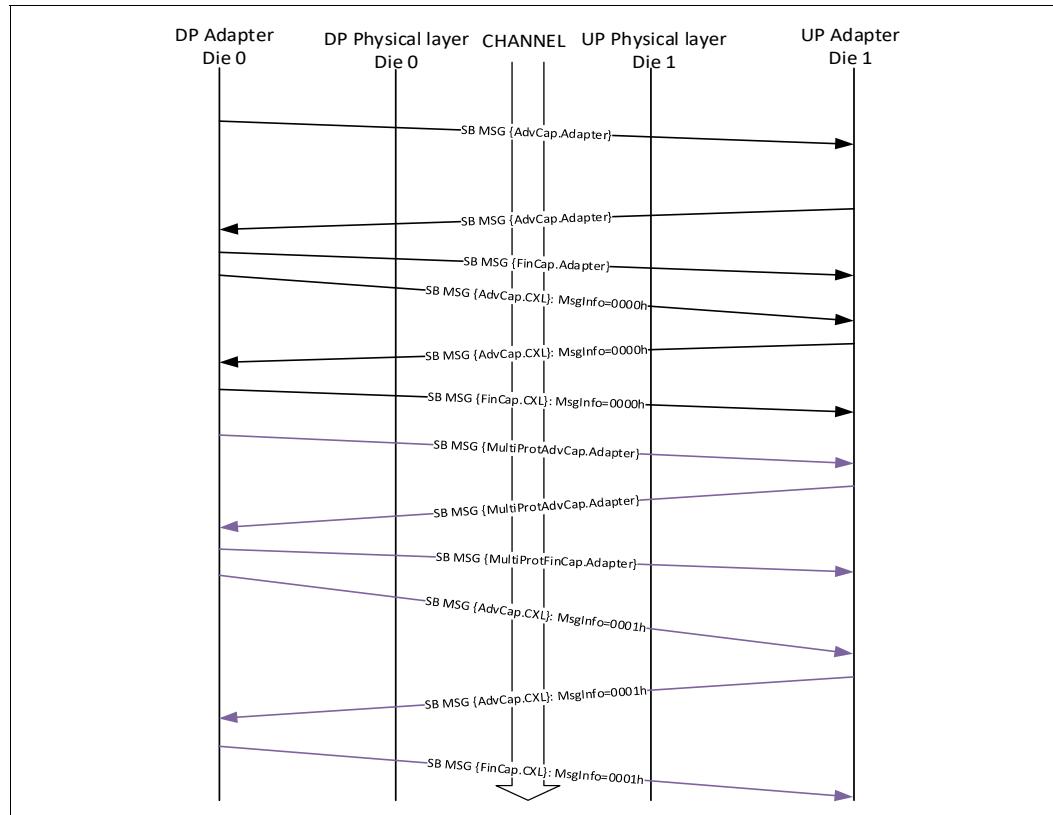
Figure 3-5. Both Stacks are CXL or PCIe

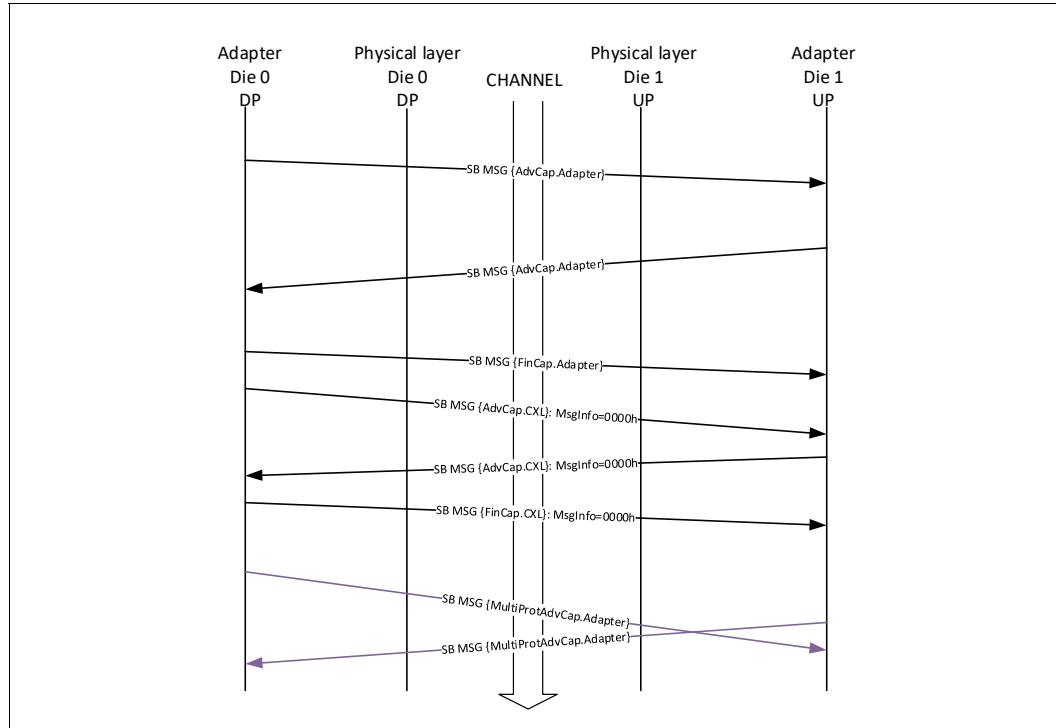
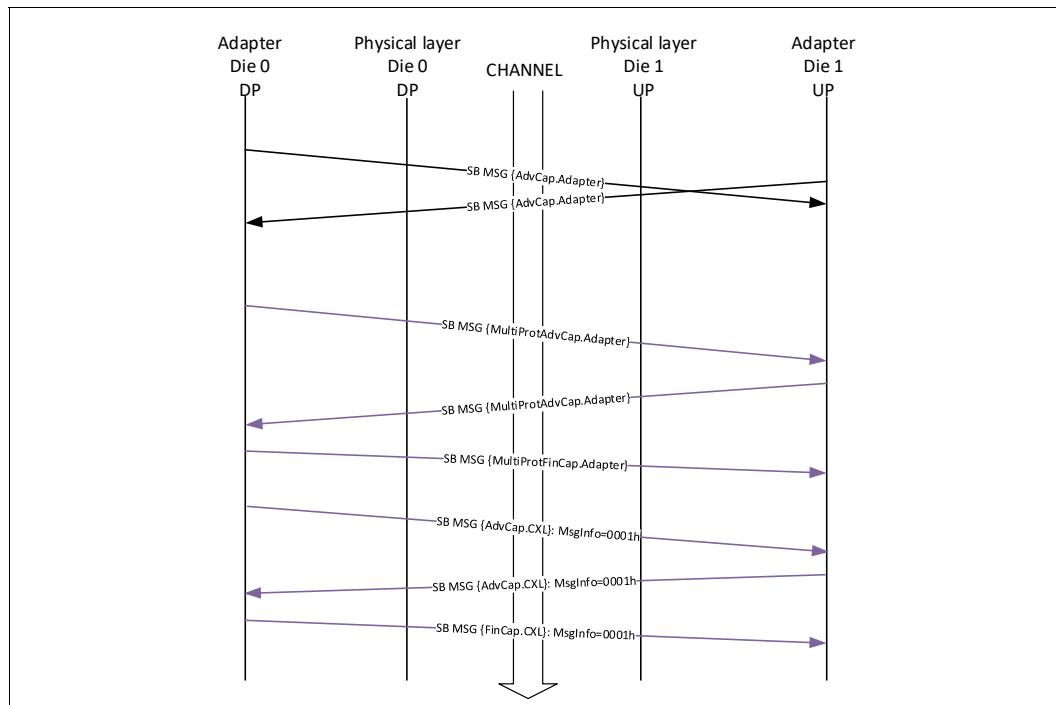
Figure 3-6. Stack 0 is PCIe, Stack 1 is Streaming**Figure 3-7. Stack 0 is Streaming, Stack 1 is PCIe**

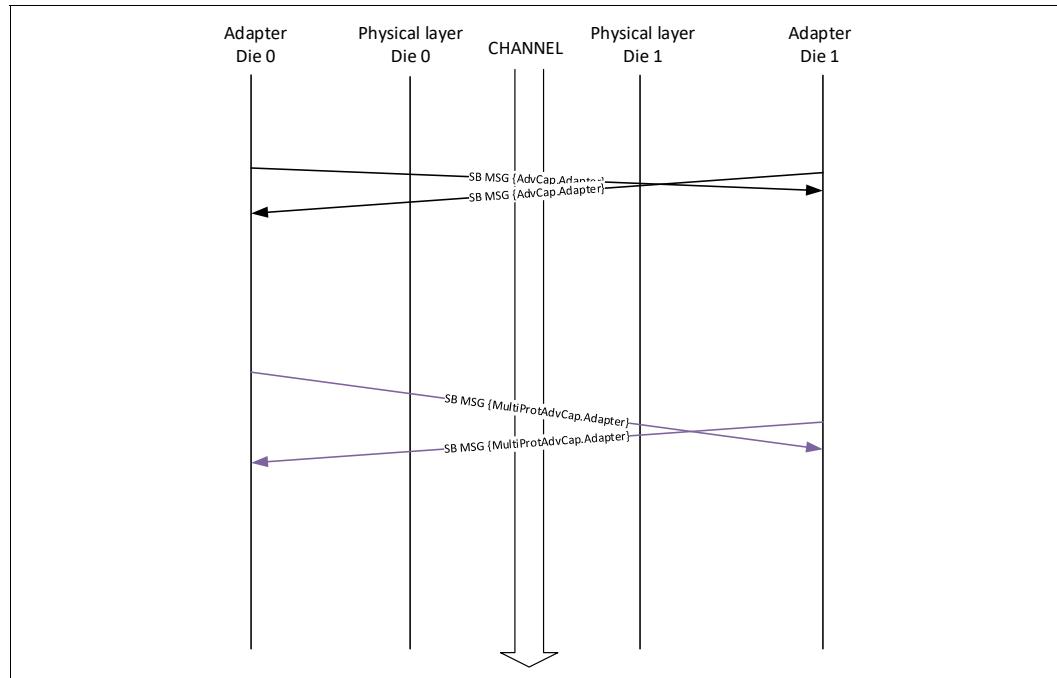
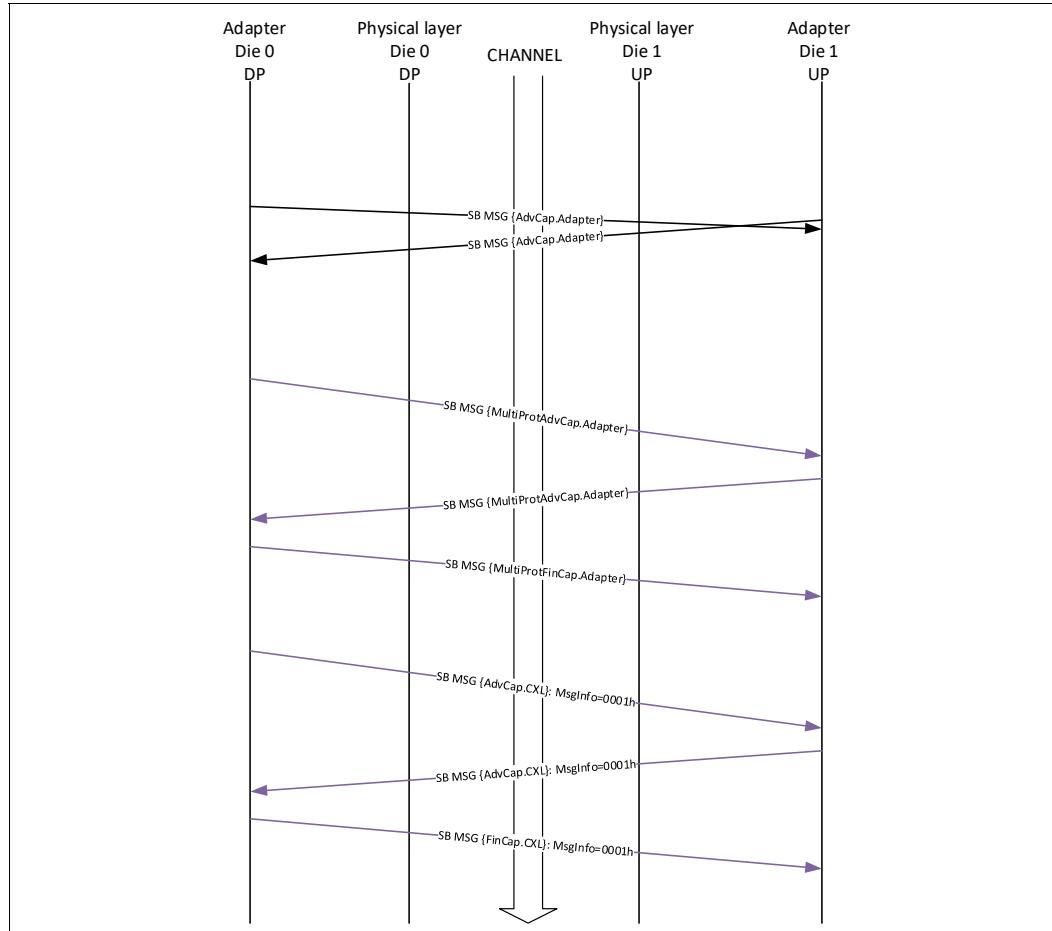
Figure 3-8. Both Stacks are Streaming

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 <i>PCIe Base Specification</i> 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

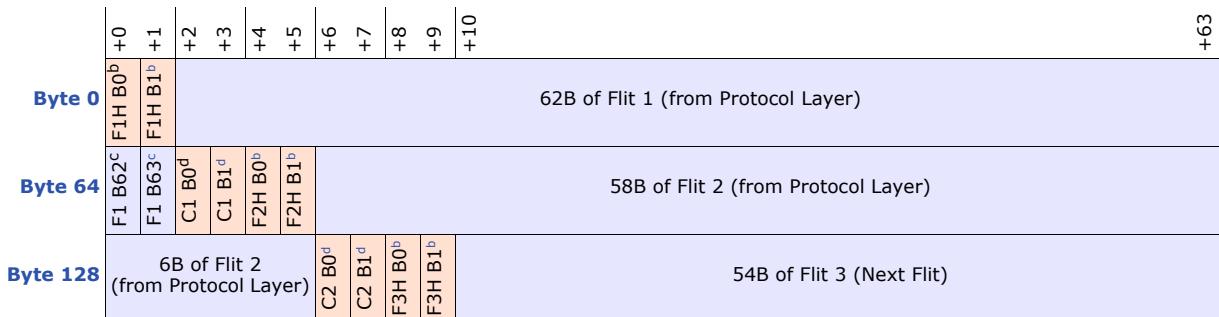
When Retry is disabled, the BER of the Link is 1E-27 or lower. In these cases, any bit error is an uncorrectable error for the Link. As a best practice, it is strongly recommended for receiver implementations to have an uncorrectable internal error condition for scenarios in which neither a valid Flit 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



- See Figure 2-1 for color mapping.
- 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.
- Flit 1 Byte 62 and Byte 63, respectively (from Protocol Layer).
- 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

	+0	+1	+2	+3	+4	+5	+6	+63
Byte 0	FH B0 ^b	FH B1 ^b						
Byte 64	2B (from Protocol Layer)	CRC B0 ^c	CRC B1 ^c	PDS B0 ^d	PDS B1 ^d			
Byte 128								64B all 0 data
Byte 192								64B all 0 data

- 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

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+10	+63
Byte 0	F1H B0 ^b	F1H B1 ^b										
Byte 64	F1 B62 ^c	F1 B63 ^c	C1 B0 ^d									
Byte 128	F2 B58 ^e	F2 B59 ^e	F2 B60 ^e	F2 B61 ^e	C1 B1 ^d	F2H B0 ^b	F2H B1 ^b					
Byte 192						C2 B0 ^d	C2 B1 ^d	PDS B0 ^f	PDS B1 ^f			
Byte 256												
Byte 320												
Byte 384												
Byte 448												

- 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.

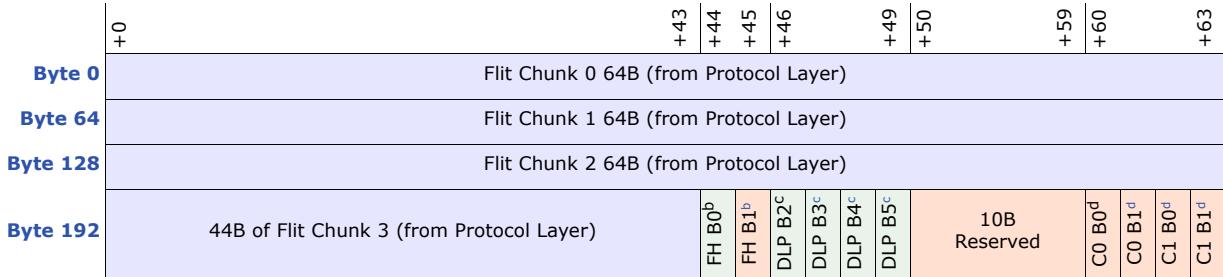
The Flit Header byte formats are shown in [Table 3-5](#) when Retry is required; otherwise, it is as shown in [Table 3-4](#).

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).

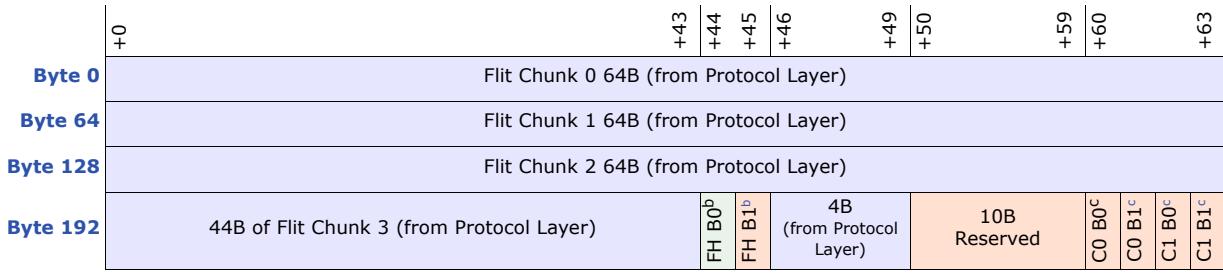
For Management Flits, Bytes 238 to 241 are driven from the Protocol Layer with Management Transport Credit Return DWORD (CRD) Bytes 0 to 3 (see [Section 8.2.5.2.2](#) for CRD format). Bytes 232 to 235 in *Format 3* and Bytes 234 to 237 in *Format 4* are driven from the Protocol Layer with 0s for Management 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:

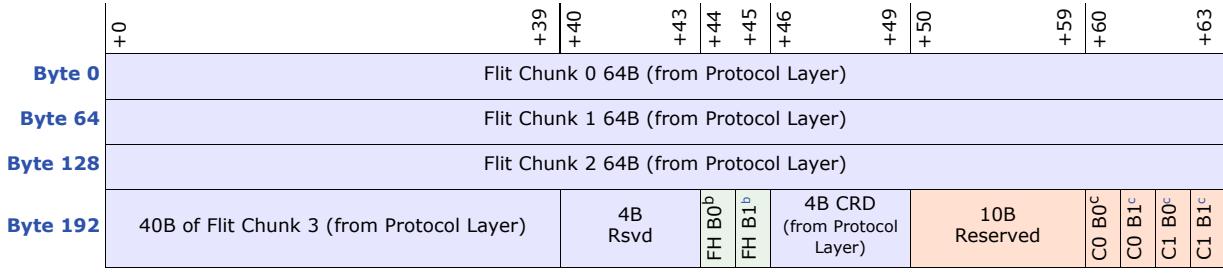
- If bits [7:6] of Byte 1 are 10b, the Adapter passes through Bytes 238 to 241 from the Protocol Layer to the Link
- If bits [7:6] of Byte 1 are 00b, Bytes 238 to 241 are treated per PCIe/CXL.io DLP rules for this flit format

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.

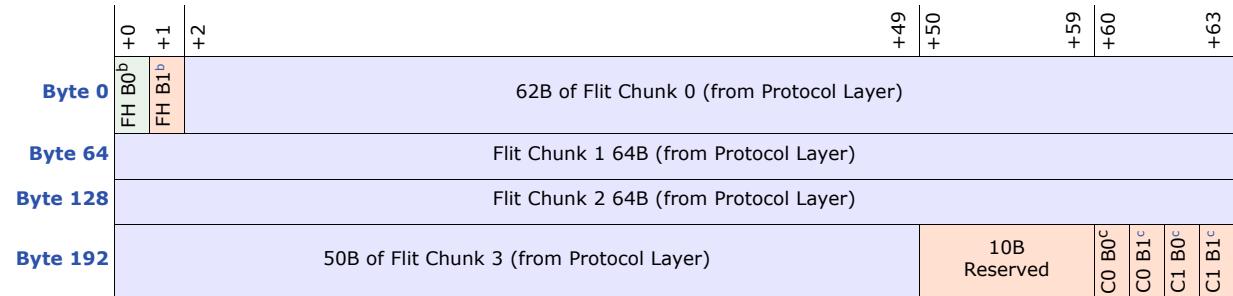
Figure 3-15. Format 3: Standard 256B End Header Flit Format for Streaming Protocol^a

- 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

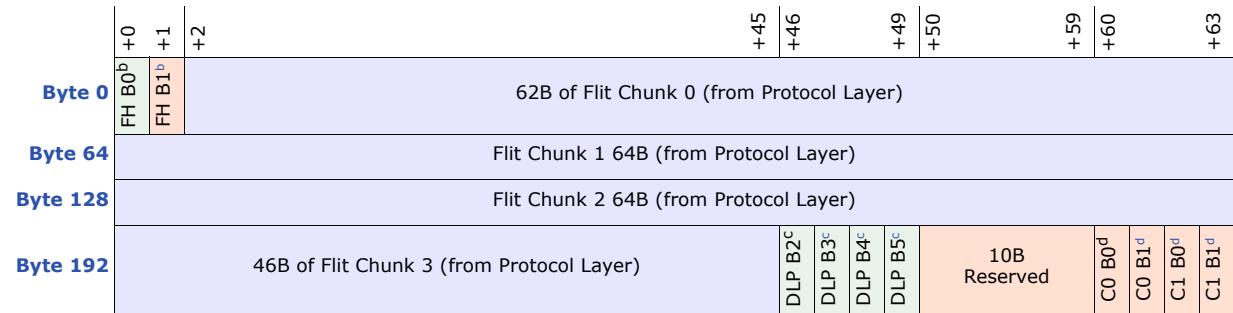


a. See Figure 2-1 for color mapping.

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



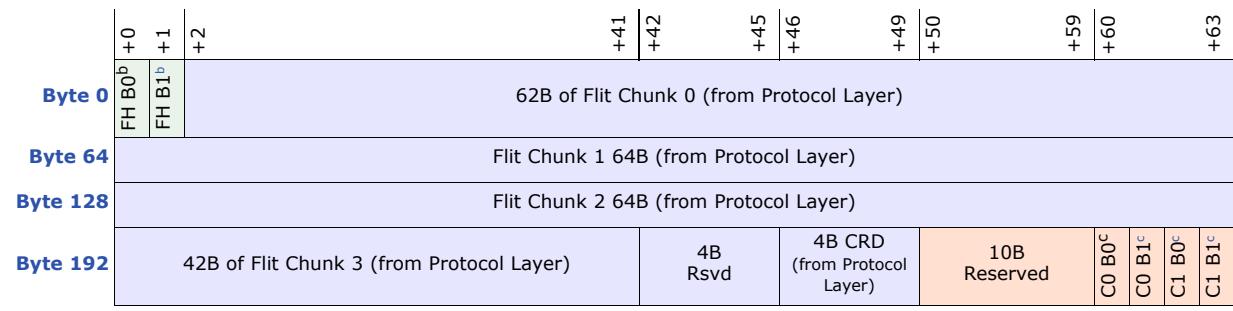
a. See Figure 2-1 for color mapping.

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

Byte 0 FH B0 ^b FH B1 ^b	+0 +1 +2	62B of Flit Chunk 0 (from Protocol Layer)				+51 +52	+57 +58			
Byte 64	58B of Flit Chunk 1 (from Protocol Layer)				DLP B2 ^c DLP B3 ^c DLP B4 ^c DLP B5 ^c CO B0 ^d CO B1 ^d	+64 +65 +66 +67				
Byte 128	Flit Chunk 2 64B (from Protocol Layer)				DLP B6 ^c DLP B7 ^c DLP B8 ^c DLP B9 ^c CO B1 ^d CO B2 ^d	+68 +69 +70 +71				
Byte 192	52B of Flit Chunk 3 (from Protocol Layer)				6B Reserved	FM B0 ^e FM B1 ^e FM B2 ^e FM B3 ^e C1 B0 ^d C1 B1 ^d	+74 +75 +76 +77			

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

Byte 0 FH B0 ^b FH B1 ^b	+0 +1 +2	62B of Flit Chunk 0 (from Protocol Layer)		+51 +52	+57 +58	+61 +62	+63
Byte 64	58B of Flit Chunk 1 (from Protocol Layer)		4B Reserved	C0 B0 ^c C0 B1 ^c			
Byte 128	Flit Chunk 2 64B (from Protocol Layer)						
Byte 192	52B of Flit Chunk 3 (from Protocol Layer)		10B Reserved	C1 B0 ^c C1 B1 ^c			

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

Byte 0 FH B0 ^b FH B1 ^b	+0 +1 +2	+47 +48	+51 +52	+57 +58	+61 +62	+63	
Byte 64	58B of Flit Chunk 1 (from Protocol Layer)		4B Reserved	C0 B0 ^c C0 B1 ^c			
Byte 128	Flit Chunk 2 64B (from Protocol Layer)						
Byte 192	48B of Flit Chunk 3 (from Protocol Layer)		4B CRD (from Protocol Layer)	10B Reserved	C1 B0 ^c C1 B1 ^c		

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

Byte 0 FH B0 ^b FH B1 ^b	+0 +1 +2	+55 +56 +57	+58	+61 +62	+63		
Byte 64	58B of Flit Chunk 1 (from Protocol Layer)		DLP B2 ^c DLP B3 ^c DLP B4 ^c DLP B5 ^c	C0 B0 ^d C0 B1 ^d			
Byte 128	Flit Chunk 2 64B (from Protocol Layer)						
Byte 192	56B of Flit Chunk 3 (from Protocol Layer)		2B Rsvd	FM B0 ^e FM B1 ^e FM B2 ^e FM B3 ^e FM B4 ^e C1 B0 ^d C1 B1 ^d	+61 +62 +63		

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

	FH B0 ^d	+0		+51		+58		+61
	FH B1 ^b	+1		+52		+62		+63
		+2						
Byte 0								
			62B of Flit Chunk 0 (from Protocol Layer)					
Byte 64			58B of Flit Chunk 1 (from Protocol Layer)					
Byte 128			Flit Chunk 2 64B (from Protocol Layer)					
Byte 192			52B of Flit Chunk 3 (from Protocol Layer)					
	H B4 ^c							
	H B5 ^c							
	H B6 ^c							
	H B7 ^c							
	H B8 ^c							
	H B9 ^c							
	H B10 ^c							
	H B11 ^c							
	H B12 ^c							
	H B13 ^c							
	C0 B0 ^d							
	C1 B0 ^d							
	C0 B1 ^d							
	C1 B1 ^d							

- 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

	FH B0 ^a FH B1 ^b	+0 +1 +2	62B of Flit Chunk 0 (from Protocol Layer)	+61
Byte 0				+62 +63
Byte 64			62B of Flit Chunk 1 (from Protocol Layer)	C0 B0 ^c C0 B1 ^c
Byte 128			Flit Chunk 2 64B (from Protocol Layer)	
Byte 192			62B of Flit Chunk 3 (from Protocol Layer)	C1 B0 ^c C1 B1 ^c

- 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

	FH B0 ^b FH B1 ^b	+0 +1	+2	+51	+52	+57	+58	+61	+62	+63
Byte 0	62B of Flit Chunk 0 (from Protocol Layer)									
Byte 64	62B of Flit Chunk 1 (from Protocol Layer)									C0 B0 ^c C0 B1 ^c
Byte 128	Flit Chunk 2 64B (from Protocol Layer)									
Byte 192	52B of Flit Chunk 3 (from Protocol Layer)				6B Rsvd	4B CRD (from Protocol Layer)	C1 B0 ^c C1 B1 ^c			

- 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
<i>Format 1</i>	Raw	Protocol Layer populates all the bytes on FDI. Adapter passes to RDI without modifications or additions.	<ul style="list-style-type: none"> • Section 3.3.1 • Figure 3-10
<i>Format 2</i>	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.	<ul style="list-style-type: none"> • Section 3.3.2 • Figure 3-11 • Figure 3-12
<i>Format 3</i>	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.	<ul style="list-style-type: none"> • Section 3.3.3 • Figure 3-14 • Figure 3-15
<i>Format 4</i>	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.	<ul style="list-style-type: none"> • Section 3.3.3 • Figure 3-17 • Figure 3-18
<i>Format 5</i>	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.	<ul style="list-style-type: none"> • Section 3.3.4 • Figure 3-20 • Figure 3-21
<i>Format 6</i>	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.	<ul style="list-style-type: none"> • 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_LatOptFmt6	
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 PCIe 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

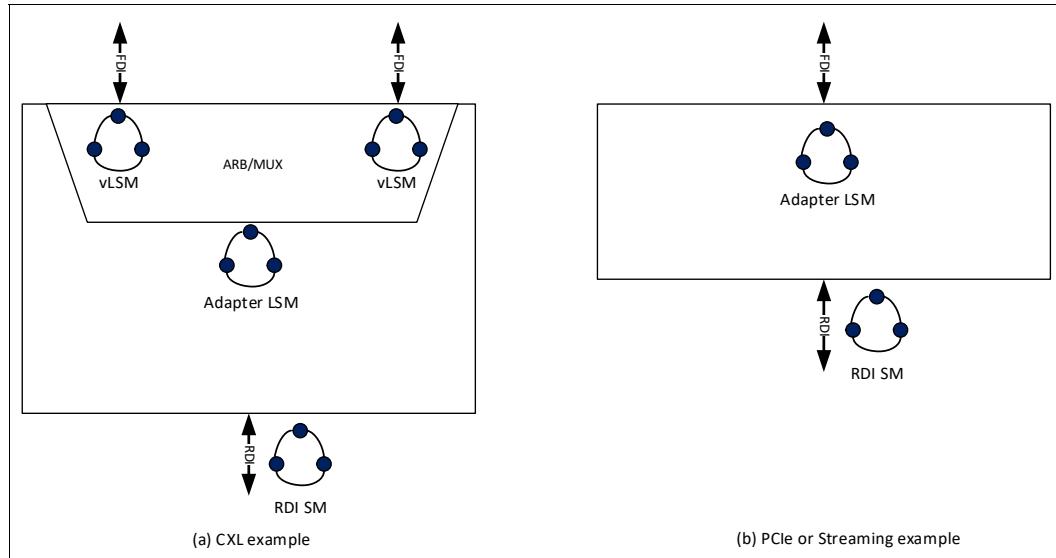
UCIE has a hierarchical approach to Link state management in order to have well-defined functionality partitioning between the different layers and also enabling common state transitions or sequencing at FDI and RDI.

[Figure 3-27](#) shows examples of state machine hierarchy for different configurations. For CXL, the ARB/MUX vLSMs are exposed on FDI `p1_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 `p1_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.

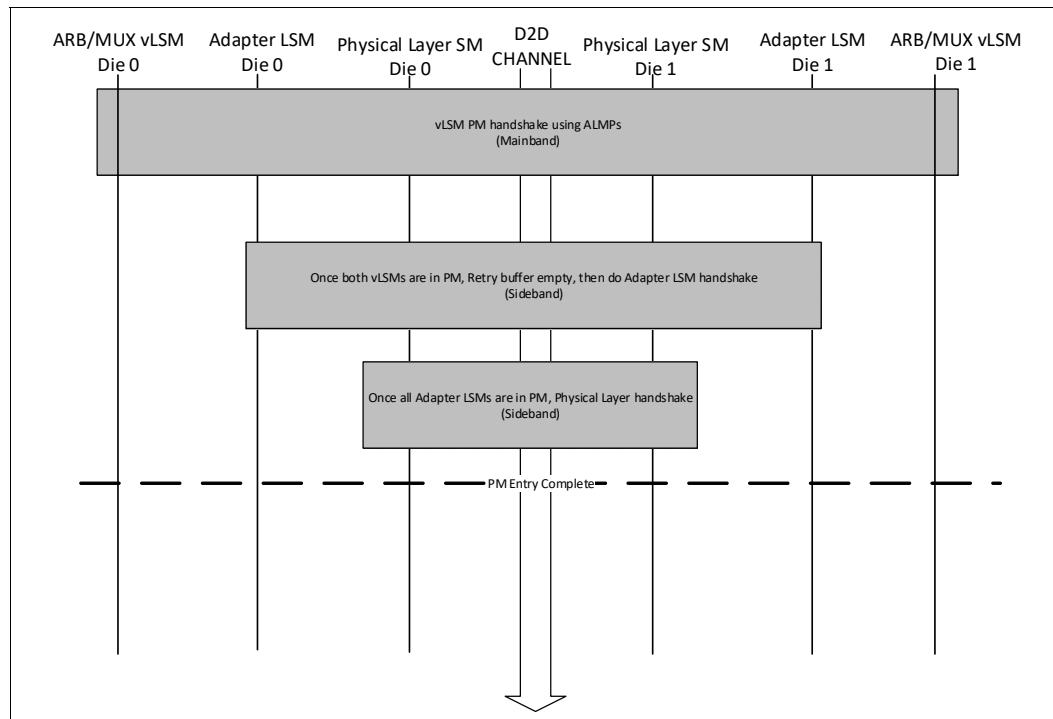
The Power management state entry sequence is as follows:

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

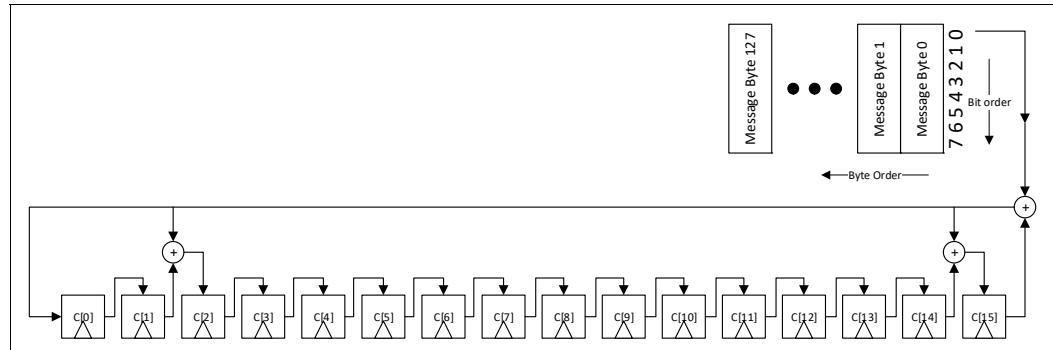
The CRC generator polynomial is $(x+1)*(x^{15} + x + 1) = x^{16} + x^{15} + x^2 + 1$. This gives a 3-bit detection guarantee for random bit errors: 2 bit detection guarantee is because of the primitive polynomial $(x^{15} + x + 1)$, and 1 additional bit error detection guarantee is provided by making 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.v` (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 (`1c1k`) 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 64B 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 \leq 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 \leq 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:

$$\text{ParityByte } X, \text{ bit } 0 = \wedge((\text{DataByte } [X]) \wedge (\text{DataByte } [X + 64*N]) \wedge (\text{DataByte } [X + 128*N]) \wedge \dots \wedge (\text{DataByte } [X + (256*256*N - 64*N)]))$$

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

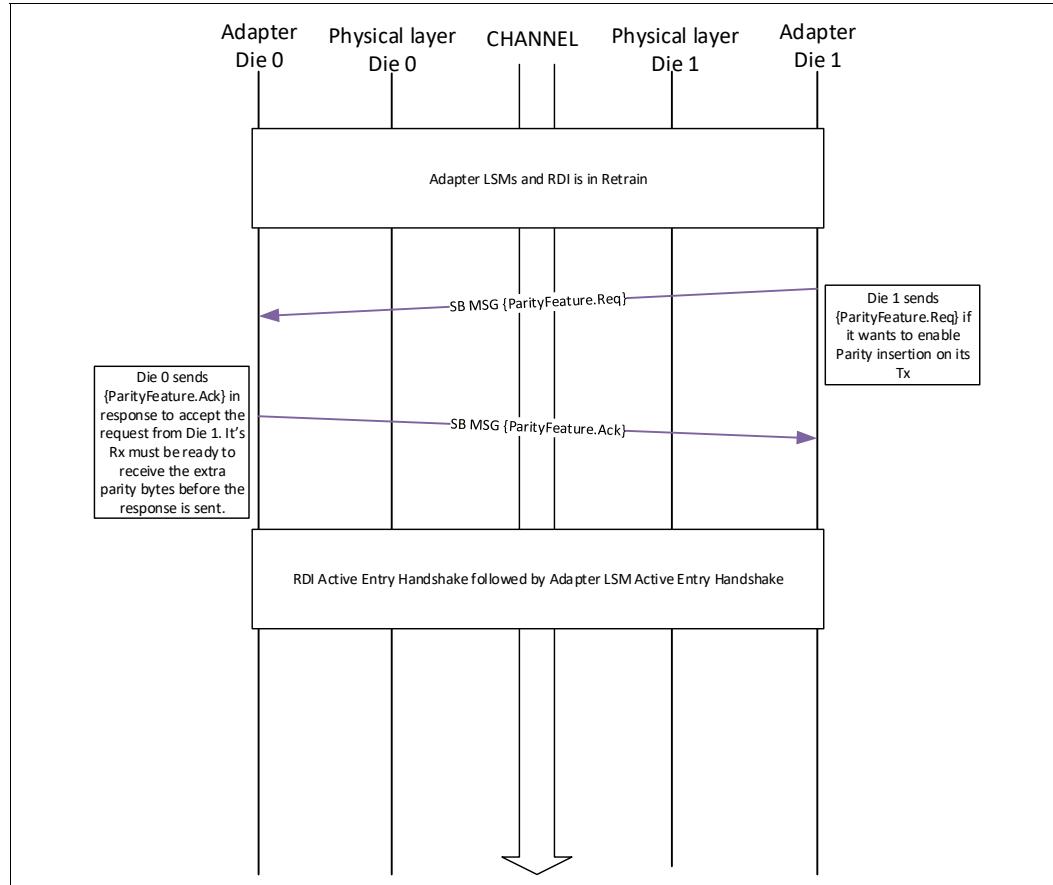
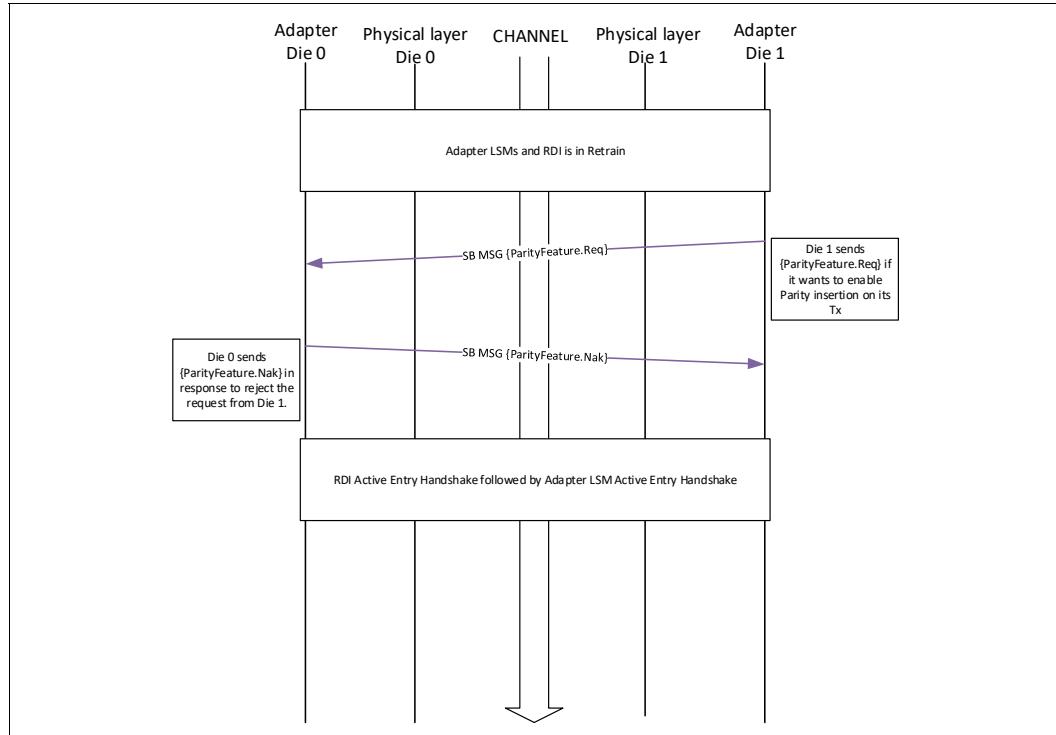


Figure 3-31. Unsuccessful Parity Feature negotiation between Die 1 Tx and Die 0 Rx

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)

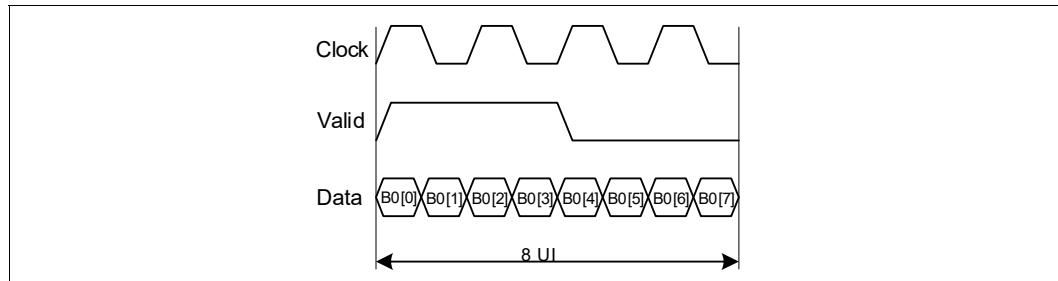
4.1 Data and Sideband Transmission Flow

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

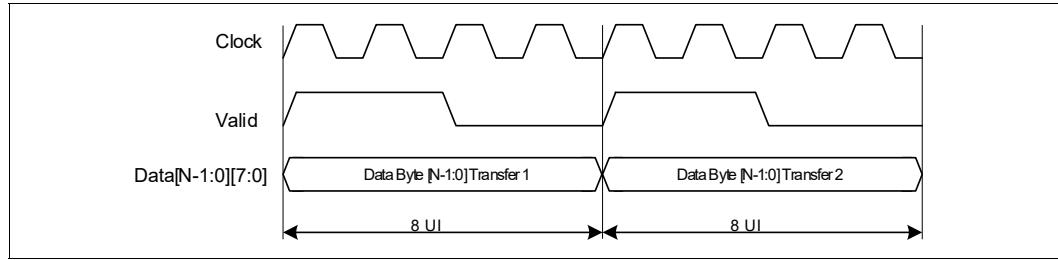
Figure 4-5. Byte to Lane mapping for Standard package x16 degraded to x8

Lane	8	9	10	11	12	13	14	15
or								
Lane UI \ 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

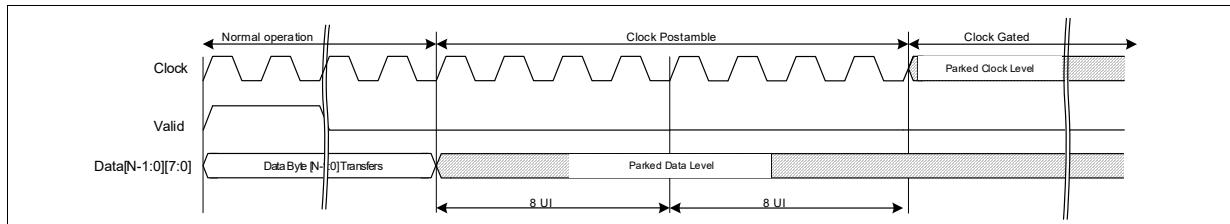
8-UI Valid ^a	Encoding
00000000b	No Flit data transfer + no credit release
00001111b	Flit data transfer valid + no credit release
11110000b	No Flit data transfer + 1 credit release
11111111b	Flit data transfer valid + 1 Credit release

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

4.1.4 Free Running Clock Mode

Free running clock mode is defined as the mode where the forwarded clock remains toggling even when the Transmitter for Valid is held low and there is no data transfer on the interface. This mode must be supported to allow disabling dynamic clock gating for normal operation or debug. This must be negotiated prior to mainband Link training through parameter exchange.

4.1.5 Sideband transmission

Each module supports a sideband interface with a serial data and clock pin pair. Sideband packet formats and encodings are shown in [Chapter 7.0](#) and the electrical characteristics are shown in [Section 5.13](#).

As shown in [Section 7.1.2](#), the sideband message formats are defined as a 64-bit header with no data, with 32 bits or 64 bits of data. A 64-bit serial packet is defined on the I/O interface to the remote die as shown in [Figure 4-8](#). 32-bit data is sent using the 64-bit serial packet with MSBs padded with 0b. Two sideband serial packets on the I/O interface are separated by a minimum of 32 bits low as shown in [Figure 4-9](#). A sideband message with data would be transferred as a 64-bit header followed by 32 bits of low followed by 64-bit data followed by 32 bits of low.

Figure 4-8. Example 64-bit Sideband Serial Packet Transfer

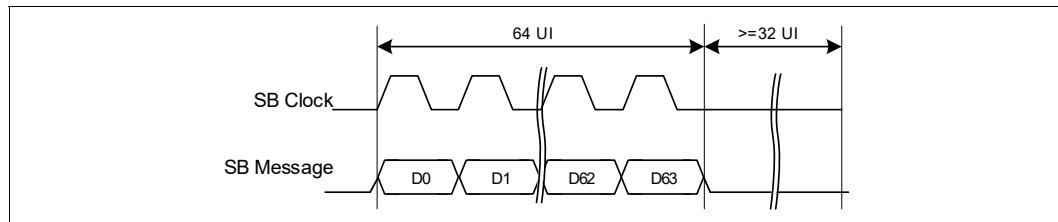
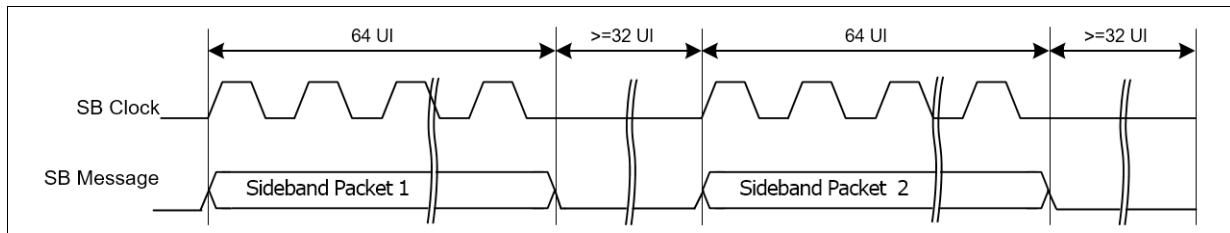


Figure 4-9. Sideband Packet Transmission: Back-to-Back



4.1.5.1 Sideband Performant Mode Operation (PMO)

Sideband designs can negotiate support for 'Sideband Performant Mode Operation (PMO)' by way of the Sideband Feature Extensions (SBFE) mechanism defined in [Section 4.5.3.3.1.1](#). The Sideband PMO bit of the {MBINIT.PARAM SBFE req/respond} sideband message (bit 1) is defined for negotiating this operation (see [Table 7-11](#)). When supporting this feature, a UCIe Link must support this capability on both its transmit and receive direction or not support the capability at all. In a multi-module Link, all modules must advertise the same capability. A UCIe Link must set the Sideband PMO bit to 1 on the {MBINIT.PARAM SBFE respond} sideband message that the UCIe Link transmits, but only if the corresponding Req message also has that bit set to 1 and its receiver is capable of Sideband PMO. Otherwise, the Sideband PMO bit must be cleared to 0.

When Sideband PMO capability is enabled, the 32-UI dead time between the 64-UI data transfers on the sideband is no longer applicable and the sideband can transmit 64-UI data back-to-back with no gaps. See [Figure 4-10](#) and [Figure 4-11](#) for illustration. The transmitter must follow this new mode after the transmitter has sent and received the {MBINIT.PARAM SBFE respond} sideband message with

the Sideband PMO bit set to 1, across all modules. The receiver must be ready to accept packets in this mode after the receiver has transmitted the {MBINIT.PARAM SBFE resp} sideband message with the Sideband PMO bit set to 1. After Sideband PMO is enabled, the transmitter operates in Performant Mode in all states until entry into the RESET state with the SB_MGMT_UP flag cleared to 0. Additionally, PMO can then only be renegotiated on a training sequence with the SB_MGMT_UP flag cleared to 0. Note that from a receiver perspective, due to timing differences, packets might be received without the Sideband Performant Mode even after the chiplet has transmitted an {MBINIT.PARAM SBFE resp} sideband message with the PMO bit set to 1. The sideband receiver must be backward compatible and be able to handle 32 UI of gaps between consecutive 64-UI transfers over the sideband Link.

Figure 4-10. Example 64-bit Sideband Serial Packet Transfer in Sideband Performant Mode

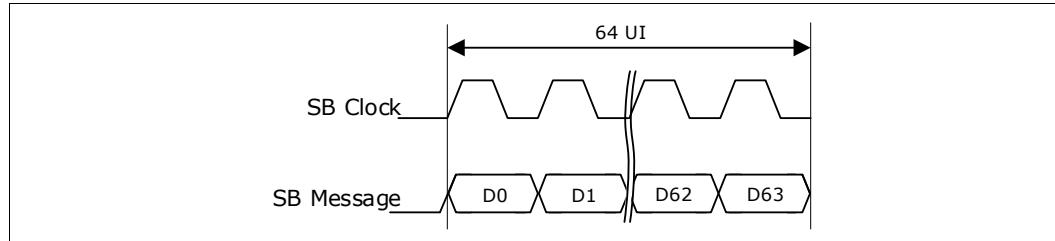
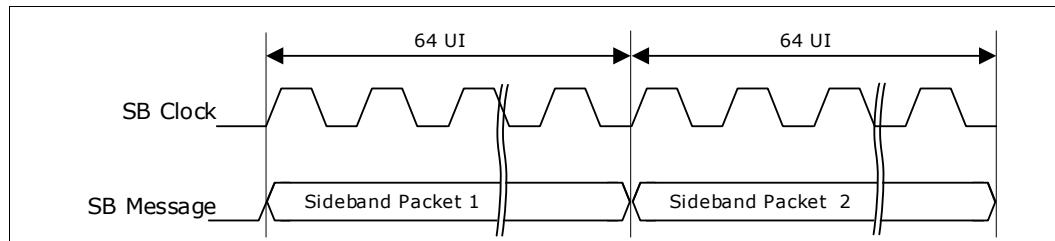


Figure 4-11. Sideband Packet Transmission: Back-to-Back in Sideband Performant Mode



4.2 Lane Reversal

In Section 4.2, Section 4.3, and Section 4.5, the following nomenclature is used:

- **TD_P**: Physical Lane for Data Transmitter
- **RD_P**: Physical Lane for Data Receiver
- **TRD_P**: Physical Lane for Redundant Data Transmitter
- **RRD_P**: Physical Lane for Redundant Data Receiver
- **TD_L**: Logical Lane for Data Transmit
- **RD_L**: Logical Lane for Data Receive
- **TCKP_P**, **TCKN_P** and **TTRK_P**: Physical Lane for Clock and Track Transmitter
- **TCKP_L**, **TCKN_L** and **TTRK_L**: Logical Lane for Clock and Track Transmitter
- **RCKP_P**, **RCKN_P** and **RTRK_P**: Physical Lane for Clock and Track Receiver
- **RCKP_L**, **RCKN_L** and **RTRK_L**: Logical Lane for Clock and Track Receiver
- **TRDCK_P**: Physical Lane for Redundant Clock/Track Transmitter
- **RRDCK_P**: Physical Lane for Redundant Clock/Track Receiver

- **TRDCK_L:** Logical Lane for Redundant Clock/Track Transmitter
- **RRDCK_L:** Logical Lane for Redundant Clock/Track Receiver
- **TVLD_P, RVLD_P:** Physical Lane for Valid Transmitter and Receiver
- **TRDVLD_P, RRDVLD_P:** Physical Lane for Redundant Valid Transmitter and Receiver
- **TVLD_L, RVLD_L:** Logical Lane for Valid Transmitter and Receiver
- **TRDVLD_L, RRDVLD_L:** Logical Lane for Redundant Valid Transmitter and Receiver

Devices must support Lane reversal of data Lanes within a Module. An example of Lane reversal is when physical Data Lane 0 on local die is connected to physical Data Lane (N-1) on the remote die (physical Data Lane 1 is connected to physical Data Lane N-2 and so on) where N = 8 for a x8 Standard Package, N = 16 for a x16 Standard Package, N = 32 for a x32 Advanced Package, and N = 64 for a x64 Advanced Package. Redundant Lanes, in case of Advanced Package, are also reversed. Lane reversal must be implemented on the Transmitter only. The Transmitter reverses the logical Lane order on Data and Redundant Lanes.

Track, Valid, Clock, and sideband signals must not be reversed.

Lane reversal is discovered and applied during initialization and training (see [Section 4.5.3.3.5](#)).

4.2.1 Lane ID

To allow Lane reversal discovery, each logical Data and redundant Lane within a module is assigned a unique Lane ID. The assigned Lane IDs are shown in [Table 4-2](#) and [Table 4-3](#) for Advanced and Standard Package modules, respectively. Note that logical Lane numbers in [Table 4-2](#) and [Table 4-3](#) represent the logical Transmitter and Receiver Lanes. For example, Logical Lane Number = 0 represents `TD_L[0]/RD_L[0]` and so on.

In [Table 4-2](#), for a x64 Advanced Package module, logical Lane numbers 64, 65, 66, and 67 represent Logical redundant Lanes `TRD_L[0]/RRD_L[0]`, `TRD_L[1]/RRD_L[1]`, `TRD_L[2]/RRD_L[2]`, `TRD_L[3]/RRD_L[3]`, respectively. For a x32 Advanced Package module, the Lane ID for `TD_L[0:31]/RD_L[0:31]`, `TRD_L[0]/RRD_L[0]` and `TRD_L[1]/RRD_L[1]` will be represented by the set of Lane ID {0...31, 64, 65} respectively.

In [Table 4-3](#), for a x16 Standard Package module, the Lane ID for `TD_L[0:15]/RD_L[0:15]` will be represented by the set of Lane ID {0...15} respectively. For a x8 Standard Package module, the Lane ID for `TD_L[0:7]/RD_L[0:7]` will be represented by the set of Lane ID {0...7} respectively.

Table 4-2. Lane ID: Advanced Package module (Sheet 1 of 2)

Logical Lane Number	Lane ID	Logical Lane Number	Lane ID
0	00000000b	34	00100010b
1	00000001b	35	00100011b
2	00000010b	36	00100100b
3	00000011b	37	00100101b
4	00000100b	38	00100110b
5	00000101b	39	00100111b
6	00000110b	40	00101000b
7	00000111b	41	00101001b
8	00001000b	42	00101010b

Table 4-2. Lane ID: Advanced Package module (Sheet 2 of 2)

Logical Lane Number	Lane ID	Logical Lane Number	Lane ID
9	00001001b	43	00101011b
10	00001010b	44	00101100b
11	00001011b	45	00101101b
12	00001100b	46	00101110b
13	00001101b	47	00101111b
14	00001110b	48	00110000b
15	00001111b	49	00110001b
16	00010000b	50	00110010b
17	00010001b	51	00110011b
18	00010010b	52	00110100b
19	00010011b	53	00110101b
20	00010100b	54	00110110b
21	00010101b	55	00110111b
22	00010110b	56	00111000b
23	00010111b	57	00111001b
24	00011000b	58	00111010b
25	00011001b	59	00111011b
26	00011010b	60	00111100b
27	00011011b	61	00111101b
28	00011100b	62	00111110b
29	00011101b	63	00111111b
30	00011110b	64	01000000b
31	00011111b	65	01000001b
32	00100000b	66	01000010b
33	00100001b	67	01000011b

Table 4-3. Lane ID: Standard Package Module

Logical Lane Number	Lane ID	Logical Lane Number	Lane ID
0	00000000b	8	00001000b
1	00000001b	9	00001001b
2	00000010b	10	00001010b
3	00000011b	11	00001011b
4	00000100b	12	00001100b
5	00000101b	13	00001101b
6	00000110b	14	00001110b
7	00000111b	15	00001111b

4.3 Interconnect redundancy remapping

As discussed in [Section 5.9](#), Advanced Package modules require redundancy remapping to recover from faulty Lanes. This section provides the details of remapping. After a successful remapping/repair, any unused repair Lanes must have their Transmitters tri-stated and Receivers disabled.

4.3.1 Data Lane repair

The specification supports remapping (repair) of up to two data Lanes for each group of 32 data Lanes. $\text{TD_P}[31:0]$ ($\text{RD_P}[31:0]$) and $\text{TD_P}[63:32]$ ($\text{RD_P}[63:32]$) are treated as two separate groups of 32 Lanes that can be independently repaired using redundant Lanes, $\text{TRD_P}[1:0]$ ($\text{RRD_P}[1:0]$) and $\text{TRD_P}[3:2]$ ($\text{RRD_P}[3:2]$), respectively. $\text{TD_P}[63:32]$ ($\text{RD_P}[63:32]$) and hence $\text{TRD_P}[3:2]$ ($\text{RRD_P}[3:2]$) do not apply to x32 Advanced Package Link.

Lane remapping is accomplished by “shift left” or “shift right” operation. A “shift left” is when data traffic of logical Lane $\text{TD_L}[n]$ on $\text{TD_P}[n]$ is multiplexed onto $\text{TD_P}[n-1]$. A shift left puts $\text{TD_L}[0]$ onto $\text{TRD_P}0$ or $\text{TD_L}[32]$ onto $\text{TRD_P}2$. A shift right operation is when data traffic $\text{TD_L}[n]$ is multiplexed onto $\text{TD_P}[n+1]$. A shift right puts $\text{TD_L}[31]$ onto $\text{TRD_P}1$ or $\text{TD_L}[63]$ onto $\text{TRD_P}3$. See the pseudo codes in [Section 4.3.3.1](#) and [Section 4.3.3.2](#) that show the changes in mapping post repair.

Note: If the lower index redundant Lane ($\text{TRD_P}[0]$ or $\text{TRD_P}[2]$) is faulty, no data lanes can be repaired for its group. Note that if the higher index redundant lane ($\text{TRD_P}[1]$ or $\text{TRD_P}[3]$) is faulty, one data lane can be repaired for its group.

After a data Lane is remapped, the Transmitter associated with the faulty physical Lane is tri-stated and the Receiver is disabled. The Transmitter and the Receiver of the redundant Lane used for the repair are enabled.

[Figure 4-12](#) shows transmit bump side of data Lane remapping for the first group of 32 Lanes. Both “shift left” and “shift right” remapping is needed to optimally repair up to any two Lanes within the group. [Figure 4-13](#) shows details of the mux structure used for data Lane repair.

Note: Example repair implementations are shown for $\text{TD_P}[31:0]$ for clarity. It should be noted that the same schemes are also applicable to $\text{TD_P}[63:32]$.

Figure 4-12. Data Lane remapping possibilities to fix potential defects

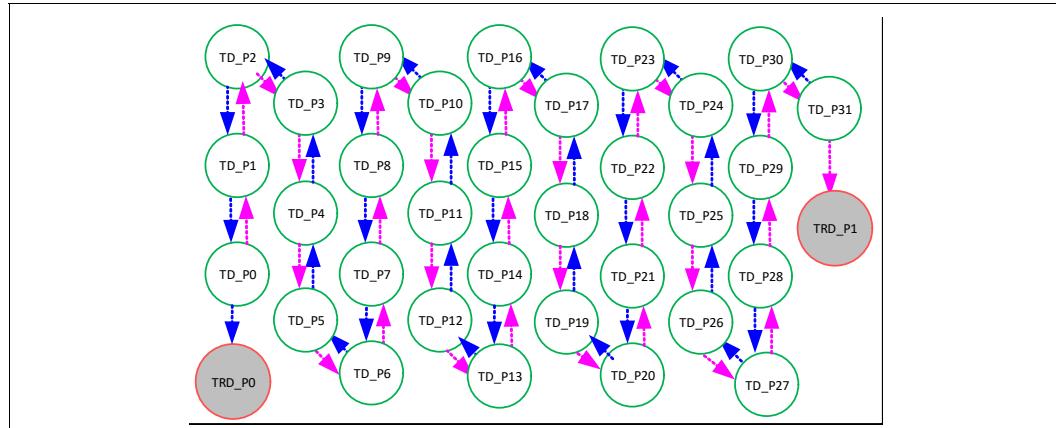
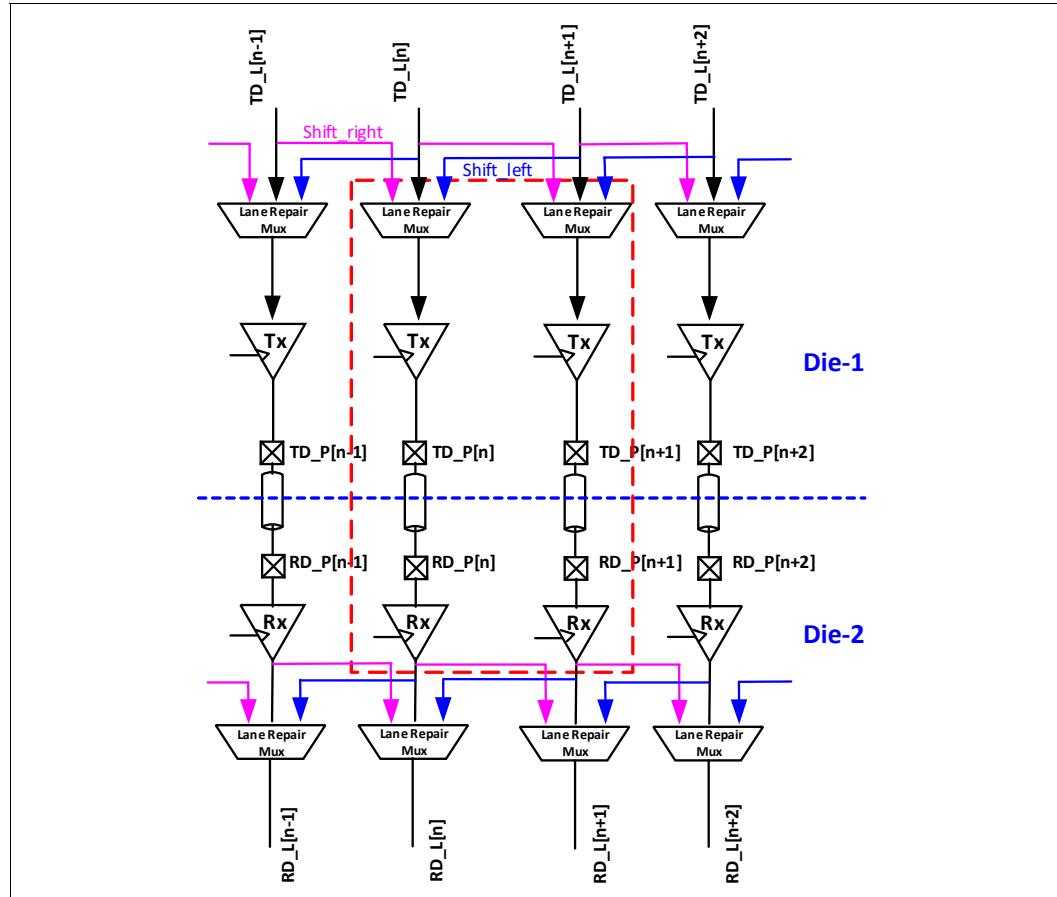


Figure 4-13. Data Lane remapping: Mux chain

4.3.2 Data Lane repair with Lane reversal

If Lanes are reversed, physical Lane 0 of Transmitter (`TD_P[0]`) is connected to physical Lane N-1 of the Receiver (`RD_P[N-1]`). N is 64 or 32 for x64 or x32 Advanced Package modules, respectively. See the pseudo codes in [Section 4.3.3.3](#) and [Section 4.3.3.4](#) that show the changes in mapping post repair.

4.3.3 Data Lane repair implementation

4.3.3.1 Single Lane repair

`TRD_P[0] (RRD_P[0])` must be used as the redundant Lane to remap any single physical Lane failure for `TD_P[31:0] (RD_P[31:0])`. `TRD[2] (RRD[2])` must be used as the redundant Lane to remap any single Lane failure for `TD_P[63:32] (RD_P[63:32])`.

Pseudo code for repair in `TD_P[31:0] (RD_P[31:0])` ($0 \leq x \leq 31$):

```
IF failure occurs in TD_P[x]:
    IF x > 0:
        FOR 0 <= i < x:
            TD_P[x-i-1] = TD_L[x-i]
            RD_L[x-i] = RD_P[x-i-1]
        TRD_P[0] = TD_L[0]
        RD_L[0] = RRD_P[0]
```

Pseudo code for repair in `TD_P[63:32] (RD_P[63:32])` ($32 \leq x \leq 63$) (this does not apply to x32 Advanced Package Link):

```
IF failure occurs in TD_P[x]:
    IF x > 32:
        FOR 0 <= i < x-32:
            TD_P[x-i-1] = TD_L[x-i]
            RD_L[x-i] = RD_P[x-i-1]
        TRD_P[2] = TD_L[32]
        RD_L[32] = RRD_P[2]
```

As shown in Figure 4-14 TD_P[29] is remapped in the direction to use TRD_P[0] as the repair resource. Figure 4-15 shows the circuit implementation.

Figure 4-14. Example of Single Lane failure remapping

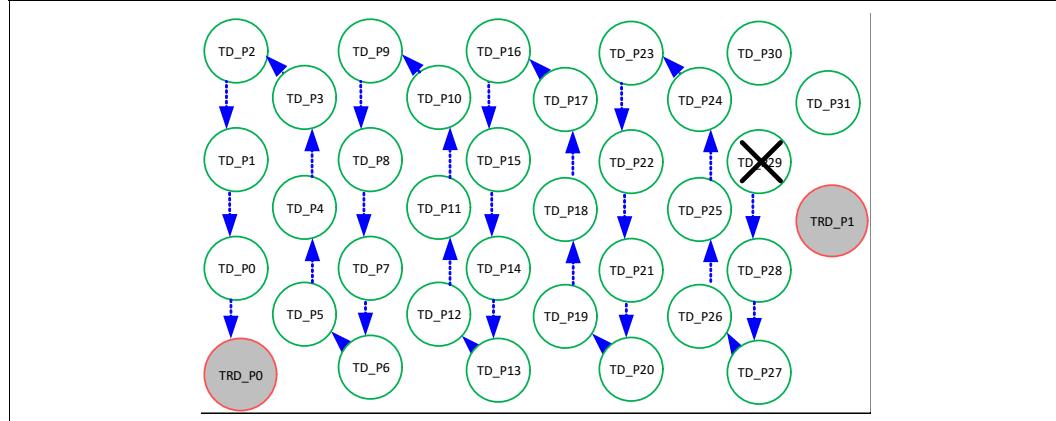
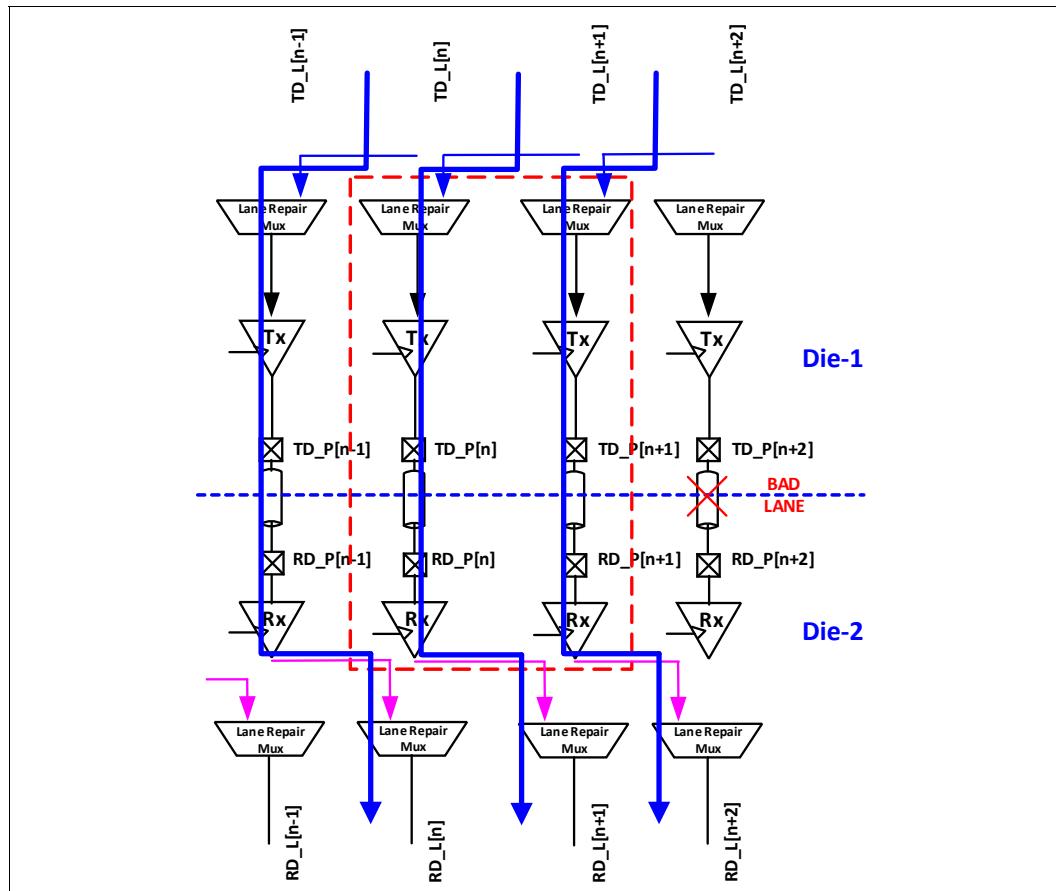


Figure 4-15. Example of Single Lane remapping implementation



4.3.3.2 Two Lane repair

Any two Lanes within a group of 32 can be repaired using the two redundant bumps. For any two physical Lane failures in **TD_P[31:0]** (**RD_P[31:0]**), the lower Lane must be remapped to **TRD_P[0]** (**RRD_P[0]**) and the upper Lane is remapped to **TRD_P[1]** (**RRD_P[1]**). For any two physical Lane failures in **TD_P[63:32]** (**RD_P[63:31]**), the lower Lane must be remapped to **TRD_P[2]** (**RRD_P[2]**) and the upper Lane is remapped to **TRD_P[3]** (**RRD_P[3]**).

Pseudo code for two Lane repair in **TD_P[31:0]** (**RD_P[31:0]**) ($0 \leq x, y \leq 31$):

```

IF failure occurs in TD_P[x], TD_P[y] AND (x < y):
    IF x > 0:
        FOR 0 <= i < x:
            TD_P[x-i-1] = TD_L[x-i]
            RD_L[x-i] = RD_P[x-i-1]
        TRD_P[0] = TD_L[0]
        RD_L[0] = RRD_P[0]
    IF y < 31:
        FOR 0 <= j < (31-y):
            TD_P[y+j+1] = TD_L[y+j]
            RD_L[y+j] = RD_P[y+j+1]
        TRD_P[1] = TD_L[31]
        RD_L[31] = RRD_P[1]

```

Pseudo code for two Lane repair in **TD_P[63:32]** (**RD_P[63:32]**) ($32 \leq x, y \leq 63$) (this does not apply to x32 Advanced Package Link):

```

IF failure occurs in TD_P[x], TD_P[y] AND (x < y):
    IF x > 32:
        FOR 0 <= i < x-32:
            TD_P[x-i-1] = TD_L[x-i]
            RD_L[x-i] = RD_P[x-i-1]
        TRD_P[2] = TD_L[32]
        RD_L[32] = RRD_P[2]
    IF y < 63:
        FOR 0 <= j < (63-y):
            TD_P[y+j+1] = TD_L[y+j]
            RD_L[y+j] = RD_P[y+j+1]
        TRD_P[3] = TD_L[63]
        RD_L[63] = RRD_P[3]

```

Shown in Figure 4-16 is an example of two (physical Lanes 25 and 26) Lane remapping. Figure 4-17 shows the circuit implementation. Both Transmitter and Receiver must apply the required remapping.

Figure 4-16. Example of Two Lane failure remapping

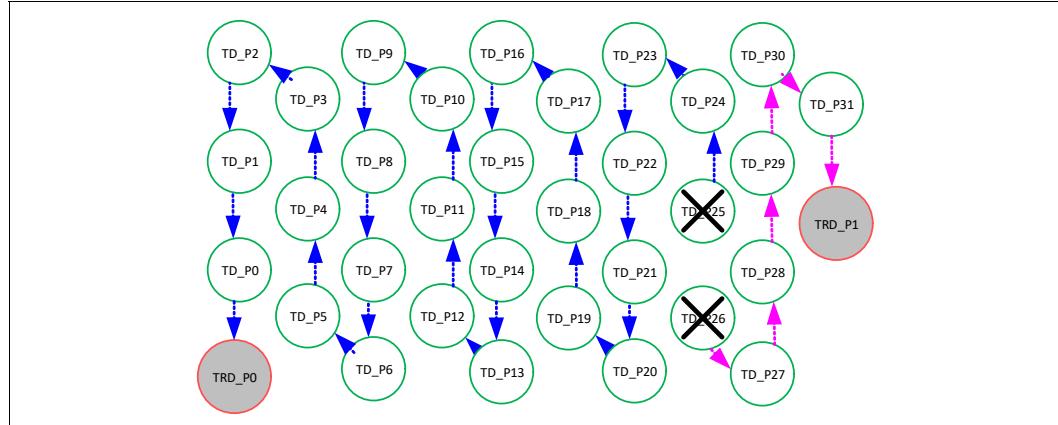
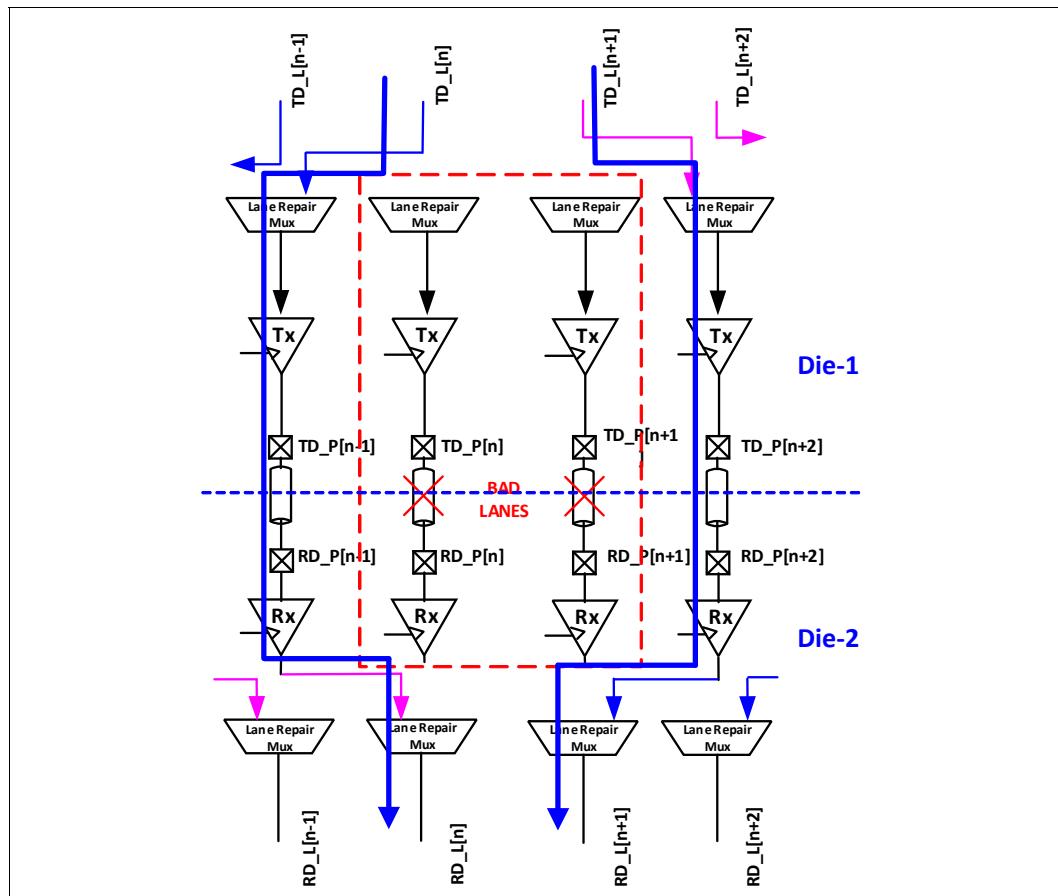


Figure 4-17. Example of Two Lane remapping implementation



4.3.3.3 Single Lane repair with Lane reversal

For repair with Lane reversal (see Section 4.3.2) Transmitter side remapping is reversed to preserve shifting order for Receiver side remapping.

4.3.3.3.1 x64 Advanced Package Pseudo Codes

Pseudo code for one Lane failure in `TD_P[31:0]` (`RD_P[32:63]`) ($0 \leq x \leq 31$):

```

IF failure occurs in TD_P[x]:
  IF x > 0:
    FOR 0 <= i < x:
      TD_P[x-i-1] = TD_L[63-x+i]
      RD_L[63-x+i] = RD_P[63-x+i+1]
    TRD_P[0] = TD_L[63]
    RD_L[63] = RRD_P[3]
  
```

Pseudo code for one Lane failure in `TD_P[63:32]` (`RD_P[0:31]`) ($32 \leq x \leq 63$):

```

IF failure occurs in TD_P[x]:
  IF x > 32:
    FOR 0 <= i < x-32:
      TD_P[x-i-1] = TD_L[63-x+i]
      RD_L[63-x+i] = RD_P[63-x+i+1]
    TRD_P[2] = TD_L[31]
    RD_L[31] = RRD_P[1]
  
```

4.3.3.3.2 x32 Advanced Package Pseudo Codes

Pseudo code for one-Lane failure in `TD_P[31:0]` (`RD_P[0:31]`) ($0 \leq x \leq 31$):

```

IF failure occurs in TD_P[x]:
  IF x > 0:
    FOR 0 <= i < x:
      TD_P[x-i-1] = TD_L[31-x+i]
      RD_L[31-x+i] = RD_P[31-x+i+1]
    TRD_P[0] = TD_L[31]
    RD_L[31] = RRD_P[1]
  
```

4.3.3.4 Two Lane repair with Lane reversal

For repair with Lane reversal (see [Section 4.3.2](#)) Transmitter side remapping is reversed to preserve shifting order for Receiver side remapping.

4.3.3.4.1 x64 Advanced Package Pseudo Codes

Pseudo code for two Lane failure in `TD_P[31:0]` (`RD_P[32:63]`) ($0 \leq x \leq 31$):

```

IF failure occurs in TD_P[x], TD_P[y] AND ( $x < y$ ):
    IF  $x > 0$ :
        FOR  $0 \leq i < x$ :
            TD_P[x-i-1] = TD_L[63-x+i]
            RD_L[63-x+i] = RD_P[63-x+i+1]
        TRD_P[0] = TD_L[63]
        RD_L[63] = RRD_P[3]
        IF  $y < 31$ :
            FOR  $0 \leq j < (31-y)$ :
                TD_P[y+j+1] = TD_L[63-y-j]
                RD_L[63-y-j] = RD_P[63-y-(j+1)]
        TRD_P[1] = TD_L[32]
        RD_L[32] = RRD_P[2]

```

Pseudo code for two-Lane failure in `TD_P[63:32]` (`RD_P[0:31]`) ($32 \leq x \leq 63$):

```

IF failure occurs in TD_P[x], TD_P[y] AND ( $x < y$ ):
    IF  $x > 32$ :
        FOR  $0 \leq i < x-32$ :
            TD_P[x-i-1] = TD_L[63-x+i]
            RD_L[63-x+i] = RD_P[63-x+(i+1)]
        TRD_P[2] = TD_L[31]
        RD_L[31] = RRD_P[1]
        IF  $y < 63$ :
            FOR  $0 \leq j < (63-y)$ :
                TD_P[y+j+1] = TD_L[63-y-j]
                RD_L[63-y-j] = RD_P[63-y-(j+1)]
        TRD_P[3] = TD_L[0]
        RD_L[0] = RRD_P[0]

```

4.3.3.4.2 x32 Advanced Package Pseudo Codes

Pseudo code for two-Lane failure in TD_P[31:0] (RD_P[0:31]) ($0 \leq x \leq 31$):

```

IF failure occurs in TD_P[x], TD_P[y] AND (x < y):
    IF x > 0:
        FOR 0 <= i < x:
            TD_P[x-i-1] = TD_L[31-x+i]
            RD_L[31-x+i] = RD_P[31-x+i+1]
        TRD_P[0] = TD_L[31]
        RD_L[31] = RRD_P[1]
        IF y < 31:
            FOR 0 <= j < (31-y):
                TD_P[y+j+1] = TD_L[31-y-j]
                RD_L[31-y-j] = RD_P[31-y-(j+1)]
            TRD_P[1] = TD_L[0]
            RD_L[0] = RRD_P[0]

```

4.3.4 Clock and Track Lane remapping

The specification supports remapping of one broken Lane for **TCKP_P/RCKP_P**, **TCKN_P/RCKN_P** and **TTRK_P/RTRK_P** physical Lanes. The repair scheme is shown in [Figure 4-18](#). Clock Lane remapping allows repair of single Lane failure for both differential and pseudo-differential implementation of the clock Receiver. The circuit details are shown in [Figure 4-19](#) and [Figure 4-20](#) for differential and pseudo-differential clock Receivers respectively.

After a Lane is remapped, the Transmitter is tri-stated. The Receiver of the physical redundant (**RRDCK_P**) Lane is disabled.

Figure 4-18. Clock and Track repair

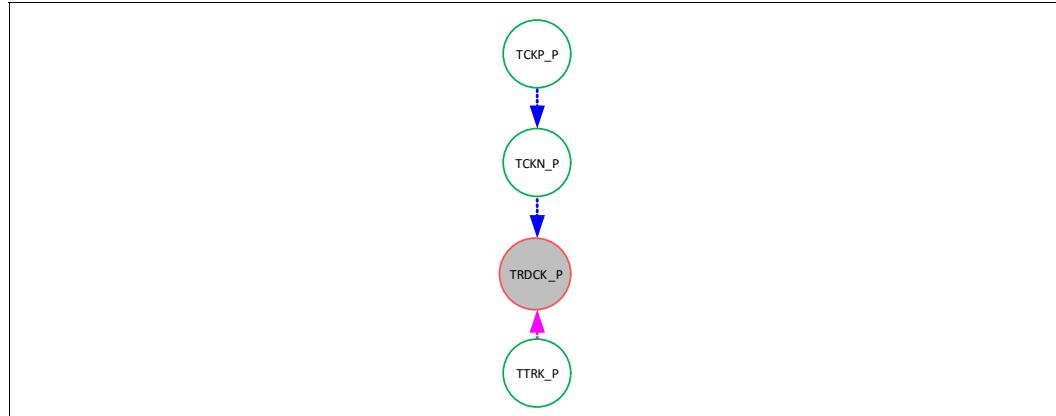


Figure 4-19. Clock and track repair: Differential Rx

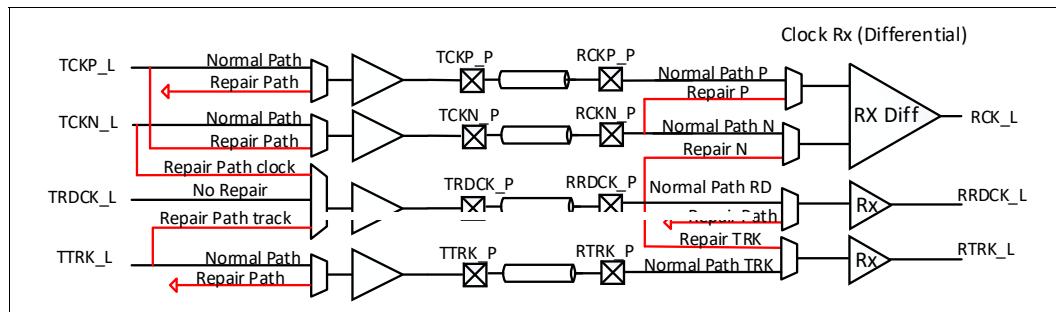
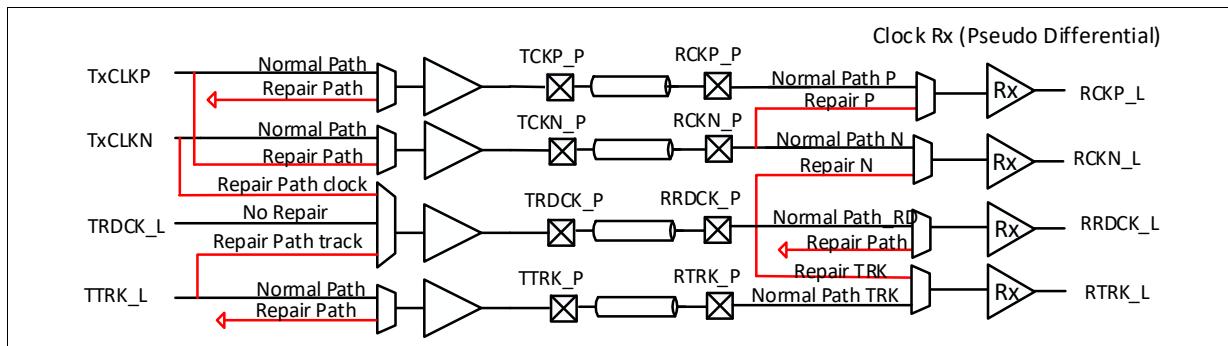


Figure 4-20. Clock and track repair: Pseudo Differential Rx



4.3.5 Clock and Track Lane repair implementation

Pseudo code for Clock and Track Lane repair:

```

IF failure occurs in TCKP_P:
    TCKN_P = TCKP_L AND TRDCK_P = TCKN_L
    RCKP_L = RCKN_P AND RCKN_L = RRDCK_P
ELSE IF failure occurs on TCKN_P:
    TRDCK_P = TCKN_L
    RCKN_L = RRDCK_P
ELSE IF failure occurs in TTRK_P:
    TRDCK_P = TTRK_L
    RTRK_L = RRDCK_P

```

The implementation of Clock and Track Lane remapping is shown in [Figure 4-21\(a\)](#), [Figure 4-21\(b\)](#) and [Figure 4-21\(c\)](#) respectively. The corresponding circuit level details of remapping implementation are shown in [Figure 4-22](#), [Figure 4-23](#) and [Figure 4-24](#).

Note that the both Transmitter and Receiver on CKRD Lane are required during detection phase and can be tri-stated and turned off if not used for repair.

Figure 4-21. Clock and Track Lane repair scheme

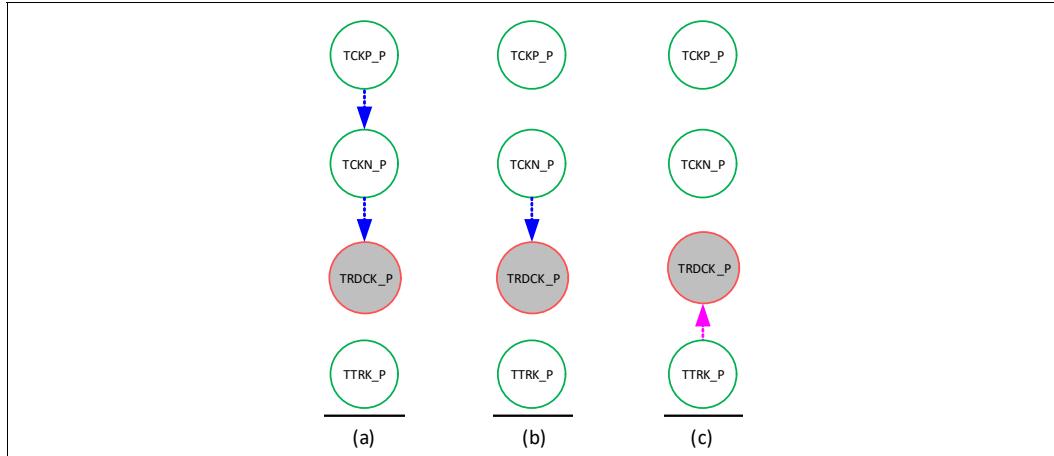


Figure 4-22. Clock and track repair: CKP repair

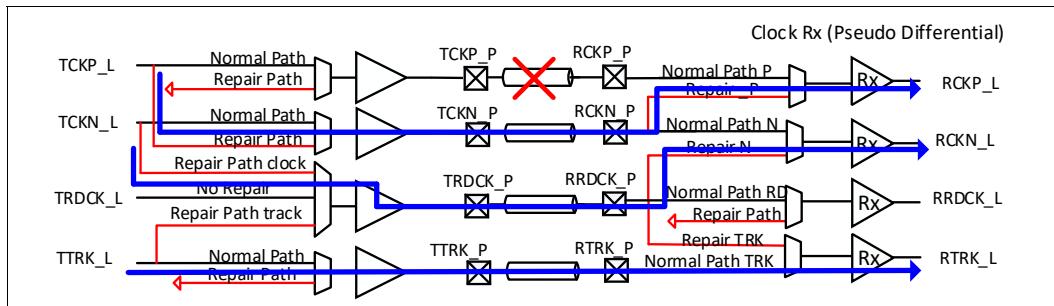
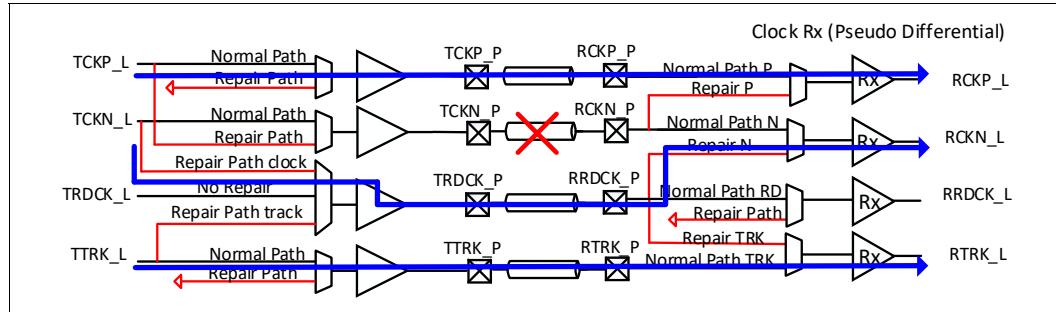
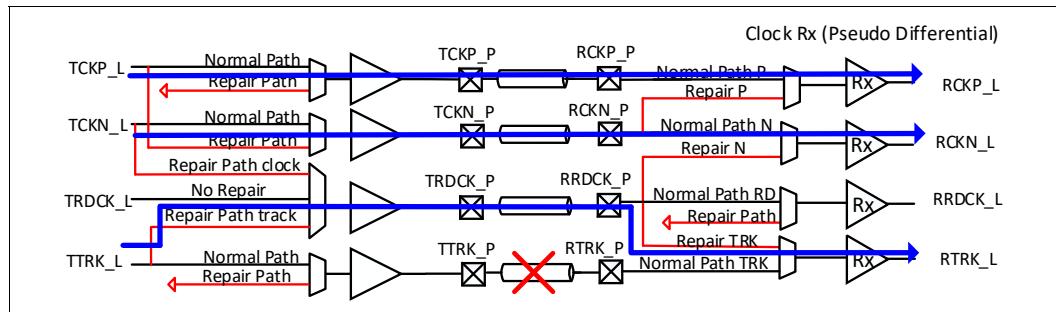


Figure 4-23. Clock and track repair: CKN repair**Figure 4-24. Clock and track repair: Track repair**

4.3.6 Valid Repair and implementation

Valid Lane has a dedicated redundant lane. If a failure is detected on the Valid physical Lane, redundant Valid physical Lane is used to send Valid. Valid failure detection and repair is performed during Link initialization and Training (Section 4.5.3.3.4).

Figure 4-25 shows the normal path for Valid and redundant valid Lanes. Figure 4-26 shows the repair path for Valid Lane failure.

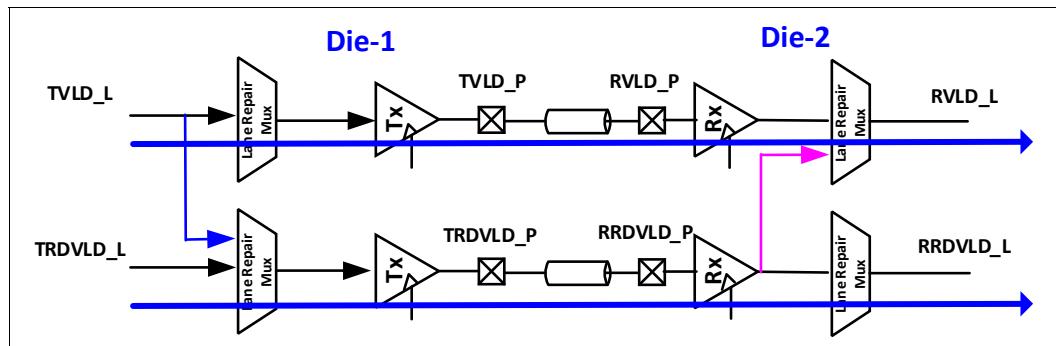
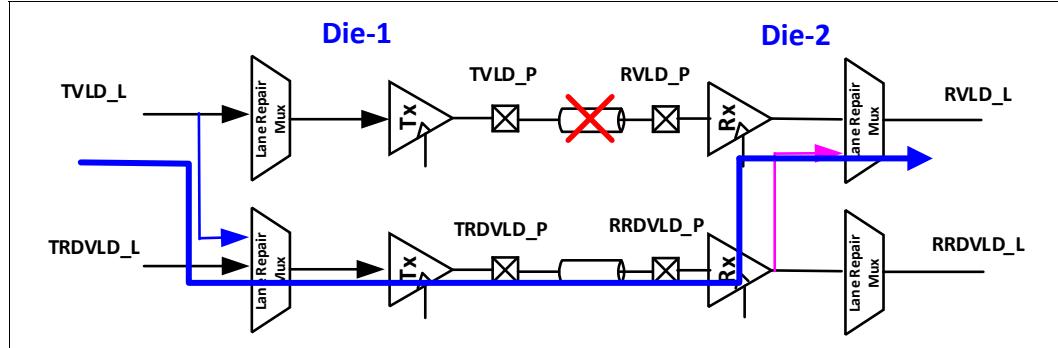
Figure 4-25. Valid repair: Normal Path

Figure 4-26. Valid Repair: Repair Path

4.3.7 Width Degrade in Standard Package Interfaces

In the case of x16 Standard Package modules where Lane repair is not supported, resilience against faulty Lanes is provided by configuring the Link to a x8 width (Logical Lanes 0 to 7 or Logical Lanes 8 to 15, which exclude the faulty Lanes). For example, if one or more faulty Lanes are in logical Lane 0 to 7, the Link is configured to x8 width using logical Lanes 8 to 15. The configuration is done during Link initialization or retraining. Transmitters of the disabled Lanes are tri-stated and Receivers are disabled.

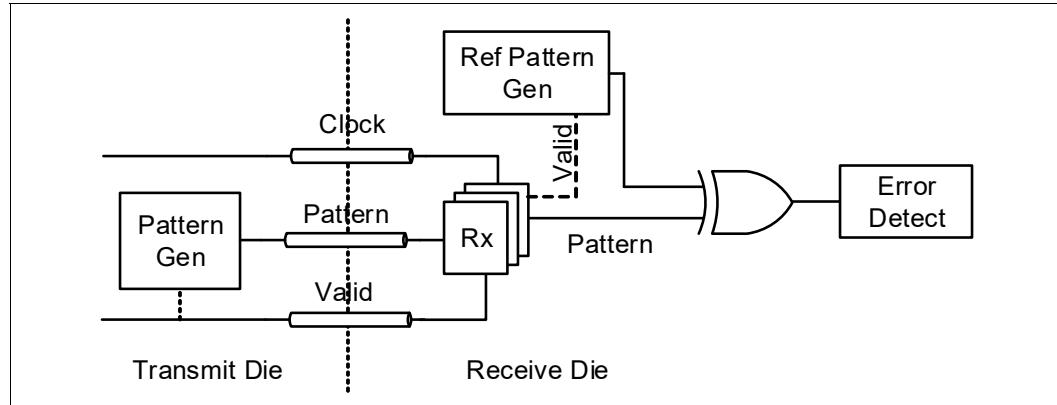
In the case of x8 Standard Package modules, resilience against faulty Lanes is provided by configuring the Link to a x4 width (Logical Lanes 0 to 3 or Logical Lanes 4 to 7, which exclude the faulty Lanes). The configuration is done during Link initialization or retraining. Transmitters of the disabled Lanes are tri-stated and Receivers are disabled.

Figure 4-5 shows the byte to Lane mapping for a width degraded x8 interface

4.4 Data to Clock Training and Test Modes

Note: Sideband commands will be identified as {SB command}.

Figure 4-27 shows the infrastructure for interface training and testing. The Transmit Die and Receive Die implement the same Linear Feedback Shift Register (LFSR) described in Section 4.4.1. The pattern sent from the Transmitter along with forwarded clock and Valid is compared with locally generated reference pattern. Both transmit and receive pattern generators must start and advance in sync. The compare circuitry checks for matching data each UI. Any mismatch between the received pattern and pattern predicted by the local pattern generator is detected as an error.

Figure 4-27. Test and training logic

The receive die must implement two types of comparison schemes

- **Per-Lane comparison:** Per-Lane comparison is used to identify the number of failing Lanes between the two dies. Any mismatch, above the set threshold between the received pattern and the expected pattern on each Lane, will set an internal error detect bit for each Lane. Once a pattern mismatch on a particular Lane is found, this error bit is set for the remainder of the test. [Figure 4-28](#) illustrates Per-Lane comparison mode. This mode will indicate per-Lane errors within the module ($N = 68$ ($64 + 4$ RD) for a x64 Advanced Package Module, $N = 34$ ($32 + 2$ RD) for a x32 Advanced Package Module, $N = 16$ for a x16 Standard Package Module, and $N = 8$ for a x8 Standard Package Module, respectively). The per-Lane comparison results can be read via sideband.
- **Aggregate comparison:** In this mode, pattern mismatches each UI on any Lane within the module are accumulated into a 16-bit error counter. The Lane errors are ORed to generate a module-level error and counted as shown in [Figure 4-29](#). This scheme can be used for module level margin and BER.

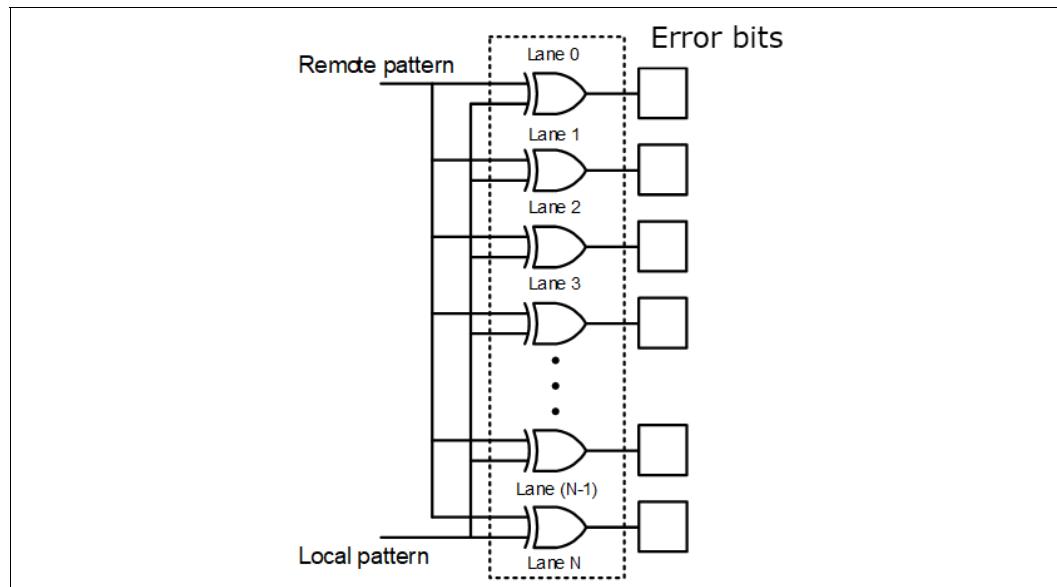
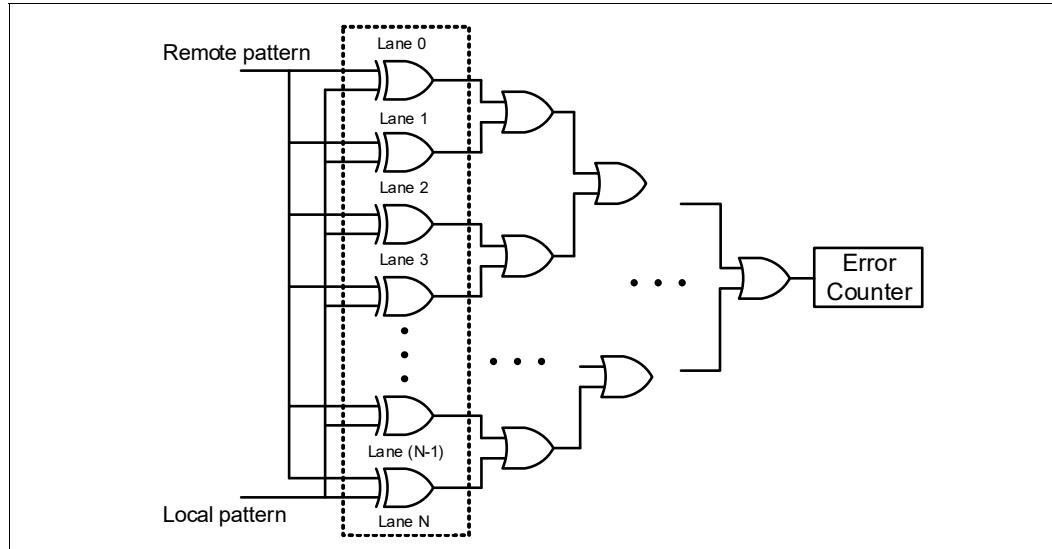
Figure 4-28. Lane failure detection

Figure 4-29. All Lane error detection

4.4.1 Scrambling and training pattern generation

A Linear feedback shift register (LFSR) is defined for scrambling and test pattern generation.

The LFSR uses the same polynomial as PCIe: $G(X)=X^{23} + X^{21} + X^{16} + X^8 + X^5 + X^2 + 1$. Each Transmitter is permitted to implement a separate LFSR for scrambling and pattern generation. Each Receiver is permitted to implement a separate LFSR using the same polynomial for de-scrambling and pattern comparison. The implementation is shown in [Figure 4-30](#). The seed of the LFSR is Lane dependent, and based on the Logical Lane number and the seed value for Lane number is modulo 8 as shown in [Table 4-4](#).

Alternatively, implementations can choose to implement one LFSR with different tap points for multiple Lanes as shown in [Figure 4-31](#). This is equivalent to individual LFSR per-Lane with different seeds.

Table 4-4. LFSR seed values

Lane	Seed
0	23'h1DBFBC
1	23'h 0607BB
2	23'h1EC760
3	23'h18C0DB
4	23'h010F12
5	23'h19CF9
6	23'h0277CE
7	23'h1BB807
RD0, RD2 ^a	23'h18C0DB
RD1, RD3 ^b	23'h010F12

a. Same as Lane 3. These are not currently used.

b. Same as Lane 4. These are not currently used.

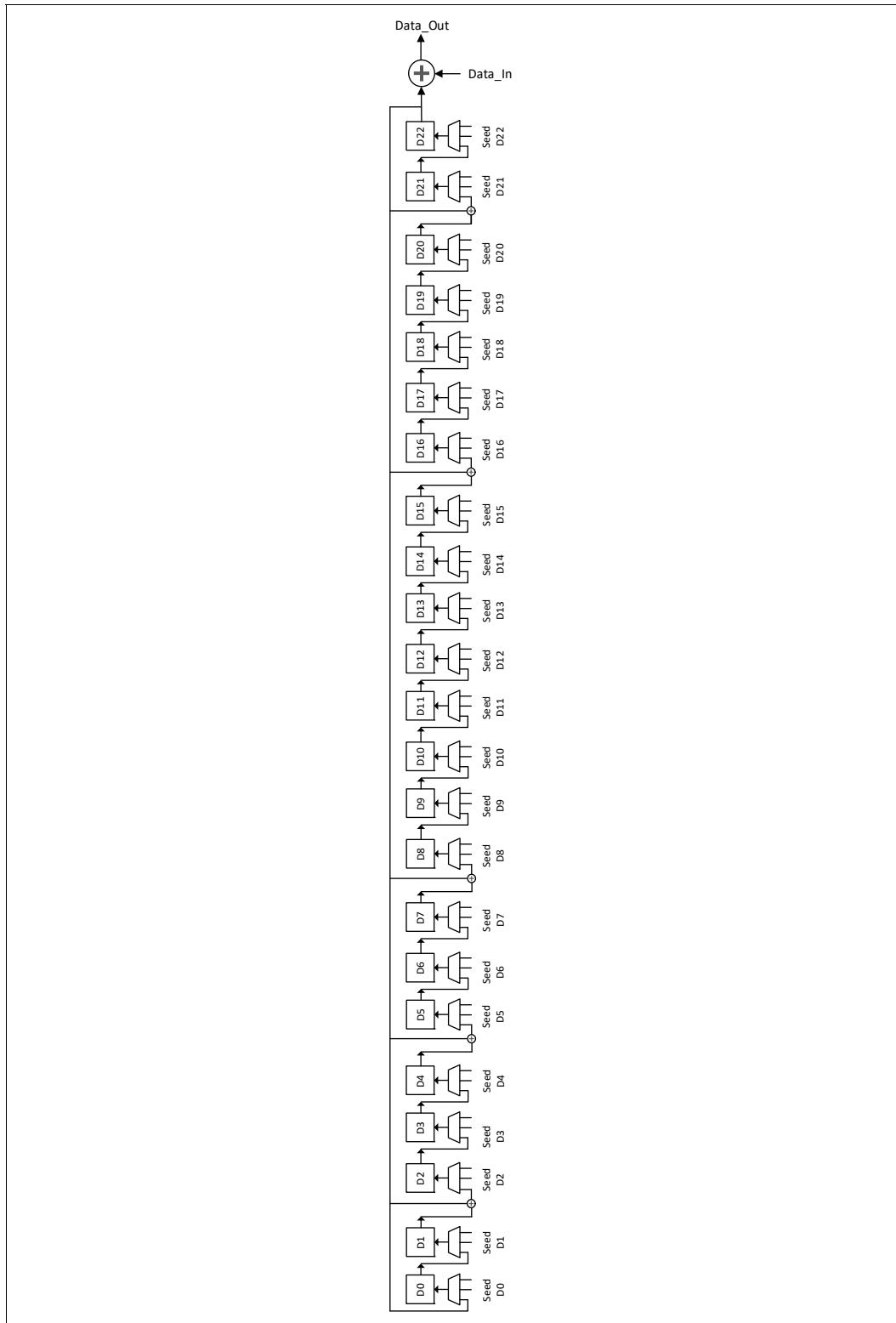
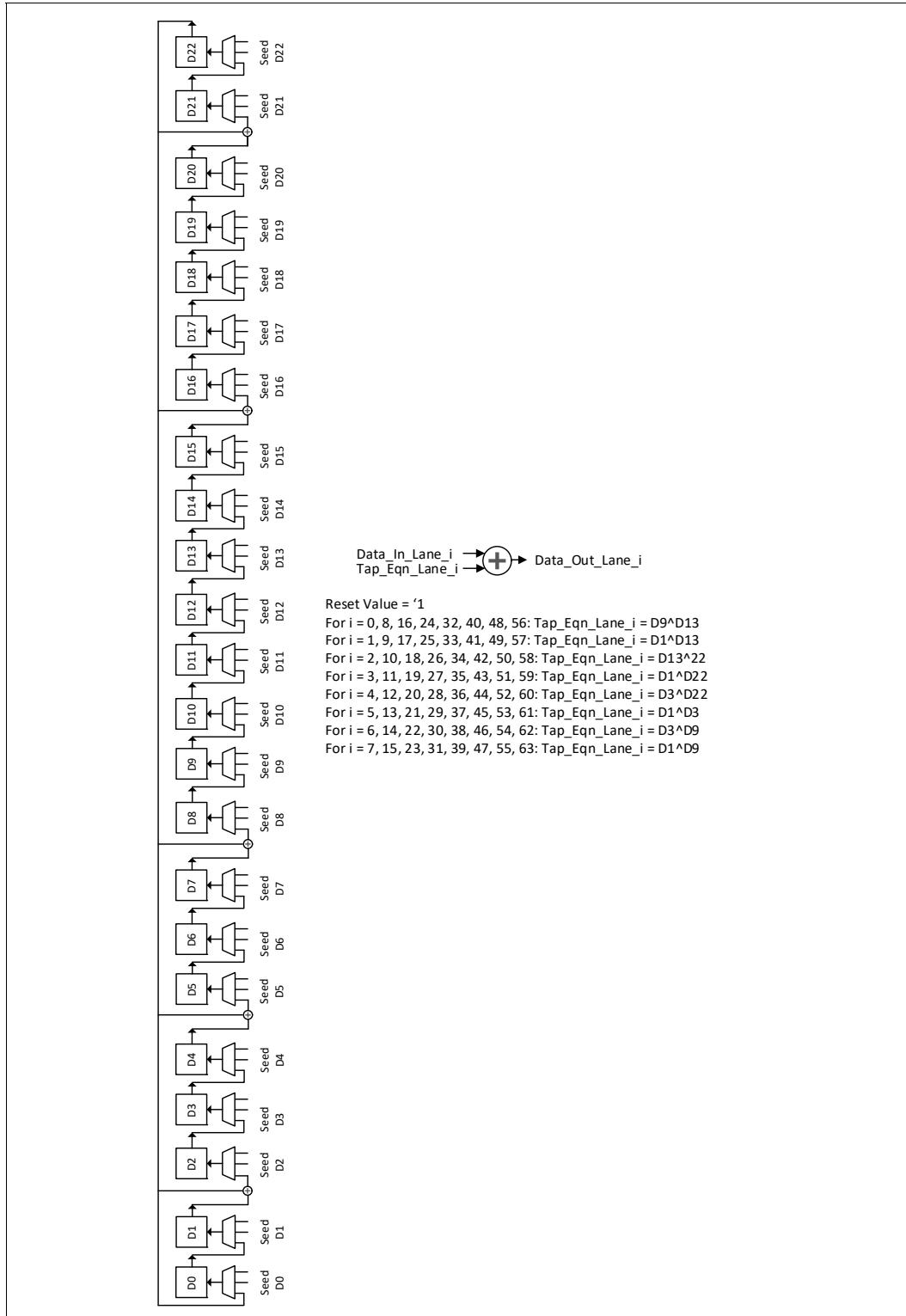
Figure 4-30. LFSR implementation

Figure 4-31. LFSR alternate implementation

4.5 Link Initialization and Training

Link initialization and training are described at a Module level. The description of terms are used in this section are as follows:

- UCIe Module Partner: A UCIe Module Partner is the corresponding UCIe Module on the remote UCIe Die to which the UCIe Module connects. For example, if two UCIe Dies A and B are connected in the standard Die rotate configuration by a 2-module UCIe Link, Modules 0 and 1; UCIe Die A's Module 0 (UCIe Module A0) is connected to UCIe Die B's Module 1 (UCIe Module B1); then A0 is the UCIe Module Partner of B1 and vice-versa.
- Phase Interpolator (PI) in this specification is used to refer any method of generating and selecting sampling clock phase.
- Timeout: Every state except RESET, Active, L1/L2, and TRAINERROR in the Link Training State Machine has a residency timeout of 8 ms. For states with sub-states, the timeout is per sub-state (i.e., if Physical Layer is in a state for greater than 8 ms, then it transitions to TRAINERROR and eventually to RESET). Physical Layer sideband handshakes for RDI state transitions with remote Link partner also timeout after 8 ms. The timeout counters must be reset if a sideband message with "Stall" encoding is received. All timeout values specified are -0% and +50% unless explicitly stated otherwise. All timeout values must be set to the specified values after Domain Reset exit. All counter values must be set to the specified values after Domain Reset.
- Electrical idle is described in [Section 5.12](#).
- When Management Transport protocol is supported over the sideband, the term SB_MGMT_UP refers to an internally maintained flag in the PHY that indicates whether the Management Transport path has been successfully initialized with the partner chiplet. If the flag is cleared to 0, the Management Transport path is not up. If the flag is set to 1, the Management Transport path is up. The flag is cleared on a Management Reset or when a Heartbeat timeout is detected (see [Section 8.2.5.1.3](#)). The flag is set after successful completion of the initialization phase of Management Transport path setup as described in [Section 8.2.3.1.2](#).

When training during Link Initialization (i.e., Physical Layer transitions out of RESET state), hardware is permitted to attempt training multiple times:

- Triggers for initiating Link Training for Management Transport path setup on the sideband are:
 - Software writes 1 to the Retrain Link bit in the Sideband Management Port Structure register (see [Section 8.1.3.6.2.1](#))
 - HW-autonomous trigger by the Management Port Gateway to automatically train the Management Transport path on the sideband without SW intervention, as occurs after Management Reset
 - SBINIT pattern (two consecutive iterations of 64-UI clock pattern and 32-UI low) is observed on any sideband Receiver clock/data pair, when the SB_MGMT_UP flag is cleared to 0
- The triggers for initiating Link Training for the mainband are:
 - Software writes 1 to Start UCIe Link Training bit in UCIe Link Control register in the UCIe Link DVSEC (see [Section 9.5.1.5](#))
 - When Management Transport protocol is supported on the UCIe link, software writes 1 to the Retrain Link bit in the Management Port Structure register of either the Sideband or Mainband Management Port that is associated with the UCIe link (see [Section 8.1.3.6.2.1](#))
 - Adapter triggers Link Training on the RDI (RDI status is Reset and there is a NOP to Active transition on the state request)

- If [Management Transport protocol is not supported] OR [the SB_MGMT_UP flag is cleared to 0], the SBINIT pattern (two consecutive iterations of 64-UI clock pattern and 32-UI low) is observed on any sideband Receiver clock/data pair
- If [Management Transport protocol is supported] AND [the SB_MGMT_UP flag is set to 1], the {SBINIT done req} sideband message was received from the module partner

If hardware fails training after an implementation-specific number of attempts, the hardware must transition to RESET and wait for a subsequent Link Training trigger. Physical Layer must escalate a fatal error to the D2D Adapter on the RDI if mainband Software-triggered or RDI-triggered Link training fails or there is a Link-up-to-Link-down transition due to a Physical Layer timeout.

Throughout this section, references to mainband transmitter and receiver behavior are called out in various state machine states. These references do not apply when the port does not have a mainband (i.e., the port is a sideband-only port without a physically present mainband, or the port is a UCIe-S port with only the sideband used. In the latter scenario, the mainband transmitters are held in tri-state throughout and the mainband receivers are disabled.

4.5.1 Link Training Basic Operations

Some basic operations in Link training are defined in this section. These will be used in mainband initialization, training and margining.

4.5.1.1 Transmitter initiated Data to Clock Point Test

In this mode, the Transmitter initiates the data to clock training on all Lanes in the module at a single PI phase. When not performing the actions relevant to this state:

- Data, Valid, and Track Transmitters drive low
- Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking)
- Data, Valid, and Clock Receivers are enabled
- Track Receiver is permitted to be disabled

The sequence of steps for this test are as follows for each UCIe Module of the UCIe Link:

1. The UCIe Module sets up the Transmitter parameters (shown in [Table 4-5](#)), sends a {Start Tx Init D to C point test req} sideband message to its UCIe Module Partner, and waits for a response. The data field of this message includes the required parameters, shown in [Table 4-5](#). The Receiver on the UCIe Module Partner must enable the pattern comparison circuits to compare incoming mainband data to the locally generated expected pattern. Once the data to clock training parameters for its Receiver are setup, the UCIe Module Partner responds with a {Start Tx Init D to C point test resp} sideband message.
2. The UCIe Module resets the LFSR (scrambler) on its mainband Transmitters and sends the {LFSR clear error req} sideband message. The UCIe Module Partner resets the LFSR and clears any prior compare results on its mainband Receivers and responds with {LFSR clear error resp} sideband message.
3. The UCIe Module sends the pattern (selected through “Tx Pattern Generator Setup”) for the selected number of cycles (“Tx Pattern Mode Setup”) on its mainband Transmitter.
4. The UCIe Module Partner performs the comparison on its Receivers for each UI during the pattern transmission based on “Rx compare setup” and logs the results.

5. The UCIe Module requests its UCIe Module Partner for the logged results in [Step 4](#) by sending {Tx Init D to C results req} sideband message. The UCIe Module Partner stops comparison on its mainband Receivers and responds with the logged results {Tx Init D to C results resp} sideband message.
6. The UCIe Module stops sending the pattern on its Transmitters and sends the {End Tx Init D to C point test req} sideband message and the UCIe Module Partner responds with {End Tx Init D to C point test resp}. When a UCIe Module has received the {End Tx Init D to C point test resp} sideband message, the corresponding sequence has completed.

Table 4-5. Data-to-Clock Training Parameters^a

Parameter	Sideband Message Field	Options
Clock sampling /PI phase control	Clock Phase control at Transmitter	Eye Center found by training
		Eye Left Edge found by training
		Eye Right Edge found by training
Tx Pattern Generator Setup	Data Pattern (for Data Lanes)	LFSR Pattern for Data Lanes
		Per-Lane ID pattern for Data Lanes as defined in Table 4-7
	Valid Pattern (for Valid Lanes)	VALTRAIN pattern four 1s followed by four 0s
Tx Pattern Mode Setup	Pattern Mode	Burst Mode: Uses Burst Count/Idle Count/Iteration Count
		Continuous Mode: Uses Burst count to indicate the number of UI of transmission. Idle Count = 0, Iteration Count = 1
Rx Compare Setup	Maximum Comparison error threshold	Maximum comparison error threshold
	Comparison Mode	Per-Lane or aggregate error compare

- a. See [Table 7-11](#) for the encodings. See also the registers in [Section 9.5.3.26](#) and [Section 9.5.3.27](#). See also the Implementation Note below this table.

IMPLEMENTATION NOTE

The Training Setup 1 and Training Setup 2 registers (see [Section 9.5.3.26](#) and [Section 9.5.3.27](#), respectively) are applicable for compliance or debug. For regular operation, implementations must follow the comparison mode, iteration or UI count specified in the corresponding states (for example, [Section 4.5.3.4.8](#) specifies 4K UI of continuous mode LFSR pattern and total (aggregate) error count); and because these patterns are fixed, Rx is permitted to ignore Burst Count/Idle Count/Iteration Count values in the sideband messages for regular operation. Training Setup 3 and Training Setup 4 registers (see [Section 9.5.3.28](#) and [Section 9.5.3.29](#), respectively) are applicable for regular operation as well as compliance and debug.

Additionally, implementation notes for the parameters in [Table 4-5](#):

- Clock sampling/PI phase control: For Tx-initiated Data to Clock point tests, this parameter is for informational purposes only. For Rx-initiated Data to Clock point tests, the Transmitter sets the PI phase for the requested setting (Eye Left edge, Eye Right edge or Eye Center) based on the best known values from the most recent training for the corresponding speed. For the states of MBTRAIN.VALVREF and MBTRAIN.DATAVREF, this would be the best known values at the Transmitter at 4 GT/s. This field is not applicable and is ignored for Data to Clock eye sweep tests.
- Tx Pattern Generator Setup and Tx Pattern Mode Setup:
 - For Continuous mode LFSR pattern, “Burst Count” indicates the number of UI of transmission (for example, it is 4096 (or 4K) in [Section 4.5.3.4.8](#)). “Idle Count” is always 0 and “Iteration Count” is always 1.
 - For Continuous mode Per-Lane ID pattern, “Burst Count” indicates the number of UI of transmission (e.g., it is 2048 for 128 iterations of the 16 bit pattern). “Idle Count” is always 0 and “Iteration Count” is always 1.
 - For Burst mode, the transmission is a “Burst Count” number of UI followed by an “Idle Count” number of UI of 0 repeating for an “Iteration Count” number of times. In the case of Per-Lane ID pattern, the “Burst Count” must be a multiple of the pattern size in UI. Burst mode is currently not used for regular operation of Link Training.
 - VALTRAIN pattern is used for training of the Valid lane. In Continuous mode, “Burst Count” indicates the number of UI of transmission (for example, it is 1024 for 128 iterations of the 8-bit pattern); “Idle Count” is always 0 and “Iteration Count” is always 1. In Burst mode, the transmission is a “Burst Count” number of UI followed by an “Idle Count” number of UI of 0 repeating for an “Iteration Count” number of times; the “Burst Count” must be a multiple of the pattern size in UI.

4.5.1.2 Transmitter initiated Data to Clock Eye Width Sweep

In this mode, the Transmitter initiates the data to clock training on all Lanes in the module and sweeps through the sampling PI phases. This mode can also be used to check system margin. When not performing the actions relevant to this state:

- Data, Valid, and Track Transmitters drive low
- Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking)
- Data, Valid, and Clock Receivers are enabled
- Track Receiver is permitted to be disabled

The sequence of steps for this test are as follows:

1. The UCIe Module sets up the Transmitter parameters (shown in [Table 4-5](#)), sends a {Start Tx Init D to C eye sweep req} sideband message to its UCIe Module Partner, and then waits for a response. The data field of this message includes the required parameters, shown in [Table 4-5](#). The Receiver on the UCIe Module Partner must enable the pattern comparison circuits to compare incoming mainband data to the locally generated expected pattern. Once the data to clock training parameters for its Receiver are setup, the UCIe Module Partner responds with a {Start Tx Init D to C eye sweep resp} sideband message.
2. The UCIe Module resets the LFSR (scrambler) on its mainband Transmitters and sends the {LFSR clear error req} sideband message. The UCIe Module Partner resets the LFSR and clears any prior compare results on its mainband Receivers and responds with {LFSR clear error resp} sideband message.
3. The UCIe Module sends the pattern (selected through "Tx Pattern Generator Setup") for the selected number of cycles ("Tx Pattern Mode Setup") on its mainband Transmitter.
4. The UCIe Module Partner performs comparison on its Receivers for each UI during the pattern transmission based on "Rx compare setup" and logs the results.
5. The UCIe Module requests its UCIe Module Partner for the logged results in [Step 4](#) by sending {Tx Init D to C results req} sideband message. The UCIe Module Partner stops comparison on its mainband Receivers and responds with the logged results {Tx Init D to C results resp} sideband message.
6. The UCIe Module sweeps through the PI phases on its forwarded clock Transmitter each time repeating Step 2 through Step 5 to find the passing PI phase range. The sweep steps and range are implementation specific.
7. The UCIe Module stops sending the pattern on its Transmitters and sends the {End Tx Init D to C eye sweep req} sideband message and the UCIe Module Partner responds with {End Tx Init D to C eye sweep resp}. When a UCIe Module has received the {End Tx Init D to C point eye sweep resp} sideband message, the corresponding sequence has completed.

4.5.1.3 Receiver initiated Data to Clock point test

In this mode, the Receiver initiates the data to clock training on all Lanes in the module at a single PI phase. When not performing the actions relevant to this state:

- Data, Valid, and Track Transmitters drive low
- Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking)
- Data, Valid, and Clock Receivers are enabled
- Track Receiver is permitted to be disabled

The sequence of steps for this test are as follows:

1. The UCIe Module enables the pattern comparison circuits to compare incoming mainband data to the locally generated expected pattern, sets up the Receiver parameters (shown in [Table 4-5](#)), sends a {Start Rx Init D to C point test req} sideband message to its UCIe Module Partner, and then waits for a response. The data field of this message includes the required parameters, shown in [Table 4-5](#). Once the data to clock training parameters for its Transmitter are setup, the UCIe Module Partner responds with a {Start Rx Init D to C point test resp} sideband message.
2. The UCIe Module Partner resets the LFSR (scrambler) on its mainband Transmitters and sends sideband message {LFSR clear error req}. The UCIe Module resets the LFSR and clears any prior compare results on its mainband Receivers and responds with {LFSR clear error resp} sideband message.

3. The UCIe Module Partner sends the pattern (selected through "Tx Pattern Generator Setup") for the selected number of cycles ("Tx Pattern Mode Setup") on its mainband Transmitter.
4. The UCIe Module performs the comparison on its mainband Receivers for each UI during the pattern transmission based on "Rx compare setup" and logs the results.
5. The UCIe Module Partner sends a sideband message {Rx Init D to C Tx count done req} sideband message once the pattern count is complete. The UCIe Module, stops comparison and responds with the sideband message {Rx Init D to C Tx count done resp}. The UCIe Module can now use the logged data for its Receiver Lanes.
6. The UCIe Module sends an {End Rx Init D to C point test req} sideband message and the UCIe Module Partner responds with an {End Rx Init D to C point test resp} sideband message. When a UCIe Module has received the {End Rx Init D to C point test resp} sideband message, the corresponding sequence has completed.

4.5.1.4 Receiver initiated Data to Clock Eye Width Sweep

In this mode, the Receiver initiates the data to clock training on all Lanes in the module at a single PI phase. When not performing the actions relevant to this state:

- Data, Valid, and Track Transmitters drive low
- Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking)
- Data, Valid, and Clock Receivers are enabled
- Track Receiver is permitted to be disabled

The sequence of steps for this test are as follows:

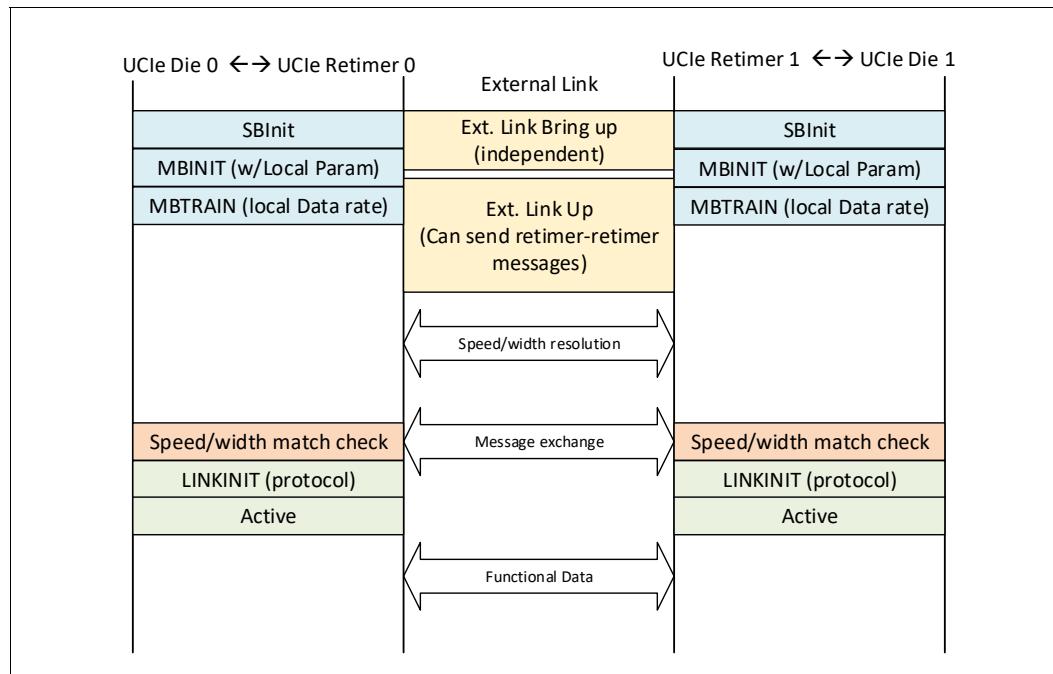
1. The UCIe Module enables the pattern comparison circuits to compare incoming mainband data to the locally generated expected pattern, sets up the Receiver parameters (shown in [Table 4-5](#)), sends a {Start Rx Init D to C eye sweep req} sideband message to its UCIe Module Partner, and then waits for a response. The data field of this message includes the required parameters, shown in [Table 4-5](#). The Transmitter on the UCIe Module Partner must be idle. Once the data to clock training parameters for its Transmitter are setup, the UCIe Module Partner responds with a {Start Rx Init D to C eye sweep resp} sideband message.
2. The UCIe Module Partner resets the LFSR (scrambler) on its mainband Transmitters and sends sideband message {LFSR clear error req}. The UCIe Module resets the LFSR and clears any prior compare results on its mainband Receivers and responds with an {LFSR clear error resp} sideband message.
3. The UCIe Module Partner sends the pattern (selected through "Tx Pattern Generator Setup") for the selected number of cycles ("Tx Pattern Mode Setup") on its mainband Transmitter.
4. The UCIe Module performs the comparison on its mainband Receivers for each UI during the pattern transmission based on "Rx compare setup" and logs the results.
5. The UCIe Module Partner requests the UCIe Module for the logged data to clock test results through a {Rx Init D to C results req} sideband message. The UCIe Module responds with the logged results for its mainband Receiver using the {Rx Init D to C results resp} sideband message. The UCIe Module Partner can determine if the pattern comparison at the PI phase passed or failed based on the results log.
6. The UCIe Module Partner sweeps through the PI phases on its forwarded clock each time it repeats [Step 2](#) through [Step 5](#) to find the passing PI phase range. The sweep pattern and range are implementation specific.

7. The UCIe Module Partner sends an {Rx Init D to C sweep done with results} sideband message with results for its mainband Transmitter. The UCIe Module can use the sweep results for its mainband Receivers.
8. The UCIe Module sends an {End Rx Init D to C eye sweep req} sideband message and the UCIe Module Partner responds with an {End Rx Init D to C eye sweep resp} sideband message. When a UCIe Module has received the {End Rx Init D to C eye sweep resp} sideband message, the corresponding sequence has completed.

4.5.2 Link Training with Retimer

The following diagram explains the initialization flow with UCIe Retimer. As shown in Figure 4-32, external and UCIe (UCIe Die to UCIe Retimer) Links are permitted to come up independently. When a UCIe Link trains up to local data rate and width, the remote UCIe information is requested over the external Link. If there is a data rate and width difference, each UCIe Link is permitted to be retrained to achieve a speed (data rate) and width match (if the UCIe Retimer requires this for operation, it must initiate Retraining from LinkSpeed state). This can happen multiple times during initial Link bring up or retraining. Once the UCIe Retimer has determined that the UCIe Link configuration is suitable for successful operation with remote Retimer partner, the UCIe Link proceeds to ACTIVE through protocol level Link initialization (LINKINIT).

Figure 4-32. Example Retimer bring up when performing speed/width match

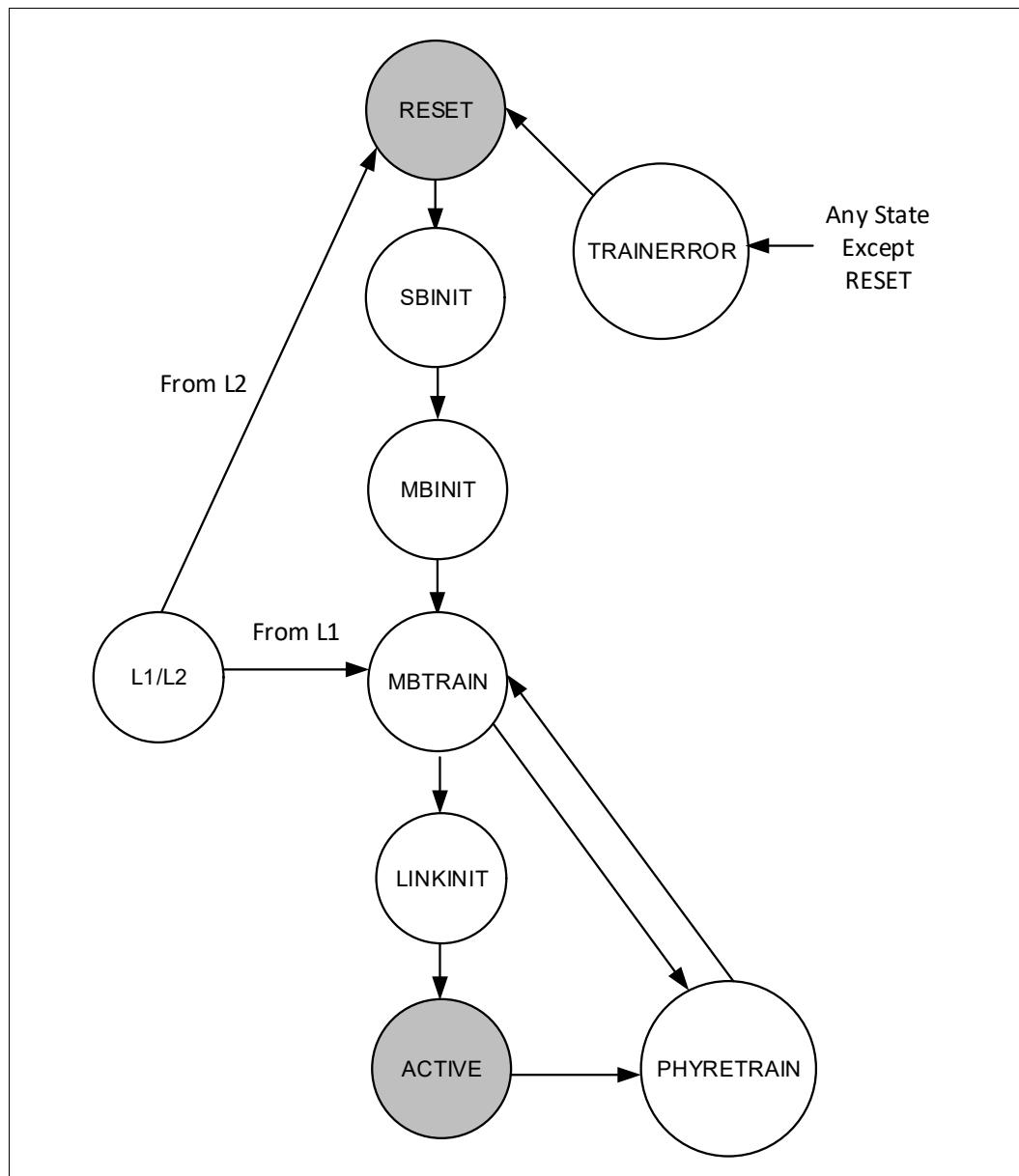


4.5.3 Link Training State Machine

A high level initialization flow is shown in Figure 4-33. A high level description of each state is shown in Table 4-6. The details and actions performed in each state are described in the following sections.

Table 4-6. State Definitions for Initialization

State	Description
RESET	This is the state following primary reset or exit from TRAINERROR.
SBINIT	Sideband initialization state where the sideband is detected, repaired (when applicable) and out of reset message is transmitted.
MBINIT	Following sideband initialization, Mainband (MB) is initialized at the lowest speed. Both dies perform on die calibration followed by interconnect repair (when applicable).
MBTRAIN	Mainband (Data, Clock and Valid signals) speed of operation is set to the highest negotiated data rate. Die-to-Die training of mainband is performed to center the clock with respect to Data.
LINKINIT	This state is used to exchange Adapter and Link management messages.
ACTIVE	This is the state in which transactions are sent and received.
L1/L2	Power Management state.
PHYRETRAIN	This state is used to begin the retrain flow for the Link during runtime.
TRAINERROR	State is entered when a fatal or non-fatal event occurs at any point during Link Training or operation.

Figure 4-33. Link Training State Machine

4.5.3.1 RESET

Physical Layer must remain in RESET for a minimum of 4 ms upon every entry to RESET, to allow PLLs to stabilize and any other Link Training initialization requirements to be met. The minimum conditions necessary to exit RESET are as follows:

- Power supplies are stable
- Sideband clock is available and running at 800 MHz
- If {
 - Physical Layer and Die to Die Adapter internal clocks are stable and available
 - Mainband clock speed is set to the slowest I/O data rate (2 GHz for 4 GT/s)
 - Local SoC/Firmware not keeping the Physical Layer in RESET
 - Link training trigger for the mainband has occurred (triggers are defined in the beginning of [Section 4.5](#))
} OR
 - {
 - Sideband Management Transport protocol is supported
 - SB_MGMT_UP = 0
 - Local SoC/Firmware is not keeping the sideband in RESET
 - Management Port Gateway indicates that it is ready for Management Transport path initialization
 - Link Training trigger for the Management Transport path has occurred (triggers are defined in the beginning of [Section 4.5](#))
}
- Data, Valid, Clock, and Track Transmitters are tri-stated
- Data, Valid, Clock, and Track Receivers are permitted to be disabled
- If [Management Transport protocol is not supported] OR [SB_MGMT_UP=0], Sideband Transmitters are held low
- Sideband Receivers are enabled

If [Management Transport protocol is supported] AND [SB_MGMT_UP=1], the Physical Layer is available for sideband packet (any sideband packets including Management Port messages) transmission/reception in the RESET state.

If [Management Transport protocol is not supported] OR [SB_MGMT_UP=0]:

- While the LTSM is in the RESET state, a Physical Layer implementation is permitted to handle any new or pending untransmitted sideband register access requests as Unsupported Requests, and if doing so, it must return a corresponding completion as well as end-to-end credit and RDI credit to the Adapter. Similarly, for register access completions, a Physical Layer implementation is permitted to discard any new or pending untransmitted completions and because completions do not consume a credit, there is no credit return back to the Adapter for these. While the LTSM is in the RESET state, an implementation may handle any new or pending untransmitted Vendor-defined messages in an implementation dependent manner. This could include dropping the message and logging an error.

4.5.3.2 Sideband Initialization (SBINIT)

In this state, the sideband (SB) interface is initialized and repaired (when applicable).

If Management Transport is supported AND the SB_MGMT_UP flag is set to 1:

- Initialization and repair of the sideband ([Step 1](#) through [Step 9](#) for Advanced Package and [Step 1](#) through [Step 7](#) for Standard Package as shown later in this section) are skipped, and only an SBINIT Done Req/Resp handshake is performed in this state (see [Step 10](#) for Advanced Package flow and [Step 8](#) for Standard Package flow as shown later in this section); otherwise, all the steps are traversed
- Physical Layer sideband is available for sideband packet (any sideband packets including Management Port messages) transmission/reception in this state

When not performing the actions relevant to this state:

- Data, Valid, Clock, and Track Transmitters remain tri-stated
- Data, Valid, Clock, and Track Receivers are permitted to be disabled
- If [Management Transport protocol is not supported] OR [the SB_MGMT_UP flag is cleared to 0] OR [a Sideband Packet is not being sent], SB Transmitters continue to be held Low
- SB Receivers continue to be enabled

The SB initialization procedure is performed at 800 MT/s with 800-MHz sideband clock.

If [Management Transport protocol is not supported] OR [SB_MGMT_UP=0]:

- While the LTSM is in the SBINIT state, a Physical Layer implementation is permitted to wait for an implementation specific amount of time to see whether SBINIT is progressing or completing, and subsequently handle any new or pending untransmitted sideband register access requests as Unsupported Requests, and if treating the request as an Unsupported Request, it must return a corresponding completion as well as end-to-end credit and RDI credit to the Adapter. Similarly, for register access completions, a Physical Layer implementation is permitted to wait for an implementation specific amount of time to see whether SBINIT is progressing or completing, and subsequently discard any new or pending untransmitted completions and because completions do not consume a credit, there is no credit return back to the Adapter for these. While the LTSM is in the SBINIT state, an implementation may handle any new or pending untransmitted Vendor-defined messages in an implementation dependent manner. This could include dropping the message and logging an error.

Advanced Package has redundant SB clock and SB data Lanes (**DATASBRD**, **CKSBRD**) in addition to **DATASB** and **CKSB**. SBINIT sequence for **Advanced Package** where interconnect repair may be needed is as follows:

- The UCIe Module must start and continue to send iterations of a 64-UI clock pattern (a clock pattern is defined as starting with 1 and toggling every UI of transmission, i.e., 1010...) and 32-UI low on both sideband data Transmitters (**TXDATASB** and **TXDATASBRD**). The UCIe Module must send strobes on both **TXCKSB** and **TXCKSBRD** during the 64-UI clock pattern transmission and be gated (held low) otherwise.
- UCIe Module Partner must sample each incoming data patterns on its sideband Receivers with both incoming sideband clocks (this forms four Receiver/clock combinations).
- A sideband data-clock Receiver combination detection is considered successful if 128 UI clock pattern is detected.

4. If a UCIe Module Partner detects the pattern successfully on at least one of its sideband data-clock Receiver combination, it must stop sending data and clock on its sideband Transmitters after four more iterations of 64-UI clock pattern and 32-UI low. This will allow for any time differences in both UCIe Module and UCIe Module Partner coming out of RESET state. The sideband Transmitter and Receiver on the UCIe Module must now be enabled to send and receive sideband messages
5. If pattern is not detected on its sideband Receiver, the UCIe Module must continue to alternate between sending the pattern on its sideband Transmitters for 1 ms, and holding low for 1 ms, for a total of 8 ms. The sideband Receiver of the UCIe Module must remain enabled during this time. Timeout occurs after 8 ms. If a timeout occurs, the UCIe Module enters TRAINERROR state.
6. If detection is successful on more than one sideband data/clock combination, the device can pick a combination based on a priority order. Pseudocode for sideband assignment:

```

CKSB sampling DATASB = Result[0] # 1: Detected; 0: Not detected
CKSBRD sampling DATASB = Result[1] # 1: Detected; 0: Not detected
CKSB sampling DATASBRD = Result[2] # 1: Detected; 0: Not detected
CKSBRD sampling DATASBRD = Result[3] # 1: Detected; 0: Not detected

IF (Result[3:0] == XXX1):
    Sideband = (DATASB/CKSB)
ELSE IF (Result[3:0] == XX10):
    Sideband = (DATASB/CKSBRD)
ELSE IF (Result[3:0] == X100):
    Sideband = (DATASBRD/CKSB)
ELSE IF (Result[3:0] ==1000):
    Sideband = (DATASBRD/CKSBRD)
Else:
    Sideband is not functional

```

7. If the sideband on the UCIe Module is enabled to send and receive sideband messages ([Step 4](#)), the UCIe Module must start and continue to send {SBINIT Out of Reset} sideband message on both **TXDATASB** and **TXDATASBRD** while sending both **TXCKSB** and **TXCKSBRD** until it detects the same message in its sideband Receivers or a timeout occurs at 8 ms.
8. If {SBINIT Out of Reset} sideband message detection is successful on its sideband Receivers, the UCIe Module stops sending the sideband message. Before sending any further sideband messages, both UCIe Module and UCIe Module Partner must apply Sideband Data/Clock assignment (called the functional sideband) based on the information included in the {SBINIT Out of Reset} sideband message.
9. Any further sideband messages must be sent and received on the functional sideband. Any sideband message exchange can now be performed.
10. The UCIe Module sends the {SBINIT done req} sideband message and waits for a response. If this message is received successfully, UCIe Module Partner responds with {SBINIT done resp} sideband message. When a UCIe Module has sent and received the {SBINIT done resp} sideband message, the UCIe Module must exit to MBINIT. The following additional rules apply when the transmitting/receiving chiplet supports Management Transport protocol. These additional rules are required to initiate mainband link training for the scenario in which the Management Transport path has already been established prior (i.e., SB_MGMT_UP flag is set to 1) and one of the mainband link training triggers (see [Section 4.5](#)) occurred with the Link Training State Machine in RESET state.

- If the Module partner that is receiving the {SBINIT done req} sideband message is in RESET state and is ready to proceed with mainband initialization, the module partner must transition to SBINIT state, respond with an {SBINIT done resp} sideband message, and then send a {SBINIT done req} sideband message.
- Module partner must ignore a received {SBINIT done req} sideband message if the module partner is EITHER [in RESET state but not yet ready to proceed with mainband initialization] OR [in a state machine state other than RESET/SBINIT].
- UCIe Module that is transmitting the {SBINIT done req} sideband message must transition to RESET state (by way of TRAINERROR state) if the UCIe Module did not receive a response within the regular 8-ms time window. In that scenario, the UCIe Module can choose to re-issue the {SBINIT done req} sideband message after transitioning to SBINIT state again from RESET state. The UCIe Module can repeat this process N number of times before waiting in RESET for a new training trigger. (The value of N is implementation-dependent.) The UCIe Module partner must collapse multiple outstanding {SBINIT done req} sideband messages and respond only with a single {SBINIT done resp} sideband message.

The next state is mainband initialization (MBINIT) if sideband message exchange is successful.

SBINIT sequence for Standard Package where interconnect Lane redundancy and repair are not supported is as follows:

1. The UCIe Module must start and continue to send iterations of 64 UI clock pattern (a clock pattern is defined as starting with 1b and toggling every UI of transmission, i.e., 1010...) and 32 UI low on its sideband Transmitter (**TXDATASB**). The UCIe Module must send strobe on its sideband clock (**TXCKSB**) during the 64-UI clock pattern duration and gated (held low) otherwise.
2. The UCIe Module Partner must sample incoming data pattern with incoming clock.
3. Sideband pattern detection is considered successful if 128 UI clock pattern is detected.
4. If the UCIe Module successfully detects the pattern, it stops sending data and clock on its sideband Transmitters after four more iterations of pattern in [Step 1](#). This will allow for any time differences in both UCIe Modules coming out of RESET. The UCIe Module sideband Transmitter and Receiver must now be enabled to send and receive sideband messages, respectively.
5. If pattern is not detected on its sideband Receiver, the UCIe Module continues to alternate between sending the pattern on its Transmitters for 1 ms, and holding low for 1 ms, for a total of 8 ms. The sideband Receiver must be enabled during this time. Timeout occurs after 8 ms. If a timeout occurs, the UCIe Module must exit to TRAINERROR. If a pattern is detected successfully at any time, as described in [Step 3](#), the UCIe Module enables sideband message transmission as described in [Step 4](#) and continues to [Step 6](#).
6. Once sideband detection is successful ([Step 5](#)), the UCIe Module must start and continue to send {SBINIT Out of Reset} sideband message on **TXDATASB** while sending **TXCKSB** until it detects the same message in its sideband Receivers or a timeout occurs.
7. If {SBINIT Out of Reset} sideband message detection is successful, the UCIe Module must stop sending the message. Any sideband message exchange can now be performed.
8. The UCIe Module must send the {SBINIT done req} sideband message. If this message is received successfully, the UCIe Module Partner responds with the {SBINIT done resp} sideband message. When the UCIe Module has sent and received the {SBINIT done resp} sideband message, the UCIe Module must exit to MBINIT. The following additional rules apply when the transmitting/receiving chiplet supports Management Transport protocol. These additional rules are required to initiate mainband link training for the scenario in which the Management Transport path has already been established prior (i.e., SB_MGMT_UP flag is set to 1) and one of the mainband link training triggers (see [Section 4.5](#)) occurred with the Link Training State Machine in RESET state.

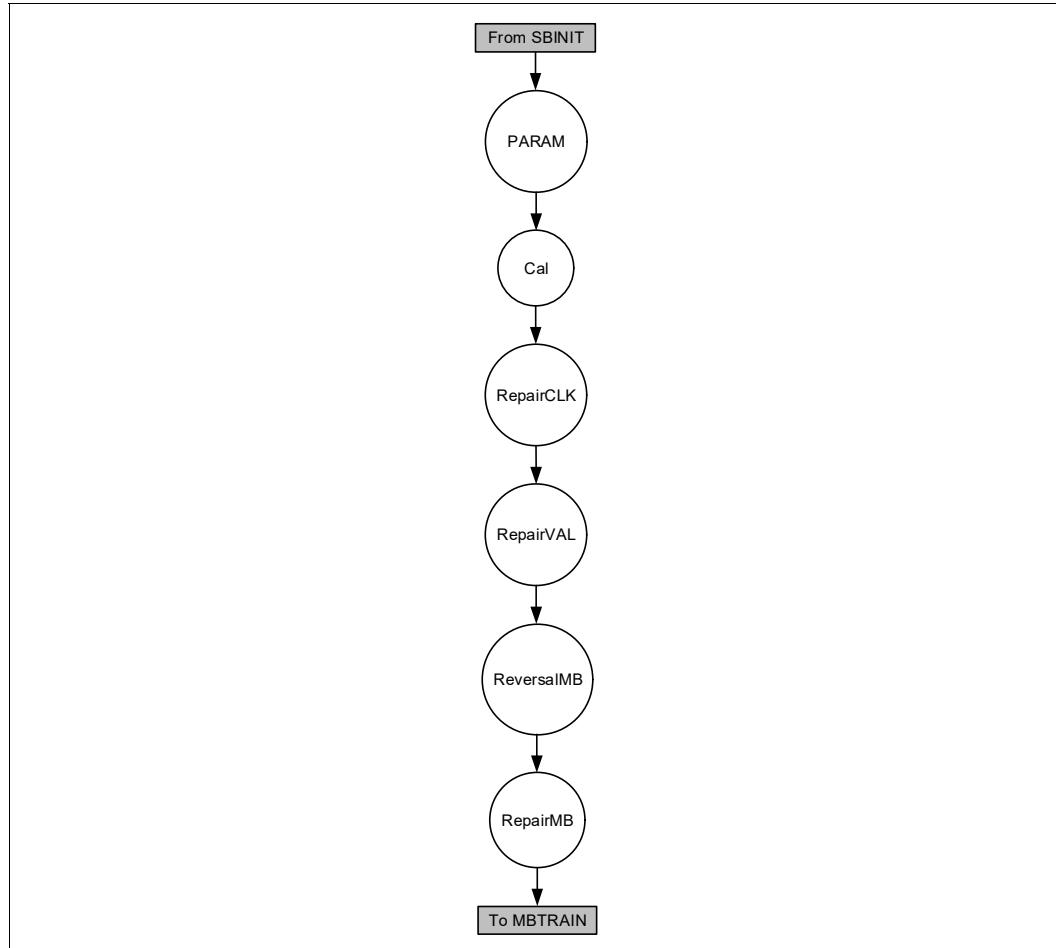
- If the Module partner that is receiving the {SBINIT done req} sideband message is in RESET state and is ready to proceed with mainband initialization, the module partner must transition to SBINIT state, respond with an {SBINIT done resp} sideband message, and then send a {SBINIT done req} sideband message.
- Module partner must ignore a received {SBINIT done req} sideband message if the module partner is EITHER [in RESET state but not yet ready to proceed with mainband initialization] OR [in a state machine state other than RESET/SBINIT].
- UCIe Module that is transmitting the {SBINIT done req} sideband message must transition to RESET state (by way of TRAINERROR state) if the UCIe Module did not receive a response within the regular 8-ms time window. In that scenario, the UCIe Module can choose to re-issue the {SBINIT done req} sideband message after transitioning to SBINIT state again from RESET state. The UCIe Module can repeat this process N number of times before waiting in RESET for a new training trigger. (The value of N is implementation-dependent.) The UCIe Module partner must collapse multiple outstanding {SBINIT done req} sideband messages and respond only with a single {SBINIT done resp} sideband message.

The next state is mainband initialization (MBINIT) if sideband message exchange is successful. For the remainder of initialization and operations, when not transmitting sideband packets, sideband Transmitters are held Low and sideband Receivers are enabled.

4.5.3.3 MBINIT

In this state, the mainband (MB) interface is initialized and repaired or width degraded (when applicable). The data rate on the mainband is set to the lowest supported data rate (4 GT/s).

For **Advanced Package** interconnect repair may be needed. Sub-states in MBINIT allows detection and repair of data, clock, track and valid Lanes. For **Standard Package**, where no Lane repair is needed, sub-states are used to check functionality at lowest data rate and width degrade if needed.

Figure 4-34. MBINIT: Mainband Initialization and Repair Flow

4.5.3.3.1 MBINIT.PARAM

This state is used to perform exchange of parameters that are required to set up the maximum negotiated speed and other PHY settings. Mainband Transmitters remain tri-stated and mainband Receivers are permitted to be disabled. The following parameters are exchanged over sideband with UCIe Module Partner:

- “Voltage swing”: The five bit value indicates the Transmitter voltage swing to the UCIe Module Partner. The UCIe Module Partner must use this value and its Receiver termination information to set the reference voltage (Vref) for its Receivers. The corresponding bits in the {MBINIT.PARAM configuration resp} sideband message are reserved.
- “Maximum Data Rate”: The four bit value indicates the Maximum supported Data rate to the UCIe Module Partner. This value must take into consideration all the required features at the data rate (BER, CRC/Retry, quadrature clock phase support etc.). The UCIe Module Partner must compare this value with its supported maximum data rate and must respond with the maximum common data rate encoding in the {MBINIT.PARAM configuration resp} sideband message. For example, a UCIe Module is 8 GT/s capable while the UCIe Module Partner advertises 16 GT/s, the UCIe Module must pick 8 GT/s and send it back in response.
- “Clock Mode”: The one bit value indicates the UCIe Module’s request to the UCIe Module Partner for a strobe or continuous clock. The UCIe Module Partner must use this information to set up the

clock mode on its clock Transmitter. The {MBINIT.PARAM configuration resp} sideband message must reflect the same value. Continuous clock mode requires the clock to be free running and is enforced after receiving the {MBTRAIN.RXCLKCAL start req} sideband message from the UCIe Module Partner. The clock remains free running through the remainder of MBTRAIN (unless MBTRAIN.LINKSPEED is exited due to errors) and in ACTIVE.

- “Clock Phase”: The one bit value indicates the UCIe Module’s request to UCIe Module Partner for the Clock Phase support on UCIe Module’s forwarded clock. This should only be set 1b if the maximum data rate advertised is permitted to do so (see [Table 5-8](#)). The corresponding bit in the {MBINIT.PARAM configuration resp} sideband message must be set to 1b if this was requested and the operational data rate allows it.
- “Module ID”: The UCIe Module sends its “Module ID”. This can be used by the UCIe Module Partner if in a multi-module configuration for Byte mapping, Module enable/disable information etc. The corresponding bits in the {MBINIT.PARAM configuration resp} sideband message are reserved.
- “UCIe-A x32”: This bit is set to 1b when APMW bit in DVSEC UCIe Link Capability register (see [Section 9.5.1.4](#)) is set to 1b (OR) ‘Force x32 width mode in x64 Module’ in the PHY Control register (see [Section 9.5.3.23](#)) is set; otherwise, the bit is set to 0b. If a x64 Advanced Package module supports width reduction to interoperate with a x32 Advanced Package Module, it uses this information from its link partner to condition the results during MBINIT.REVERSALMB. The corresponding bits in the {MBINIT.PARAM configuration resp} sideband message are reserved.
- “UCIe-S x8”: This bit is set to 1 in message {MBINIT.PARAM configuration req} when bit 20, SPMW, in the DVSEC UCIe Link Capability register (see [Section 9.5.1.4](#)) is set to 1 (OR) ‘Force x8 Width’ bit is set to 1 in the PHY Control register (see [Section 9.5.3.23](#)). Otherwise, this bit is set to 0. See [Section 4.5.3.3.6](#) for how this bit is used. The corresponding bit in the {MBINIT.PARAM configuration resp} sideband message is reserved.
- “Sideband Feature Extensions”: This bit is set to 1 if the transmitter supports sideband feature extensions (see [Section 4.5.3.3.1.1](#)).

Following is the sequence for parameter exchange:

1. The UCIe Module sends the {MBINIT.PARAM configuration req} sideband message to exchange parameters with the UCIe Module Partner.
2. After the {MBINIT.PARAM configuration req} sideband message is received, the UCIe Module Partner resolves and responds with the {MBINIT.PARAM configuration resp} sideband message.
3. After the UCIe Module has sent and received the {MBINIT.PARAM configuration resp} sideband message, the UCIe Module must exit to MBINIT.CAL.

It is strongly recommended that if interoperable parameters are not negotiated, then hardware maps this scenario to an Internal Error in the Error Log 1 register and transition the LTS to TRAINERROR, RDI to LINKERROR, and assert **p1_trainerror** on RDI. For a multi-module Link, all the parameters except “Module ID” must be the same for all the modules, and if this is not the case, it is strongly recommended that hardware maps this scenario to the same error escalation path.

When management transport is supported, the additional conditions required for the Link training state machine to exit MBINIT.PARAM state to MBINIT.CAL are:

- Management Transport was not negotiated.
- (OR) Management Transport was negotiated and it was either initialized successfully or an error was detected during initialization.
- (OR) SB_MGMT_UP flag is already set on entry into MBINIT state

AND

There is no MBINIT Stall condition (see [Section 4.5.3.3.1.2](#))

AND

Port is ready to train mainband.

4.5.3.3.1.1 Management Transport Path Negotiation

Management Transport protocol over the sideband is optional and chiplets use the mechanism described in this section to negotiate support for it. A sample negotiation flow is shown in [Figure 4-35](#) for a single module design. Sideband Feature Extensions Supported (SFES) is bit 14 in the {MBINIT.PARAM configuration req} sideband message (see [Table 7-11](#)). Note that MBINIT.PARAM state handshake relating to management path negotiation described in this section, is performed on all transitions through the MBINIT state. If SB_MGMT_UP flag is set (see [Section 8.2.3.1.2](#) for when this happens) at entry into MBINIT state, management transport traffic continues without interruption in the MBINIT.PARAM state.

Figure 4-35. Example Sideband Management Transport Protocol Negotiation – Single-module Scenario

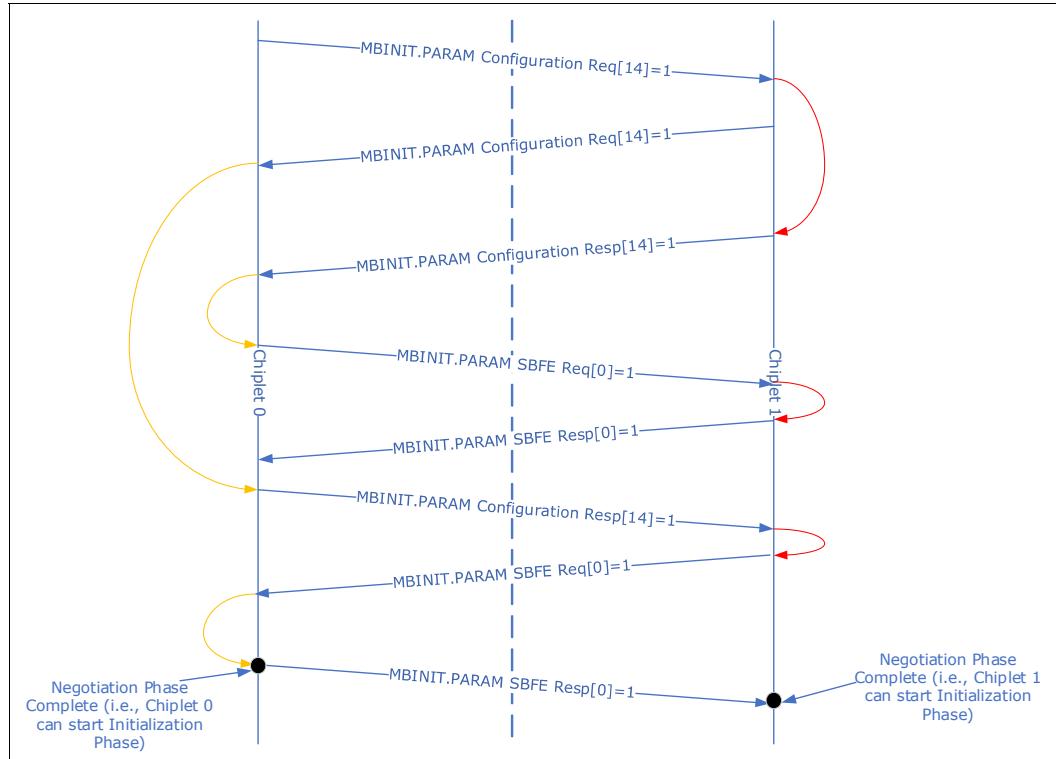
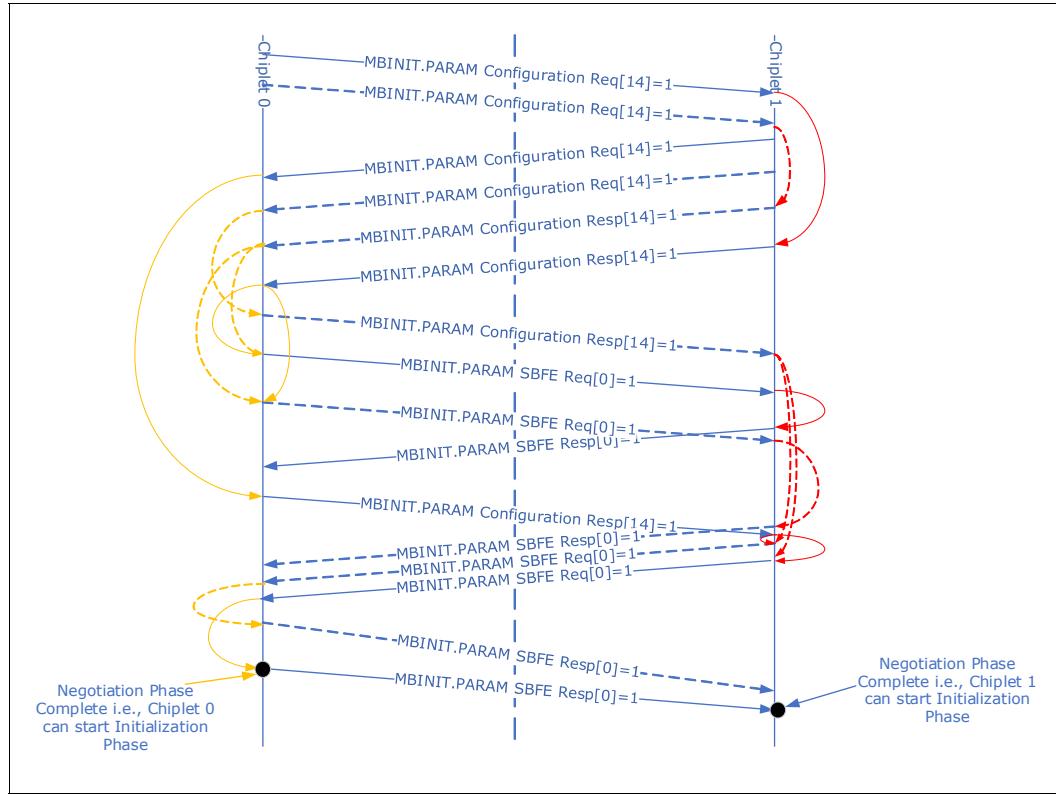


Figure 4-36. Example Sideband Management Transport Protocol Negotiation – Two-module Scenario^a



a. Solid lines are for Module 0. Dashed lines are for Module 1.

- Unless otherwise specified, a chiplet can optionally check for violation of any Negotiation Phase rules (discussed in the subsequent bullets), and when a violation is detected, the chiplet initiates a TRAINERROR handshake (see [Section 4.5.3.8](#)) to return the LTSI to RESET state.
- Sideband Feature Extensions Supported (SFES) bit in the {MBINIT.PARAM configuration req} sideband message (see [Table 7-11](#), it is bit [14] in the message) is defined to indicate support for extended sideband features (of which Management Transport is one), during MBINIT.PARAM state of link training.
 - 0 => Sideband Feature Extensions are not supported, 1 => Sideband Feature Extensions are supported.
 - All modules in a multi-module design must have the same value for this bit in the Req message.
 - After the SB_MGMT_UP flag is set, the value of this bit must remain the same on all subsequent transitions through the MBINIT.PARAM state, until that flag is cleared.
- If the Remote link partner supports sideband feature extensions and it received the SFES bit set to 1 in the {MBINIT.PARAM configuration req} sideband message, the Remote link partner will set SFES bit in the {MBINIT.PARAM configuration resp} sideband message that it sends out; otherwise, the bit is cleared to 0 in the resp message.
 - All modules in a multi-module design must have the same value for this bit in the resp message.

- If the SFES bit in the {MBINIT.PARAM configuration resp} sideband message is received as set to 1 across all modules, then the chiplet negotiates the next level of details of extended sideband features supported with remote link partner. If the SFES bit in the {MBINIT.PARAM configuration resp} sideband message is received as cleared to 0 in any module, then MBINIT.PARAM state is exited to MBINIT.CAL.
 - {MBINIT.PARAM SBFE req} sideband message (see [Table 7-11](#)) is sent to the remote link partner on all modules which then sends back an {MBINIT.PARAM SBFE resp} sideband message on all modules. This handshake happens independently in each direction.
 - SBFE Req[0]/Resp[0] (also referred to as the MTP bit) indicates support for transmission/reception of Management Port Messages. Remote link partner must set the MTP bit in the {MBINIT.PARAM SBFE resp} sideband message if it was set in the {MBINIT.PARAM SBFE req} sideband message, and it supports receiving Management Encapsulation messages.
 - After the SB_MGMT_UP flag is set to 1, the value of this bit must remain the same on all subsequent transitions through the MBINIT.PARAM state, until that flag is cleared to 0.
- When negotiating SFES in {MBINIT.PARAM configuration req/resp}, if a chiplet advertised SFES support in the Req message, the chiplet must also advertise that support in the Resp message provided the associated Req message had that capability advertised. If the chiplet did not advertise SFES support in the Req message, then the chiplet must not advertise that support in the Resp message.
- For a multi-module UCIe Link, the negotiation is performed independently per module.
 - A Physical Layer implementation may advertise MTP bit in the SBFE Req message only on a subset of the modules.

Note: A message sent on a given Module ID could be received on a different Module ID on the partner sideband Receiver. Hence all sideband links in a multi-module design must be capable of receiving MPMs even if they are limited to only supporting transmit of these messages on a subset of sideband links. See [Figure 4-37](#) and [Figure 4-38](#) for examples of multi-module transmit/receive scenarios that illustrate this point.

- After the {MBINIT.PARAM SBFE resp} sideband message has been transmitted to the remote link partner and {MBINIT.PARAM SBFE resp} sideband message has been received from the remote link partner are complete (successfully or unsuccessfully) during MBINIT.PARAM across all modules, the PHY informs the Management Port Gateway of the following:
 - Negotiated link count with management transport support on the transmit side, using the `pm_param_local_count[N-1:0]` signals (see [Section 10.1](#) for more details) at the end of the negotiation phase. This is the value RxQ-Local. A link is considered to have negotiated management transport support on the transmit side if the link transmitted the {MBINIT.PARAM SBFE req} sideband message with the MTP bit set to 1 and received the corresponding {MBINIT.PARAM SBFE resp} sideband message with its MTP bit also set to 1.
 - Negotiated link count with management transport support on the receive side, using the `pm_param_remote_count[N-1:0]` signals (see [Section 10.1](#) for more details). This is the value RxQ-Remote. A link is considered to have negotiated management transport support on the receive side if the link received the {MBINIT.PARAM SBFE req} sideband message with the MTP bit set to 1 and transmitted the corresponding {MBINIT.PARAM SBFE resp} sideband message with its MTP bit also set to 1.
- A module must be able to receive initialization phase-related messages (see [Section 8.2.3.1.2](#)) once it has transmitted {MBINIT.PARAM SBFE resp}.
- Negotiation phase ends when {MBINIT.PARAM SBFE resp} has been sent and received across all modules.

- While in SBINIT state, if the SB_MGMT_UP flag transitioned from 1 to 0, the chiplet must move the LTSM to TRAINERROR state -> RESET state.
- While in MBINIT state, if the SB_MGMT_UP flag transitioned from 1 to 0, the chiplet must perform a TRAINERROR handshake and move the LTSM to TRAINERROR state -> RESET state.

Figure 4-37. Example Sideband MPM Logical Flow with Two Modules and No Module Reversal

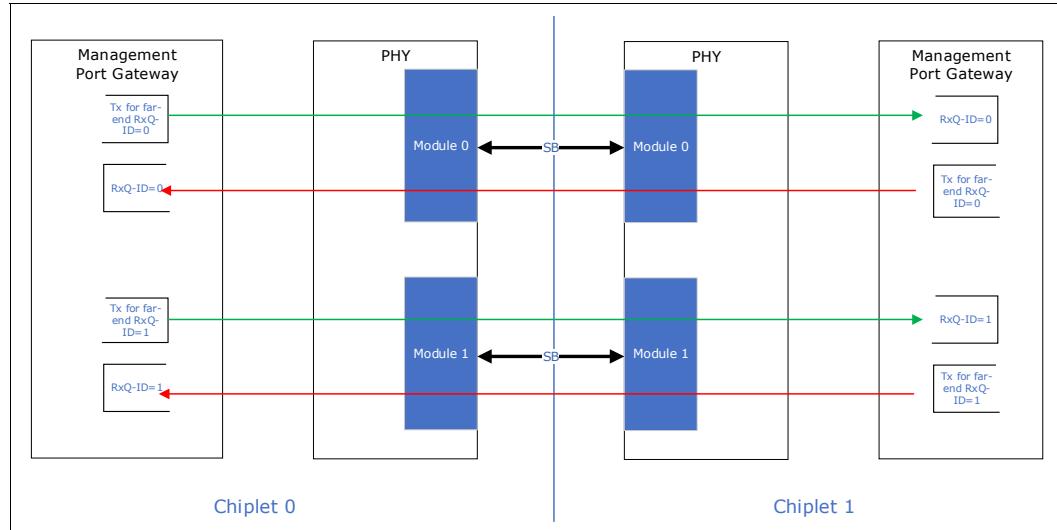
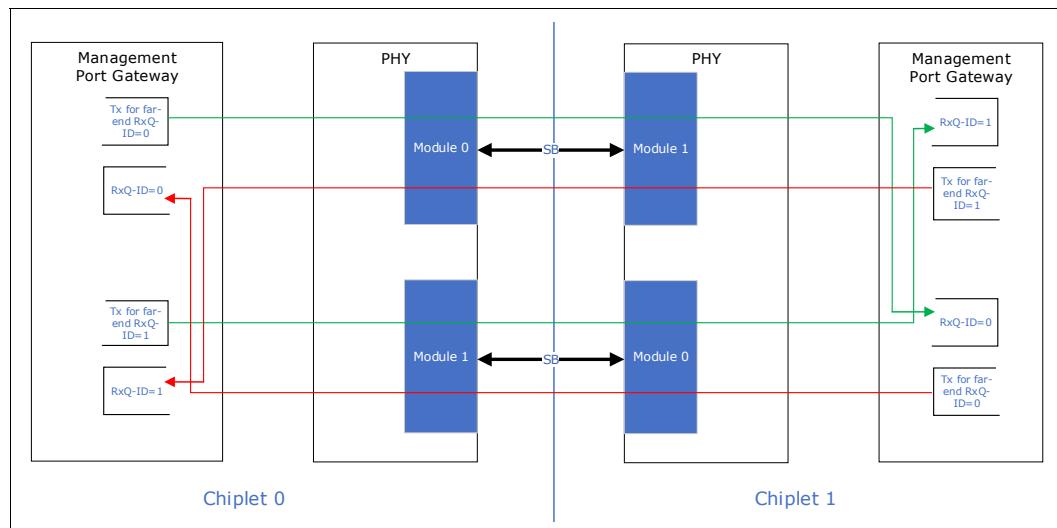


Figure 4-38. Example Sideband MPM Logical Flow with Two Modules and Module Reversal



4.5.3.3.1.2 MBINIT Stall Capability

To support firmware download and other functionality that might have to be configured before the mainband link can start training, a capability is provided to “pause” the MBINIT state machine after the PARAM sub-state has completed.

This is an optional capability that is enabled only when both chiplets have indicated support for Sideband Feature Extensions as described in Section 4.5.3.3.1.1.

To “pause” link training after MBINIT.PARAM, either side can send a “Stall” encoding of FFFFh in the MsgInfo field of the {MBINIT.PARAM SBFE resp} sideband message. For example, if a chiplet needs to download firmware by way of the partner port before the chiplet can bring up the mainband, the partner port can respond with “stall” encoding as stated above. Stall encoding instructs the other side to pause and not move beyond the MBINIT.PARAM state. The payload in the {MBINIT.PARAM SBFE resp} sideband message with stall encoding is still valid and must be accurate to the responder’s capabilities. An {MBINIT.PARAM SBFE resp} sideband message with “Stall” encoded must be sent once every 4 ms until the sender determines that it no longer needs to stall, at which time the sender either sends the {MBINIT.PARAM SBFE resp} message without the stall encoding (in which case the state machine advances to MBINIT.CAL state if other conditions allow (see [Section 8.2.3.1.2](#))), OR the sender does not send an {MBINIT.PARAM SBFE resp} sideband message, the link times out, and the link transitions from TRAINERROR state to RESET state and starts over again.

It is legal for one side to indicate a stall and the other side to not indicate a stall. In that case, both sides are stalled. It is also valid for either side to explicitly request an entry to TRAINERROR state while in MBINIT.PARAM state. This can occur if either side is not yet ready to train the mainband.

See [Section 4.5.3.3.1.3](#) for details of what happens at the end of MBINIT.PARAM when either end is a sideband-only port.

Support for receiving an {MBINIT.PARAM SBFE resp} sideband message with “stall” encoding is required in all ports that advertise the SFES bit set to 1 in the {MBINIT.PARAM configuration req} sideband message. Support for transmitting the {MBINIT.PARAM SBFE resp} sideband message with “stall” encoding is implementation-dependent. For example, if a design needs to support the firmware download feature, the design can support this capability if the design cannot complete firmware download within 8 ms.

[Figure 4-39](#) shows a scenario in which the link training state machine initially moves through RESET -> SBINIT -> MBINIT.PARAM with “stall” -> TRAINERROR -> RESET. During this phase, a chiplet “stalls” in MBINIT.PARAM for additional Init time, such as for downloading chiplet firmware. When the chiplet Init is complete, the chiplet either initiates entry to TRAINERROR state with a TRAINERROR handshake message, OR the chiplet can stop sending the {MBINIT.PARAM SBFE resp} sideband message with “stall” encoding, which would eventually trigger entry to TRAINERROR state initiated by the partner chiplet. In the second phase, the state machine moves through to SBINIT -> MBINIT.PARAM without “stall” -> MBINIT.CAL, thus training the mainband. This flow is useful for scenarios in which a chiplet potentially needs to change the advertised parameters for Link training after chiplet Init. Note that the transition to TRAINERROR in this case does not escalate to RDI transitioning to LinkError (or `pl_trainerror` assertion on RDI).

Figure 4-39. MBINIT “Stall” Example 1

Sideband Management path setup occurs first, followed by mainband training but with an entry to RESET in between.

“Stall” in MBINIT.PARAM is used to gain additional time for chiplet initialization during the first entry to MBINIT.PARAM.

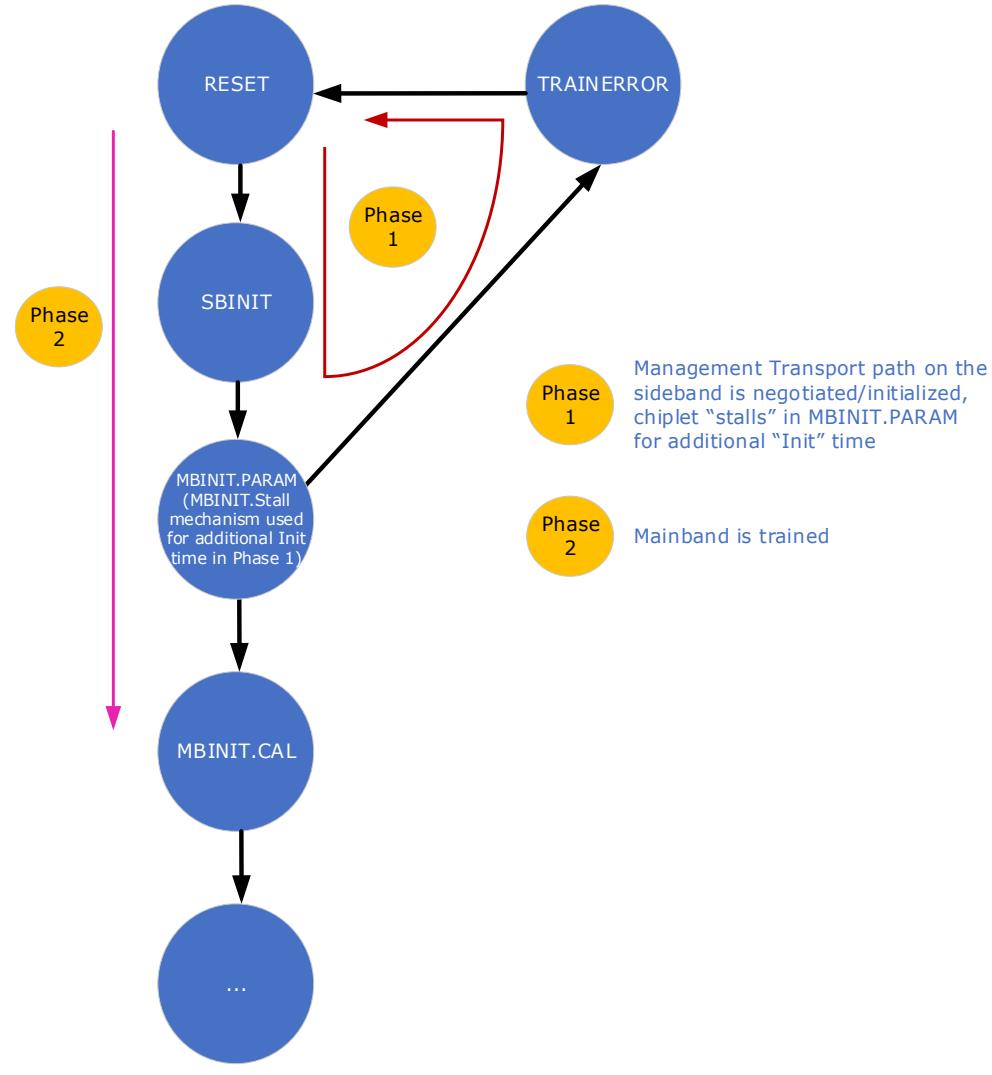
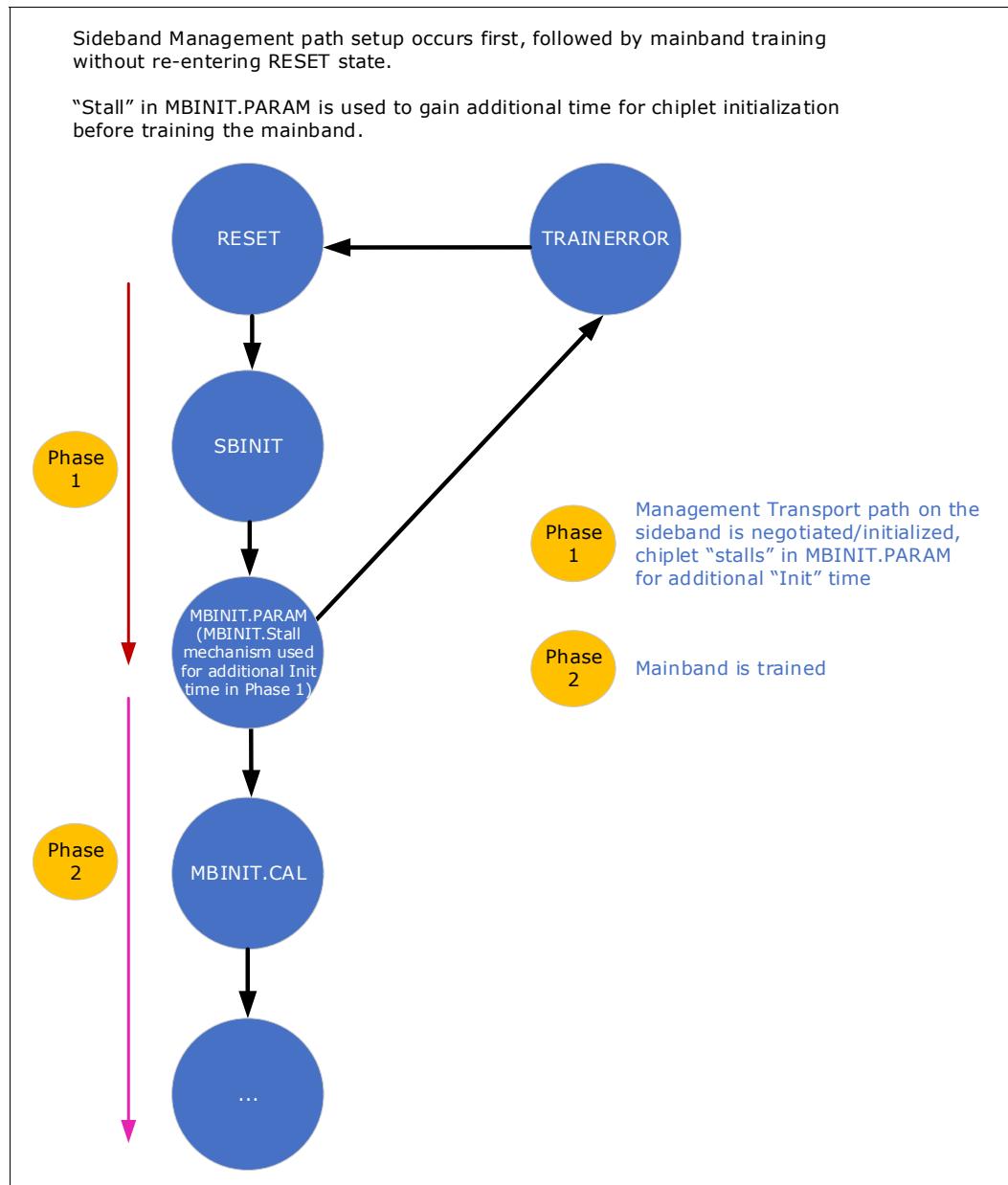


Figure 4-40 shows a scenario in which the link training state machine initially moves through RESET -> SBINIT -> MBINIT.PARAM with “stall”. The chiplet “stalls” in MBINIT.PARAM for additional Init time. After the chiplet Init is complete, the chiplet sends an {MBINIT.PARAM SBFE resp} sideband message without a “stall” encoding that triggers state machine entry to MBINIT.CAL. Mainband training resumes from that point.

Figure 4-40. MBINIT “Stall” Example 2



4.5.3.3.1.3 UCIe-S Sideband Only (SO) Port

A UCIe-S Sideband-only port will advertise the SFES bit in an {MBINIT.PARAM configuration req} sideband message. If that negotiation is successful, the port advertises bit 2 (SO bit) in an {MBINIT.PARAM SBFE req} sideband message to indicate that the port is a "Sideband-only port". If the port receives an {MBINIT.PARAM SBFE resp} sideband message with the SO bit set to 1, training pauses on both sides at the exit of the MBINIT.PARAM phase until the next Management Reset or a transition to the TRAINERROR state is triggered (see [Section 4.5.3.8](#)). State residency timeout is disabled in MBINIT.PARAM. If the remote link partner did not set the SO bit in the {MBINIT.PARAM SBFE resp} sideband message, the link goes to the TRAINERROR state. This is an SiP integration error and is fatal. All links that support management transport over sideband (i.e., those links that set bit 0 as 1 in the {MBINIT.PARAM SBFE req} sideband message that they transmit) must set the SO bit to 1 in the {MBINIT.PARAM SBFE resp} sideband message that they transmit, provided that the corresponding bit was set to 1 in the {MBINIT.PARAM SBFE req} sideband message that they received.

In multi-sideband-only Link configuration (this refers to a multi-module configuration with sideband-only modules; see [Section 8.2.1](#)), the chiplet must advertise the same value for the SO bit in the {MBINIT.PARAM SBFE req} or {MBINIT.PARAM SBFE resp} sideband message across all links. Receivers can optionally check for violation of this rule and then trigger a transition to the TRAINERROR state if the transmitter violates this rule.

4.5.3.2 MBINIT.CAL

This state is used to perform any calibration needed (e.g., Tx Duty Cycle correction, Receiver offset and Vref calibration). This state is common for both **Advanced Package** and **Standard Package**.

1. The UCIe Module maintains tri-state on all its mainband Transmitters, and mainband Receivers are permitted to be disabled in this state. The UCIe Module is permitted to perform implementation specific steps for Transmitter and Receiver calibration.
2. The UCIe Module must send the {MBINIT.CAL Done req} sideband message, and then wait for a response. If this message is received successfully, the UCIe Module Partner responds with the {MBINIT.CAL Done resp} sideband message. Once the UCIe Module has sent and received {MBINIT.CAL Done resp}, it must exit to MBINIT.REPAIRCLK.

4.5.3.3 MBINIT.REPAIRCLK

This sub-state is used to detect and apply repair (if needed) to clock and track Lanes for **Advanced Package** and for functional check of clock and track Lanes for **Standard Package**.

Following is the sequence for **Advanced Package**:

Clock repair mapping is described in [Section 4.3](#). Each clock, track and their redundant physical Lanes (`TCKP_P/RCKP_P`, `TCKN_P/RCKN_P`, `TTRK_P/RTRK_P`, and `TRDCK_P/RRDCK_P`) are independently checked to detect possible electrical opens or electrical shorts between the two clock pins. Single-ended clock Receivers or independent detection mechanism is required to ensure clock repair. The UCIe Module must enable Transmitters and Receivers on Clock, Track and their redundant Lanes. All other Transmitters are maintained in tri-state and Receivers are permitted to be disabled.

1. The UCIe Module sends the {MBINIT.REPAIRCLK init req} sideband message and waits for a response. The UCIe Module Partner when ready to receive pattern on `RCKP_L`, `RCKN_L`, `RTRK_L`, and `RRDCK_L` responds with {MBINIT.REPAIRCLK init resp}.
2. The UCIe Module must now send 128 iterations of clock repair pattern (16 clock cycles followed by 8 cycles of low) on its `TCKP_L` only (`TCKN_L`, `TTRK_L`, and `TRDCK_L` are tri-stated). Clock repair pattern must not be scrambled.

3. The UCIe Module Partner detects this pattern on **RCKP_L**, **RCKN_L**, **RTRK_L**, and **RRDCK_L**. Detection is considered successful if at least 16 consecutive iterations of clock repair pattern are detected. The UCIe Module Partner logs the detection result for **RCKP_L**, **RCKN_L**, **RTRK_L**, and **RRDCK_L**.
4. The UCIe Module after completing pattern transmission sends {MBINIT.REPAIRCLK result req} sideband message to get the logged result and waits for a response.
5. The UCIe Module Partner responds with {MBINIT.REPAIRCLK result resp} sideband message with log result of **RCKP_L**, **RCKN_L**, **RTRK_L**, and **RRDCK_L**. If detection is successful on **RCKP_L** only and not on any of **RCKN_L**, **RTRK_L**, and/or **RRDCK_L**, no repair is needed. Else if detection is unsuccessful on any of **RCKP_L**, **RCKN_L**, **RTRK_L**, and **RRDCK_L**, repair is needed on the physical Lane **TCKP_P/RCKP_P**. Else an electrical short is implied.
6. After receiving the {MBINIT.REPAIRCLK result resp} sideband message, the UCIe Module sends the sideband message {MBINIT.REPAIRCLK init req} and waits for a response. The UCIe Module Partner when ready to receive pattern on **RCKP_L**, **RCKN_L**, **RTRK_L**, **RRDCK_L** responds with {MBINIT.REPAIRCLK init resp}.
7. After receiving the {MBINIT.REPAIRCLK init resp} sideband message, the UCIe Module must send 128 iterations of clock repair pattern (16 clock cycles followed by 8 cycles of low) on its **TCKN_L** only. (**TCKP_L**, **TTRK_L**, and **TRDCK_L** are tri-stated)
8. The UCIe Module Partner detects this pattern on all **RCKP_L**, **RCKN_L**, **RTRK_L**, and **RRDCK_L**. Detection is considered successful if at least 16 consecutive cycles of clock repair pattern are detected. The UCIe Module Partner logs the detection result for **RCKP_L**, **RCKN_L**, **RTRK_L**, and **RRDCK_L**.
9. The UCIe Module after completing the pattern transmission, sends {MBINIT.REPAIRCLK result req} sideband message to get the logged result.
10. The UCIe Module Partner on receiving it responds with {MBINIT.REPAIRCLK result resp} sideband message with logged result of **RCKP_L**, **RCKN_L**, **RTRK_L**, and **RRDCK_L**. If detection is successful on **RCKN_L** and not on **RCKP_L**, **RTRK_L**, **RRDCK_L**, no repair is needed. Else if detection is unsuccessful on any of **RCKP_L**, **RCKN_L**, **RTRK_L**, and **RRDCK_L**, repair is needed on the physical Lane **TCKN_P/RCKN_P**. Else an electrical short is implied.
11. After receiving the {MBINIT.REPAIRCLK result resp} sideband message, the UCIe Module sends the sideband message {MBINIT.REPAIRCLK init req}. The UCIe Module Partner when ready to receive pattern on **RCKP_L**, **RCKN_L**, **RTRK_L**, **RRDCK_L** responds with {MBINIT.REPAIRCLK init resp}.
12. After receiving the {MBINIT.REPAIRCLK init resp} sideband message, the UCIe Module sends 128 iterations of clock repair pattern (16 clock cycles followed by 8 cycles of low) on **TRDCK_L** only. (**TCKP_L**, **TTRK_L**, and **TCKN_L** tri-stated)
13. The UCIe Module Partner detects this pattern on all **RCKP_L**, **RCKN_L**, **RTRK_L**, and **RRDCK_L**. Detection is considered successful if at least 16 consecutive cycles of clock repair pattern are detected. The UCIe Module Partner logs the detection result for **RCKP_L**, **RCKN_L**, **RTRK_L**, and **RRDCK_L**.
14. The UCIe Module device after completing the pattern transmission sends {MBINIT.REPAIRCLK result req} sideband message to get the logged result.

15. The UCIe Module Partner responds with {MBINIT.REPAIRCLK result resp} sideband message with logged result of **RCKP_L**, **RCKN_L**, **RTRK_L**, and **RRDCK_L**. If detection is successful only on **RRDCK_L** and not on **RCKP_L**, **RTRK_L**, **RCKN_L**, **TRDCK_P/RRDCK_P** is available as a repair resource. Else if detection is unsuccessful on any of **RCKP_L**, **RCKN_L**, **RTRK_L**, and **RRDCK_L**, the physical Lane **TRDCK_P/RRDCK_P** is not available as a repair resource. Else an electrical short is implied.
16. After receiving the {MBINIT.REPAIRCLK result resp} sideband message, the UCIe Module sends the sideband message {MBINIT.REPAIRCLK init req} and waits for a response. The UCIe Module Partner when ready to receive pattern on **RCKP_L**, **RCKN_L**, **RTRK_L**, **RRDCK_L** responds with {MBINIT.REPAIRCLK init resp}.
17. After receiving the {MBINIT.REPAIRCLK init resp} sideband message, the UCIe Module sends 128 iterations of clock repair pattern (16 clock cycles followed by 8 cycles of low) on **TTRK_L** only. (**TCKP_L**, **TCKN_L**, and **TRDCK_L** are tri-stated).
18. The UCIe Module Partner detects this pattern on all **RCKP_L**, **RCKN_L**, **RTRK_L**, and **RRDCK_L**. Detection is considered successful if at least 16 consecutive cycles of clock repair pattern are detected. The UCIe Module Partner logs the detection result for **RCKP_L**, **RCKN_L**, **RTRK_L**, and **RRDCK_L**.
19. The UCIe Module after completing pattern transmission sends {MBINIT.REPAIRCLK result req} sideband message to get the logged result and waits for a response.
20. The UCIe Module Partner stops comparison and responds with {MBINIT.REPAIRCLK result resp} sideband message with logged result of **RCKP_L**, **RCKN_L**, **RTRK_L**, and **RRDCK_L**. If detection is successful only on **RTRK_L** and not on **RCKP_L**, **RCKN_L**, **RRDCK_L**, no repair is needed. Else if detection is unsuccessful on any of **RCKP_L**, **RCKN_L**, **RTRK_L**, and **RRDCK_L**, repair is needed on the physical Lane **TTRK_P/RTRK_P**. Else an electrical short is implied.
21. Clock or Track is unrepairable if any of the following are true:
 - If repair is needed on any two of **RCKP_L**, **RCKN_L**, and **RTRK_L**
 - Electrical short is detected
 - **RRDCK_L** is unavailable for repair when repair is needed
 If the clock or track is unrepairable, the UCIe Module and UCIe Module Partner must exit to TRAINERROR after performing TRAINERROR handshake (see [Section 4.5.3.8](#)).
- If repair is required on only one of the clock or track lanes and a repair resource is available, then the UCIe Module applies repair on its clock/track Transmitter and sends the {MBINIT.REPAIRCLK apply repair req} sideband message with repair information. If a repair is needed for one of the clock or track pins (**CKP**, **CKN**, or **TRK**) and a repair resource is available, repair is applied as described in [Section 4.3](#). The UCIe Module Partner applies repair and sends {MBINIT.REPAIRCLK apply repair resp} sideband message.
22. If a repair is applied, UCIe Module must check the repair success by applying clock repair pattern and checking on the Receiver.
 - a. The UCIe Module sends sideband message {MBINIT.REPAIRCLK check repair init req} to initiate check repair and waits for a response. The UCIe Module Partner responds with sideband message {MBINIT.REPAIRCLK check repair init resp} and is ready to receive and check clock repair pattern.
 - b. After receiving the {MBINIT.REPAIRCLK check repair init resp} sideband message, the UCIe Module sends 128 iterations of clock repair pattern (16 clock cycles followed by 8 cycles of low) on **TCKP_L**. The UCIe Module Partner detects this pattern **RCKN_L**, **RCKP_L**, **RTRK_L**. Detection is considered successful if at least 16 consecutive cycles of clock repair pattern are detected. The UCIe Module requests for check result request using the sideband message

{MBINIT.REPAIRCLK check results req} and the UCIe Module Partner responds with the sideband message {MBINIT.REPAIRCLK check results resp}. Repair is considered successful if pattern is detected only on **RCKP_L**. If repair is unsuccessful, the UCIe Module and UCIe Module Partner must exit to TRAINERROR after performing TRAINERROR handshake (see [Section 4.5.3.8](#)).

- c. Step a and Step b are repeated for **TCKN_L** and **TTRK_L**.
- 23. If repair is successful or repair is not required, the UCIe Module sends {MBINIT.REPAIRCLK done req} sideband message and the UCIe Module Partner responds with {MBINIT.REPAIRCLK done resp} sideband message. When the UCIe Module has sent and received {MBINIT.REPAIRCLK done resp}, it must exit to REPAIRVAL.

For **Standard Package**, clock and track Lanes are checked for functional operation at the lowest data rate. The sequence is as follows:

1. The UCIe Module sends the sideband message {MBINIT.REPAIRCLK init req} and waits for a response. When ready to receive pattern on **RCKP_L**, **RCKN_L**, and **RTRK_L**, the UCIe Module Partner responds with {MBINIT.REPAIRCLK init resp}. On receiving the sideband message {MBINIT.REPAIRCLK init resp}, the UCIe Module sends 128 iterations of clock repair pattern (16 clock cycles followed by 8 cycles of low) on **TCKP_L**, **TCKN_L**, and **TTRK_L**. Clock repair pattern must not be scrambled.
2. The UCIe Module Partner detects this pattern on **RCKP_L**, **RCKN_L**, and **RTRK_L**. Detection is considered successful if at least 16 consecutive cycles of clock repair pattern are detected. The UCIe Module Partner logs the detection result for **RCKP_L**, **RCKN_L**, and **RTRK_L**.
3. After completing pattern transmission, the UCIe Module sends {MBINIT.REPAIRCLK result req} sideband message to get the logged result.
4. The UCIe Module Partner stops comparison and responds with {MBINIT.REPAIRCLK result resp} sideband message with logged result of **RCKP_L**, **RCKN_L**, and **RTRK_L**.
5. If detection is unsuccessful on any one of **RCKP_L**, **RCKN_L**, and **RTRK_L**, the UCIe Module and UCIe Module Partner must exit to TRAINERROR after performing TRAINERROR handshake.
6. If detection is successful, the UCIe Module sends {MBINIT.REPAIRCLK done req} sideband message and the UCIe Module Partner responds with {MBINIT.REPAIRCLK done resp} sideband message. When the UCIe Module has sent and received the sideband message {MBINIT.REPAIRCLK done resp}, it must exit to MBINIT.REPAIRVAL.

4.5.3.3.4 MBINIT.REPAIRVAL

The UCIe Module sets the clock phase at the center of the data UI. The UCIe Module Partner must sample the received Valid with the received forwarded Clock. All Data Lanes must be held low during this state. Track and Data Receivers are permitted to be disabled. When not performing the actions relevant to this state:

- Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking)
- Clock Receivers are enabled
- When not transmitting the VALTRAIN pattern, the transmitters for **TVLD_L** and **TRDVLD_L** are disabled (tri-stated)
- The receivers for **RVLD_L** and **RRDVLD_L** are enabled

This state can be used to detect and apply repair (if needed) to Valid Lane for **Advanced Package** and for functional check of Valid for **Standard Package**.

Following is the sequence for **Advanced Package**:

1. The UCIe Module sends the sideband message {MBINIT.REPAIRVAL init req} and waits for a response. When ready to receive pattern on **RVLD_L** and **RRDVLD_L**, the UCIe Module Partner clears any previous comparison results and responds with {MBINIT.REPAIRVAL init resp}.
2. The UCIe Module on receiving {MBINIT.REPAIRVAL init resp} must send 128 iterations of VALTRAIN pattern (four 1's followed by four 0's) on **TVLD_L** along with the forwarded clock. VALTRAIN pattern must not be scrambled.
3. The UCIe Module Partner detects pattern on **RVLD_L** and **RRDVLD_L**. Detection is considered successful if at least 16 consecutive iterations of VALTRAIN pattern are detected. The Receiver logs the detection result for **RVLD_L** and **RRDVLD_L**.
4. After completing the pattern transmission, the UCIe Module must send {MBINIT.REPAIRVAL result req} sideband message and wait to get the logged result.
5. The UCIe Module Partner must respond with {MBINIT.REPAIRVAL result resp} sideband message with result in the previous step. If detection is successful on **RVLD_L** only and not on **RRDVLD_L**, no repair is needed. If detection is successful on both **RVLD_L** and **RRDVLD_L** or only on **RRDVLD_L**, the interconnect cannot be repaired. If detection is unsuccessful on both **RVLD_L** and **RRDVLD_L**, Valid Lane (**TVLD_P/RVLD_P**) needs repair.
6. After receiving {MBINIT.REPAIRVAL result resp}, [Step 1](#) must be repeated.
7. The UCIe Module sends 128 iterations of VALTRAIN repair pattern (four 1's followed by four 0's) on **TRDVLD_L** along with the forwarded clock. VALTRAIN pattern must not be scrambled.
8. The UCIe Module Partner detects pattern on **RVLD_L** and **RRDVLD_L**. Detection is considered successful if at least 16 consecutive iterations of VALTRAIN pattern are detected. The Receiver logs the detection result for **RVLD_L** and **RRDVLD_L**.
9. After completing the pattern transmission, the UCIe Module must send {MBINIT.REPAIRVAL result req} sideband message to get the logged result.
10. The UCIe Module Partner must stop comparison and respond with {MBINIT.REPAIRVAL result resp} sideband message with result from the previous step. If detection is successful on **RRDVLD_L** only and not on **RVLD_L**, **TRDVLD_P/RRDVLD_P** is available for repair. If detection is successful on both **RVLD_L** and **RRDVLD_L** or only on **RVLD_L**, the interconnect cannot be repaired. If detection is unsuccessful on both **RVLD_L** and **RRDVLD_L**, **TRDVLD_P/RRDVLD_P** is not available and cannot be used for Valid Lane repair.
11. If repair is required on (**TVLD_P/RVLD_P**) and repair resource is available, the UCIe Module applies repair on its Valid Transmitter (see [Section 4.3.7](#)) and sends sideband message {MBINIT.REPAIRVAL apply repair req} with repair information. The UCIe Module Partner, after receiving the message, must apply repair on its Valid Receiver and must respond with sideband message {MBINIT.REPAIRVAL apply repair resp}.
12. If a repair is applied, device must check the repair success by repeating [Step 1](#) through [Step 4](#).
13. If repair is successful or repair is not required, the UCIe Module must send {MBINIT.REPAIRVAL done req} sideband message and the UCIe Module Partner must respond with {MBINIT.REPAIRVAL done resp}. When a UCIe Module has sent and received {MBINIT.REPAIRVAL done resp} sideband message, it must exit to REVERSALMB. Else if repair is unsuccessful or Valid Lane is unrepairable (in [Step 11](#)), the UCIe Module must exit to TRAINERROR after completing the TRAINERROR handshake.

For **Standard Package**, Valid Lane is checked for functional operation at the lowest data rate. Following is the flow:

1. The UCIe Module must send the sideband message {MBINIT.REPAIRVAL init req} and wait for a response. The UCIe Module Partner when ready to receive pattern on **RVLD_L**, must respond with {MBINIT.REPAIRVAL init resp}.
2. After receiving the sideband message {MBINIT.REPAIRVAL init resp}, the UCIe Module sends 128 iterations of VALTRAIN pattern (four 1's followed by four 0's) on **TVLD_L** along with the forwarded clock.
3. The UCIe Module Partner detects this pattern on **RVLD_L**. Detection is considered successful if at least 16 consecutive iterations of valid repair pattern are detected. The Receiver logs the detection result for **RVLD_L**.
4. After completing pattern transmission, the UCIe Module must send {MBINIT.REPAIRVAL result req} sideband message and wait to get the logged result.
5. The UCIe Module Partner must stop comparison and respond with {MBINIT.REPAIRVAL result resp} sideband message with result in the previous step.
6. If detection fails, the UCIe Module must exit to TRAINERROR after completing the TRAINERROR handshake.
7. If detection is successful, the UCIe Module must send {MBINIT.REPAIRVAL done req} sideband message and the UCIe Module Partner responds with {MBINIT.REPAIRVAL done resp}. When a UCIe Module has sent and received {MBINIT.REPAIRVAL done resp} sideband message, it must exit to REVERSALMB.

4.5.3.3.5 MBINIT.REVERSALMB

This state is entered only if Clock and Valid Lanes are functional. In this state, Data Lane reversal is detected. All the Transmitters and Receivers are enabled. The UCIe Module sets the forwarded clock phase at the center of the data UI. The UCIe Module Partner must sample the incoming Data with the incoming forwarded clock. The Track Transmitter is held low and the Track Receiver is permitted to be disabled. When not performing the actions relevant to this state:

- Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking)
- Clock Receivers are enabled
- Data and Valid Transmitters are held low
- Data and Valid Receivers are enabled

A 16-bit “Per Lane ID” pattern, shown in [Table 4-7](#), is a Lane-specific pattern using the Lane ID described in [Section 4.2.1](#). Example of “Per Lane ID” pattern for Lane 1 and Lane 31 are shown in [Table 4-8](#). When “Per Lane ID” pattern is used, it must not be scrambled.

Table 4-7. Per Lane ID Pattern

Pattern	0	1	0	1	Lane ID ^a												0	1	0	1
Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15				

a. Note that bit 0 of Lane ID maps to bit 4 in the Per Lane ID pattern, bit 1 to bit 5 and so on until bit 7 to bit 11.

Table 4-8. Per Lane ID Pattern Examples

Lane 1	0	1	0	1	1	0	0	0	0	0	0	0	0	1	0	1
Lane 31	0	1	0	1	1	1	1	1	0	0	0	0	0	1	0	1
Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Following is the Reversal MB sequence for **Advanced Package** and **Standard Package**:

1. The UCIe Module must send the {MBINIT.REVERSALMB init req} sideband message. When ready to receive "Per Lane ID" pattern and perform per-Lane pattern comparison, the UCIe Module Partner must respond with {MBINIT.REVERSALMB init resp}.
2. On Receiving the {MBINIT.REVERSALMB init resp} sideband message or entering from [Step 8](#), the UCIe Module must send {MBINIT.REVERSALMB clear error req} sideband message. Upon receiving this message, the UCIe Module Partner clears any prior errors and responds with {MBINIT.REVERSALMB clear error resp}. After receiving {MBINIT.REVERSALMB clear error resp}, the UCIe Module sends 128 iterations of Per Lane ID pattern (see [Table 4-7](#)) on all N data Lanes with correct Valid framing on the Valid Lane (see [Section 5.11](#) and [Section 4.1.2](#)) along with the forwarded clock. [Table 4-7](#) and [Table 4-8](#) show examples of the Per Lane ID pattern. N is 68 (64 Data + 4 RD) for a x64 Advanced Package. N is 34 (32 Data + 2 RD) for a x32 Advanced Package. N is 16 for a x16 Standard Package. N is 8 for a x8 Standard Package.
3. The UCIe Module Partner must perform a per-Lane compare on its Receivers on all N Lanes (see [Section 4.4](#)). Detection on a Lane is considered successful if at least 16 consecutive iterations of "Per Lane ID" pattern are detected. The UCIe Module Partner logs the detection result for its Receiver Lanes to be used for Lane reversal Detection.
4. After sending 128 iterations of "Per Lane ID" pattern, the UCIe Module stops sending the pattern and sends {MBINIT.REVERSALMB result req} sideband message to get the logged result.
5. The UCIe Module Partner stops comparison and must respond with {MBINIT.REVERSALMB result resp} sideband message with per Lane result (see [Table 7-11](#) for the message format).
6. If majority of the Lanes show success (since some Lanes may need repair), Lane reversal is not needed. Skip to [Step 11](#). Note that if exactly 50% of the Lanes showed success, Lane reversal is applied.
7. The UCIe Module applies Lane reversal on its Transmitters (see [Section 4.2](#)).
8. Following the Lane reversal application on its Transmitters, the UCIe Module repeats [Step 2](#) through [Step 5](#).
9. If majority of Lanes show success, the Lane reversal is needed. Lane reversal preserved for rest of the device operation. Skip to [Step 11](#).
10. The UCIe Module must exit to TRAINERROR after completing the TRAINERROR handshake.
11. The UCIe Module must send {MBINIT.REVERSALMB done req} sideband message and the UCIe Module Partner responds with {MBINIT.REVERSALMB done resp}. When the UCIe Module has sent and received {MBINIT.REVERSALMB done resp} sideband message, it must exit to REPAIRMB.

If a x64 Advanced Package Module that supports interoperation with a x32 Advanced Package module had received "UCIe-A x32" as 1b during parameter exchanges, it must recognize that it is connected to a x32 Advanced Package Module and appropriately interpret the received {MBINIT.REVERSALMB result resp} sideband message. In this scenario, the x64 applies steps 6 through 9 to lower 32 data lane set and 2 repair lane set. The x64 module applies Lane reversal (if required) within the lower 32 data lane set and 2 repair lane set.

If a x32 Advanced Package Module had received "UCIe-A x32" as 0b during parameter exchanges, it must recognize that it is connected to a x64 Advanced Package Module and appropriately interpret the

received {MBINIT.REVERSALMB result resp} sideband message, looking for majority of success in the lower 32 data lane set and 2 repair lane set of the x64 module (the x64 module will always place the results of its receiver on the lower half of its data/repair lane set).

If a x16 Standard Package Module that supports interoperation with a x8 Standard Package Module had its SPMW bit set to 1b OR has transmitted or received "UCIE-S x8" as 1b during parameter exchanges, the x16 Standard Package Module must recognize that it needs to operate in x8 mode and appropriately interpret the received {MBINIT.REVERSALMB result resp} sideband message. In this scenario, the x16 Standard Package Module applies [Step 6](#) through [Step 9](#) to the lower-8 data-lane set. Additionally, the x16 Standard Package Module applies Lane reversal (if required) within the lower-8 data-lane set.

When a x8 Standard Package Module receives the {MBINIT.REVERSALMB result resp} sideband message, the module must look for majority of success in the bits that correspond to the lower-8 data-lane set only.

4.5.3.3.6 MBINIT.REPAIRMB

This state is entered only after Lane reversal detection and application is successful. All the Transmitters and Receivers on a UCIE Module are enabled. The UCIE Module sets the clock phase at the center of the data UI on its mainband Transmitter for data Lanes (including the redundant Lanes for Advanced Package). The UCIE Module Partner must sample the incoming Data with the incoming forwarded clock on its mainband Receivers for data Lanes (including the redundant Lanes for Advanced Package). The Track Transmitter is held low and the Track Receiver is permitted to be disabled. When not performing the actions relevant to this state:

- Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking)
- Clock Receivers are enabled
- Data and Valid Transmitters are held low
- Data and Valid Receivers are enabled

In this state, the mainband Lanes are detected and repaired (if needed) for **Advanced Package**. In this state, functional checks and width degrade (if needed) are performed for **Standard Package**.

Following is the sequence for **Advanced Package**:

1. The UCIE Module must send the {MBINIT.REPAIRMB start req} sideband message and waits for a response. The UCIE Module Partner responds with {MBINIT.REPAIRMB start resp}.
2. The UCIE Module device performs Transmitter-initiated Data-to-Clock point test as described in [Section 4.5.1.1](#) on its Transmitter Lanes (all the data Lanes including the redundant Lanes). The Receiver must check for the pass/fail criteria on the data Lanes and the redundant Lanes.
 - a. The transmit pattern must be set up to send 128 iterations of continuous mode "Per Lane ID" Pattern. "Per Lane ID" pattern must not be scrambled. The Receiver must be set up to perform per Lane comparison.
 - b. Detection on a Receiver Lane is considered successful if at least 16 consecutive iterations of "Per Lane ID" pattern are detected.
 - c. LFSR Reset has no impact in MBINIT.REPAIRMB.
3. The UCIE Module receives the per-Lane pass/fail information over sideband message at the end of Transmitter initiated data to clock point test [Step 2](#).

4. If Lane repair is required and the necessary repair resources are available, the UCIe Module applies repair on its mainband Transmitters for data Lanes as described in [Section 4.3.1](#), and sends {MBINIT.REPAIRMB Apply repair req} sideband message. Upon receiving this sideband message, the UCIe Module Partner applies repair on its mainband Receivers for data Lanes as described in [Section 4.3.1](#), and sends {MBINIT.REPAIRMB Apply repair resp} sideband message. Otherwise, if the number of Lane failures are more than repair capability (see [Section 4.3](#)), the mainband is unrepairable and the UCIe Module must exit to TRAINERROR after performing the TRAINERROR handshake.
5. If repair is not required, perform [Step 7](#).
6. If Lane repair is applied ([Step 4](#)), the applied repair is checked by UCIe Module by repeating [Step 2](#) and [Step 3](#). If post repair Lane errors are logged in [Step 5](#), the UCIe Module must exit to TRAINERROR after performing the TRAINERROR handshake. If repair is successful, perform [Step 7](#).
7. The UCIe Module sends {MBINIT.REPAIRMB end req} sideband message and the UCIe Module Partner responds with {MBINIT.REPAIRMB end resp}. When UCIe Module has sent and received {MBINIT.REPAIRMB end resp}, it must exit to MBTRAIN.

For **Standard Package**, mainband is checked for functional operation at the lowest data rate. Following is the sequence of steps:

1. The UCIe Module sends the {MBINIT.REPAIRMB start req} sideband message and waits for a response. The UCIe Module Partner responds with the {MBINIT.REPAIRMB start resp} sideband message when ready to receive the pattern on its mainband Receivers for data Lanes.
2. The UCIe Module performs Transmitter-initiated Data-to-Clock point test as described in [Section 4.5](#).
 - a. The transmit pattern must be set up to send 128 iterations of continuous mode "Per Lane ID" Pattern. The Receiver must be set up to perform per Lane comparison.
 - b. Detection on a Receiver Lane is considered successful if at least 16 consecutive iterations of "Per Lane ID" pattern are detected.
 - c. LFSR Reset has no impact in MBINIT.REPAIRMB.
3. The UCIe Module must send the {MBINIT.REPAIRMB apply degrade req} sideband message indicating the functional Lanes on its Transmitter using one of the logical Lane map encodings from [Table 4-9](#). Encodings 100b and 101b can be used only when the SPMW bit is set to 1 or the UCIe-S x8 bit was set to 1 in the transmitted or received {MBINIT.PARAM configuration req} sideband message. If the remote Link partner indicated a width degrade in the functional Lanes, the UCIe Module must apply the corresponding width degrade to its Receiver. If the remote Link partner indicated all Lanes are functional (i.e. a Lane map code of 011b), the UCIe Module sets its Transmitter and Receiver to the width corresponding to the functional Lane encoding determined on its Transmitter. The UCIe Module sends the {MBINIT.REPAIRMB apply degrade resp} sideband message after setting its Transmitter and Receiver lanes to the relevant width. If the width on the Transmitter or Receiver has changed, both Link partners must repeat [Step 2](#). If the width on the Transmitter or Receiver has not changed, proceed to [Step 4](#). If a "Degrade not possible" encoding is sent or received in the {MBINIT.REPAIRMB apply degrade req} sideband message, the UCIe Module must exit to TRAINERROR after performing the TRAINERROR handshake.
4. The UCIe Module sends the {MBINIT.REPAIRMB end req} sideband message and the UCIe Module Partner responds with the {MBINIT.REPAIRMB end resp} sideband message. When the UCIe Module has sent and received the {MBINIT.REPAIRMB end resp} sideband message, it must exit to MBTRAIN.

Table 4-9. Standard Package Logical Lane Map

Lane Map Code	Functional Lanes
000b	None (Degrade not possible)
001b	Logical Lanes 0 to 7
010b	Logical Lanes 8 to 15
011b	0 - 15
100b	Logical Lanes 0 to 3
101b	Logical Lanes 4 to 7

IMPLEMENTATION NOTE

Consider an example in which Die A is communicating with Die B over a Standard Package PCIe Link.

During the first iteration of Step 2 of MBINIT.REPAIRMB, let's say that Tx on Die A detects errors on Lane ID 1 and not on any other Lanes, but Tx on Die B detects errors on Lane ID 10 and not on any other Lanes. Thus, as per the rules in Step 3, Die A sends {MBINIT.REPAIRMB apply degrade req} with a Lane map code of 010b. Similarly, in Step 3, Die B sends {MBINIT.REPAIRMB apply degrade req} with a Lane map code of 001b. The Rx on Die B disables Lanes 0 to 7, and the Tx on Die B tri-states Lanes 8 to 15. The Rx on Die A disables Lanes 8 to 15, and the Tx on Die A tri-states Lanes 0 to 7. Following the rules in Step 3, each die goes back and repeats Step 2.

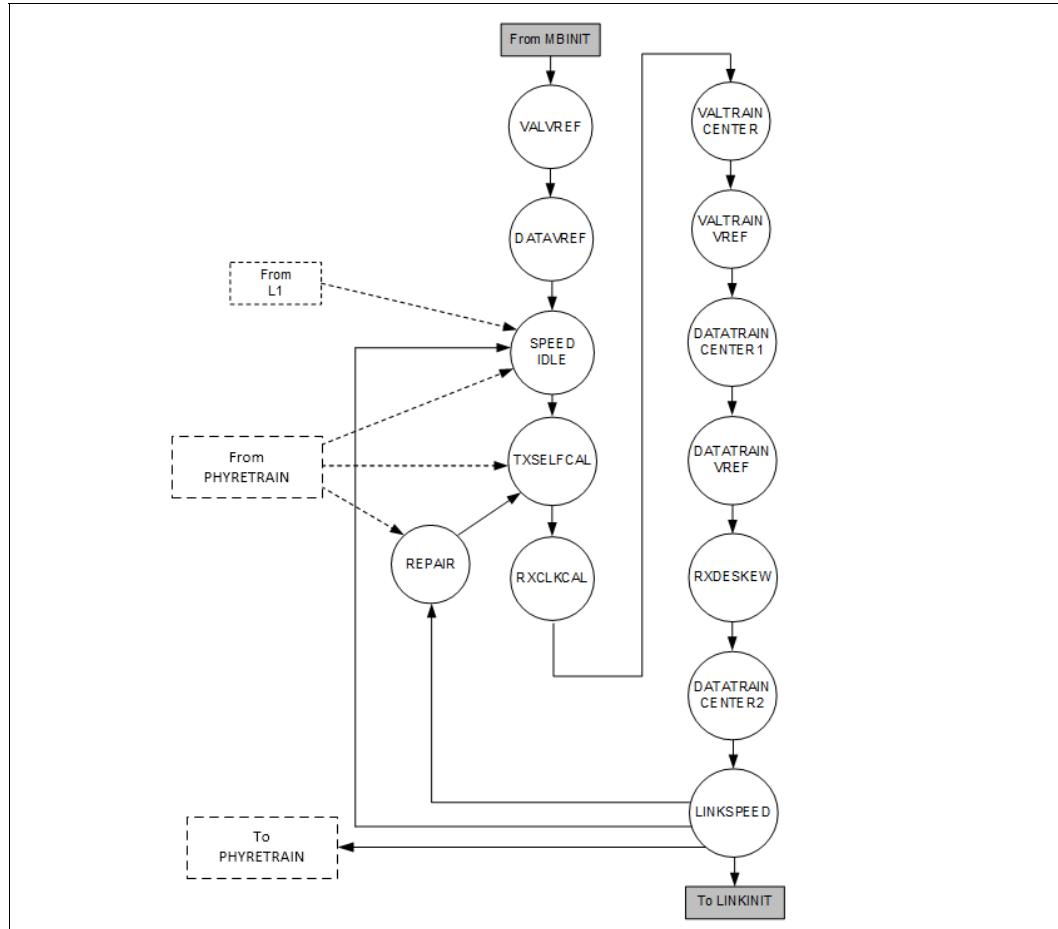
In this second iteration of Step 2, both Die know that some of the Lanes are disabled, and they will ignore the information related to the disabled Lanes in {Tx Init D to C results resp} (e.g., Die A will ignore the information related to Lanes 0 to 7 and perform only the Transmitter-initiated Data to Clock point test on Lanes 8 to 15).

Let's say that in the second iteration of Step 2, no errors are reported on the enabled Lanes. In Step 3, Die A sends {MBINIT.REPAIRMB apply degrade req} with a Lane map code of 010b. Similarly, in Step 3, Die B sends {MBINIT.REPAIRMB apply degrade req} with a Lane map code of 001b. Because the width of the Tx and Rx have not changed, both Die proceed to Step 4.

4.5.3.4 MBTRAIN

MBTRAIN state is used to setup operational speed and perform clock to data centering. At higher speeds, additional calibrations like Rx clock correction, Tx and Rx deskew may be needed to ensure Link performance. MBTRAIN uses sub-states to perform all the required calibration and training. PCIe Modules must enter each sub-state and the exit from each sub-state is coordinated between PCIe Module Partners through sideband handshakes. If a particular action within a sub-state is not needed, the PCIe Module is permitted to exit the sub-state through the relevant sideband handshake without performing the described operations in that sub-state.

Devices enter this state once the MBINIT is completed. This state is common for **Advanced** and **Standard Packages**.

Figure 4-41. Mainband Training

4.5.3.4.1 MBTRAIN.VALVREF

Receiver reference voltage (Vref) to sample the incoming Valid is optimized in this state. The data rate on the mainband continues to be at the lowest supported data rate (4 GT/s). The UCIe Module Partner must set the forwarded clock phase at the center of the data UI on its mainband Transmitters. The UCIe Module must sample the pattern on Valid signal with the forwarded clock. All data Lanes and Track must be held low during Valid Lane reference voltage training. Track Receivers are permitted to be disabled. When not performing the actions relevant to this state:

- Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking)
- Clock Receivers are enabled

The sequence for Valid Receiver reference voltage adjustment is as follows:

1. The UCIe Module must send the {MBTRAIN.VALVREF start req} sideband message and wait for a response. When {MBTRAIN.VALVREF start req} sideband message is received, the UCIe Module Partner responds with {MBTRAIN.VALVREF start resp}.

2. UCIe Module optimizes Vref on its Valid Receiver by adjusting Receiver reference voltage and performing one or more Receiver-initiated Data-to-Clock point tests (see [Section 4.5.1.3](#)) (AND/OR) one or more Receiver-initiated Data-to-Clock eye width sweeps (see [Section 4.5.1.4](#)).
 - a. The transmit pattern must be set to send 128 iterations of continuous mode “VALTRAIN” (four 1s and four 0s) pattern (see [Table 4-5](#)). This pattern must not be scrambled. The Receiver must be set up to perform comparison on the Valid Lane.
 - b. Detection on a Receiver Lane is considered successful if “VALTRAIN” pattern detection errors are less than the set threshold (per Lane comparison threshold in [Section 9.5.3.29](#)).
 - c. It should be noted that LFSR RESET has no impact in MBTRAIN.VALVREF.
3. The UCIe Module must send {MBTRAIN.VALVREF end req} sideband message after the Vref optimization (One way to perform Vref Optimization is to step through Vref and perform [Step 2](#) at each setting). When {MBTRAIN.VALVREF end req} is received, the UCIe Module Partner must respond with {MBTRAIN.VALVREF end resp}. When the UCIe Module has sent and received the sideband message {MBTRAIN.VALVREF end resp}, it must exit to MBTRAIN.DATAVREF.

4.5.3.4.2 MBTRAIN.DATAVREF

Receiver reference voltage (Vref) to sample the incoming data is optimized in this state. The data rate on the UCIe Module mainband continues to be at the lowest supported data rate (4 GT/s). The Transmitter sets the forwarded clock phase at the center of the data UI. The Track Transmitter is held low and the Track Receiver is permitted to be disabled. When not performing the actions relevant to this state:

- Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking)
- Clock Receivers are enabled
- Data and Valid Transmitters are held low
- Data and Valid Receivers are enabled

The sequence for data Receiver reference voltage adjustment is as follows:

1. The UCIe Module sends the sideband message {MBTRAIN.DATAVREF start req}. When {MBTRAIN.DATAVREF start req} sideband message is received, the UCIe Module Partner responds with {MBTRAIN.DATAVREF start resp}.
2. The UCIe Module optimizes data Vref by adjusting data Receiver reference voltage and performing one or more Receiver-initiated Data-to-Clock point tests (see [Section 4.5.1.3](#)) (AND/OR) one or more Receiver-initiated Data-to-Clock eye width sweeps (see [Section 4.5.1.4](#)).
 - a. The Transmitter must be set up to send 4K UI of continuous mode LFSR pattern described in [Section 4.4.1](#). The LFSR pattern on Data must be accompanied by correct Valid framing on the Valid Lane as described in [Section 4.1.2](#). The Receiver must be set up to perform per Lane comparison.
 - b. Detection on a Receiver Lane is considered successful if total error count is less than the set error threshold (see [Section 9.5.3.29](#)).
3. The UCIe Module must send {MBTRAIN.DATAVREF end req} sideband message after the Vref optimization (One way to perform Vref Optimization is to step through Vref and perform step 2 at each setting). When {MBTRAIN.DATAVREF end req} is received, the UCIe Module Partner must respond with {MBTRAIN.DATAVREF end resp}. Once the UCIe Module has sent and received the sideband message {MBTRAIN.DATAVREF end resp}, it must exit to MBTRAIN.SPEEDIDLE.

4.5.3.4.3 MBTRAIN.SPEEDIDLE

This is an electrical idle state to allow frequency change. Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking). Clock Receivers are enabled. Data, Valid, and Track Transmitters are held low.

The following rules apply:

1. The data rate is determined as follows:
 - If this state is entered from MBTRAIN.DATAVREF, the UCIe Module transitions to a data rate based on the highest common speed between the two devices (see [Section 4.5.3.3.1](#)).
 - Else if this state is entered from L1, the operating speed in the last ACTIVE state before entering L1 must be restored.
 - Else if this state is entered from LINKSPEED or PHYRETRAIN (speed degrade), and the current speed is not 4 GT/s, the next-lower data rate must be picked.
 - Else the UCIe Module must exit to TRAINERROR after performing the TRAINERROR handshake.
2. The width of the Link is set to the width previously determined at the exit of MBINIT.REPAIRMB or MBTRAIN.REPAIR (whichever was the most-recent state relative to entry into SPEEDIDLE).
3. Upon completing the transition to the required data rate, the UCIe Module must send {MBTRAIN.SPEEDIDLE done req} sideband message. When {MBTRAIN.SPEEDIDLE done req} sideband message is received, UCIe Module Partner responds with {MBTRAIN.SPEEDIDLE done resp}. Once the UCIe Module has sent and received the {MBTRAIN.SPEEDIDLE done resp} sideband message, it must exit to MBTRAIN.TXSELCAL.

4.5.3.4.4 MBTRAIN.TXSELCAL

The UCIe Module calibrates its circuit parameters independent of the UCIe Module Partner. Data, Clock, Valid, and Track Transmitters are tri-stated. Data, Clock, Valid, and Track Receivers are permitted to be disabled.

1. UCIe Module is permitted to perform implementation specific Transmitter-related calibration.
2. Upon completion of calibration, the UCIe Module must send the {MBTRAIN.TXSELCAL Done req} sideband message. When {MBTRAIN.TXSELCAL Done req} sideband message is received, the UCIe Module Partner must respond with {MBTRAIN.TXSELCAL Done resp}. When the UCIe Module has sent and received the {MBTRAIN.TXSELCAL Done resp} sideband message, it must exit to MBTRAIN.RXCLKCAL.

4.5.3.4.5 MBTRAIN.RXCLKCAL

In this state, Data, Valid Transmitters are held low (Data and Valid Receivers are permitted to be disabled). When not performing the actions relevant to this state, if Strobe mode was advertised by the UCIe Module partner, the Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking). When not performing the actions relevant to this state, if continuous clock mode was advertised by the UCIe Module partner and the {MBTRAIN.RXCLKCAL start req} sideband message has been received, then the Clock Transmitters are providing the free-running forwarded clock.

1. The UCIe Module, when ready to perform calibration on its Clock receive path, sends the {MBTRAIN.RXCLKCAL start req} sideband message. When the {MBTRAIN.RXCLKCAL start req} sideband message is received, the UCIe Module Partner starts sending the forwarded clock and Track. Subsequently, the UCIe Module Partner sends the {MBTRAIN.RXCLKCAL start resp} sideband message. The Transmitter clock must be free running and all Data Lanes and Valid must be held low. The UCIe Module is permitted to use the forwarded clock to perform any clock path-

related and Clock-to-Track-related calibration. The UCIe Module Partner must not adjust any circuit or PI phase parameters on its Transmitters within this state.

2. When the required calibration (if any) is performed, the UCIe Module sends {MBTRAIN.RXCLKCAL done req} sideband message. When {MBTRAIN.RXCLKCAL done req} is received, the UCIe Module Partner stops sending forwarded clock and responds by sending {MBTRAIN.RXCLKCAL done resp} sideband message. When a UCIe Module has sent and received {MBTRAIN.RXCLKCAL done resp} sideband message, it must exit to MBTRAIN.VALTRAINCENTER.

4.5.3.4.6 MBTRAIN.VALTRAINCENTER

To ensure the valid signal is functional, valid to clock training is performed before the data Lane training. The Receiver samples the pattern on Valid with the forwarded clock. Receiver reference voltage is set to the optimized value achieved through Vref training (see [Section 4.5.3.4.1](#) and [Section 4.5.3.4.2](#)). All data and Track Transmitters are held low during valid to clock training. When not performing the actions relevant to this state, if Strobe mode was advertised by the UCIe Module partner, then the Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking). When not performing the actions relevant to this state, if continuous clock mode was advertised by the UCIe Module partner, then the Clock Transmitters are providing the free-running forwarded clock.

Following is the MBTRAIN.VALTRAINCENTER sequence:

1. The UCIe Module sends a sideband message {MBTRAIN.VALTRAINCENTER start req}. The UCIe Module Partner responds with {MBTRAIN.VALTRAINCENTER start resp}.
2. The UCIe Module must perform one or more Transmitter-initiated Data-to-Clock eye width sweeps (see [Section 4.5.1.2](#)) (AND/OR) one or more Transmitter-initiated Data-to-Clock point tests (see [Section 4.5.1.1](#)) to determine the correct Valid to clock centering.
 - a. The transmit pattern must be set to send 128 iterations of continuous mode "VALTRAIN" (four 1s and four 0s) pattern (see [Table 4-5](#)). This pattern must not be scrambled. The Receiver must be set up to perform comparison on the Valid Lane.
 - b. Detection on a Receiver Lane is considered successful if "VALTRAIN" pattern detection errors are less than set threshold (per Lane comparison threshold in [Section 9.5.3.29](#)).
 - c. It should be noted that LFSR RESET has no impact in MBTRAIN.VALVREF.
3. The UCIe Module can use the received results log to assess valid functionality and margins. Following this, step 4 must be performed.
4. The UCIe Module must send {MBTRAIN.VALTRAINCENTER done req} sideband message. When {MBTRAIN.VALTRAINCENTER done req} is received the UCIe Module Partner responds with {MBTRAIN.VALTRAINCENTER done resp}. Once the UCIe Module has sent and received {MBTRAIN.VALTRAINCENTER done resp} sideband message, the UCIe Module must exit to MBTRAIN.VALTRAINVREF.

4.5.3.4.7 MBTRAIN.VALTRAINVREF

UCIe Module is permitted to optionally optimize the reference voltage (Vref) to sample the incoming Valid at the operating data rate. All Data and Track Transmitters are held low during Valid-to-Clock training. When not performing the actions relevant to this state, if Strobe mode was advertised by the UCIe Module partner, then the Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking). When not performing the actions relevant to this state, if continuous clock mode was advertised by the UCIe Module partner, then the Clock Transmitters are providing the free-running forwarded clock.

The sequence for Valid Receiver reference voltage adjustment is as follows:

1. The UCIe Module must send the sideband message {MBTRAIN.VALTRAINVREF start req}. When {MBTRAIN.VALTRAINVREF start req} sideband message is received, the UCIe Module Partner responds with {MBTRAIN.VALTRAINVREF start resp}.
2. UCIe Module optionally optimizes Vref by adjusting Receiver reference voltage on its Valid Receiver and performing one or more Receiver-initiated Data-to-Clock eye width sweeps (see [Section 4.5.1.4](#)) (AND/OR) one or more Receiver-initiated Data-to-Clock point tests (see [Section 4.5.1.3](#)). Step 2 is optional and implementation-specific.
 - a. If Valid centering is performed, the transmit pattern must be set to send 128 iterations of continuous mode "VALTRAIN" (four 1s and four 0s) pattern (see [Table 4-5](#)). This pattern must not be scrambled. The Receiver must be set up to perform comparison on the Valid Lane.
 - b. Detection on a Receiver Lane is considered successful if "VALTRAIN" pattern detection errors are less than set threshold (per Lane comparison threshold in [Section 9.5.3.29](#)).
 - c. It should be noted that LFSR RESET has no impact in MBTRAIN.VALVREF.
3. The UCIe Module must send {MBTRAIN.VALTRAINVREF end req} sideband message after the Vref optimization is complete. When {MBTRAIN.VALTRAINVREF end req} is received, the UCIe Module Partner must respond with {MBTRAIN.VALTRAINVREF end resp}. Once the UCIe Module has sent and received the sideband message {MBTRAIN.VALTRAINVREF end resp}, it must exit to MBTRAIN.DATATRAINCENTER1.

4.5.3.4.8 MBTRAIN.DATATRAINCENTER1

In this state, the UCIe Module performs Data-to-Clock training (including valid). LFSR patterns described in [Section 4.4.1](#) must be used in this state. The Track Transmitter is held Low. When not performing the actions relevant to this state:

- Clock Receivers are enabled
- Data and Valid Transmitters are held low
- Data and Valid Receivers are enabled
- If Strobe mode was advertised by the UCIe Module partner, then the Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking)
- If continuous clock mode was advertised by the UCIe Module partner, then the Clock Transmitters are providing the free-running forwarded clock

Following is the MBTRAIN.DATATRAINCENTER1 sequence:

1. The UCIe Module sends the {MBTRAIN.DATATRAINCENTER1 start req} sideband message. When {MBTRAIN.DATATRAINCENTER1 start req} sideband message is received, the UCIe Module Partner responds with {MBTRAIN.DATATRAINCENTER1 start resp}.
2. The UCIe Module must perform one or more Transmitter-initiated Data-to-Clock eye width sweeps (see [Section 4.5.1.2](#)) (AND/OR) one or more Transmitter-initiated Data-to-Clock point tests (see [Section 4.5.1.1](#)) to determine the correct data to clock centering and adjust Transmitter per-bit deskew (if needed).
 - a. The Transmitter must be set up to send 4K UI of continuous mode LFSR pattern described in [Section 4.4.1](#). The LFSR pattern on data must be accompanied by correct valid framing on the Valid Lane as described in [Section 4.1.2](#). The Receiver must be set up to perform per Lane comparison.
 - b. Detection on a Receiver Lane is considered successful if total error count is less than the set error threshold (see [Section 9.5.3.29](#)).

3. If the test is a success, the UCIe Module must set the clock phase to sample the data eye at the optimal point to maximize eye margins. The UCIe Module must send {MBTRAIN.DATATRAINCENTER1 end req} sideband message. When {MBTRAIN.DATATRAINCENTER1 end req} sideband message is received, the UCIe Module Partner responds with {MBTRAIN.DATATRAINCENTER1 end resp}. Once the UCIe Module has sent and received the {MBTRAIN.DATATRAINCENTER1 end resp} sideband message, it must exit to MBTRAIN.DATATRAINVREF.

4.5.3.4.9 MBTRAIN.DATATRAINVREF

UCIe Module is permitted to optionally optimize the reference voltage (Vref) on its data Receivers to optimize sampling of the incoming Data at the operating data rate. The Track Transmitter is held Low. When not performing the actions relevant to this state:

- Clock Receivers are enabled
- Data and Valid Transmitters are held low
- Data and Valid Receivers are enabled
- If Strobe mode was advertised by the UCIe Module partner, then the Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking)
- If continuous clock mode was advertised by the UCIe Module partner, then the Clock Transmitters are providing the free-running forwarded clock

The sequence for data Receiver reference voltage adjustment is as follows:

1. The UCIe Module must send the sideband message {MBTRAIN.DATATRAINVREF start req}. When {MBTRAIN.DATATRAINVREF start req} sideband message is received, the UCIe Module Partner responds with {MBTRAIN.DATATRAINVREF start resp}.
2. UCIe Module optionally optimizes Vref by adjusting Receiver reference voltage and performing one or more Receiver-initiated Data-to-Clock eye width sweeps (see [Section 4.5.1.4](#)) (AND/OR) one or more Receiver-initiated Data-to-Clock point tests (see [Section 4.5.1.3](#)). Step 2 is optional and implementation specific. If Data Vref optimization is performed:
 - a. The Transmitter must be set up to send 4K UI of continuous mode LFSR pattern described in [Section 4.4.1](#). The LFSR pattern on data must be accompanied by correct valid framing on the Valid Lane as described in [Section 4.1.2](#). The Receiver must be set up to perform per Lane comparison.
 - b. Detection on a Receiver Lane is considered successful if total error count is less than the set error threshold (see [Section 9.5.3.29](#)).
3. The UCIe Module must send {MBTRAIN.DATATRAINVREF end req} sideband message after the Vref optimization is complete. When {MBTRAIN.DATATRAINVREF end req} is received, the UCIe Module Partner responds with {MBTRAIN.DATATRAINVREF end resp}. Once the UCIe Module has sent and received the sideband message {MBTRAIN.DATATRAINVREF end resp}, it must exit to MBTRAIN.RXDESKEW.

Note: It is possible that the eye opening in this step is insufficient (test fails) and a per-bit deskew may be needed on the Receiver. Thus, the UCIe Module must exit to MBTRAIN.RXDESKEW.

4.5.3.4.10 MBTRAIN.RXDESKEW

The UCIe Module is permitted to optionally perform per Lane deskew on its Receivers to improve timing margin in this state. The Track Transmitter is held Low. When not performing the actions relevant to this state:

- Clock Receivers are enabled

- Data and Valid Transmitters are held low
- Data and Valid Receivers are enabled
- If Strobe mode was advertised by the UCIe Module partner, then the Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking)
- If continuous clock mode was advertised by the UCIe Module partner, then the Clock Transmitters are providing the free-running forwarded clock

Following is the MBTRAIN.RXDESKEW sequence:

1. The UCIe Module must send the sideband message {MBTRAIN.RXDESKEW start req}. When {MBTRAIN.RXDESKEW start req} sideband message is received, the UCIe Module Partner responds with {MBTRAIN.RXDESKEW start resp}.
2. UCIe Module optionally performs per Lane deskew on its Receivers by one or more Receiver-initiated Data-to-Clock eye width sweeps (see [Section 4.5.1.4](#)) (AND/OR) one or more Receiver-initiated Data-to-Clock point tests (see [Section 4.5.1.3](#)). Step 2 is optional and implementation specific. If per Lane deskew is performed:
 - a. The Transmitter must be set up to send 4K UI of continuous mode LFSR pattern described in [Section 4.4.1](#). The LFSR pattern on data must be accompanied by correct valid framing on the Valid Lane as described in [Section 4.1.2](#). The Receiver must be set up to perform per Lane comparison.
 - b. Detection on a Receiver Lane is considered successful if total error count is less than the set threshold (see [Section 9.5.3.29](#)).
3. The UCIe Module must send {MBTRAIN.RXDESKEW end req} sideband message after the deskew is performed or skipped. When {MBTRAIN.RXDESKEW end req} is received, the UCIe Module Partner must respond with {MBTRAIN.RXDESKEW end resp}. Once UCIe Module has sent and received the sideband message {MBTRAIN.RXDESKEW end resp}, it must exit to MBTRAIN.DATATRAINCENTER2.

4.5.3.4.11 MBTRAIN.DATATRAINCENTER2

This state is needed for the UCIe Module to recenter clock to aggregate data in case the UCIe Module Partner's Receiver performed a per Lane deskew. The Track Transmitter is held Low. When not performing the actions relevant to this state:

- Clock Receivers are enabled
- Data and Valid Transmitters are held low
- Data and Valid Receivers are enabled
- If Strobe mode was advertised by the UCIe Module partner, then the Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking)
- If continuous clock mode was advertised by the UCIe Module partner, then the Clock Transmitters are providing the free-running forwarded clock

Following is the MBTRAIN.DATATRAINCENTER2 sequence:

1. The UCIe Module sends the sideband message {MBTRAIN.DATATRAINCENTER2 start req}. When {MBTRAIN.DATATRAINCENTER2 start req} sideband message is received, the UCIe Module Partner responds with {MBTRAIN.DATATRAINCENTER2 start resp}.
2. The UCIe Module must perform one or more Transmitter-initiated Data-to-Clock eye width sweeps (see [Section 4.5.1.2](#)) (AND/OR) one or more Transmitter-initiated Data-to-Clock point tests (see [Section 4.5.1.1](#)) to determine the correct data-to-eye centering.

- a. The Transmitter must be set up to send 4K UI of continuous mode LFSR pattern described in [Section 4.4.1](#). The LFSR pattern on data must be accompanied by correct valid framing on the Valid Lane as described in [Section 4.1.2](#). The Receiver must be set up to perform per Lane comparison.
 - b. Detection on a Receiver Lane is considered successful if total error count is less than the set error threshold (see [Section 9.5.3.29](#)).
3. The UCIe Module uses the received training results to calculate the final eye center and set the clock phase to sample the data eye at the optimal point to maximize eye margins. The UCIe Module must send the {MBTRAIN.DATATRAINCENTER2 end req} sideband message. When {MBTRAIN.DATATRAINCENTER2 end req} sideband message is received, the UCIe Module Partner responds with {MBTRAIN.DATATRAINCENTER2 end resp}. Once UCIe Module has sent and received {MBTRAIN.DATATRAINCENTER2 end resp} sideband message, it must exit to MBTRAIN.LINKSPEED.

4.5.3.4.12 MBTRAIN.LINKSPEED

In this state, the UCIe Module checks Link stability at the operating date rate. The Track Transmitter is held Low. When not performing the actions relevant to this state:

- Clock Receivers are enabled
- Data and Valid Transmitters are held low
- Data and Valid Receivers are enabled
- If Strobe mode was advertised by the UCIe Module partner, then the Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking)
- If continuous clock mode was advertised by the UCIe Module partner, then the Clock Transmitters are providing the free-running forwarded clock

The following steps must be executed sequentially, when applicable:

1. The UCIe Module sends the sideband message {MBTRAIN.LINKSPEED start req}. When {MBTRAIN.LINKSPEED start req} sideband message is received, the UCIe Module Partner responds with {MBTRAIN.LINKSPEED start resp}.
2. The UCIe Module must perform a Transmitter-initiated Data-to-Clock point test (see [Section 4.5.1.1](#)) with the final clock sampling phase calculated in the previous MBTRAIN.DATACENTER2 state. LFSR pattern described in [Section 4.4.1](#) must be used in this state.
 - a. The Transmitter must be set up to send 4K UI of continuous mode LFSR pattern described in [Section 4.4.1](#). The LFSR pattern on data must be accompanied by correct valid framing on the Valid Lane as described in [Section 4.1.2](#). The Receiver must be set up to perform per Lane comparison.
 - b. Detection on a Receiver Lane is considered successful if total error count is less than the set threshold (see [Section 9.5.3.29](#)).

3. For single-Module instantiations, if errors are encountered, the UCIe Module sets its Transmitters to an electrical idle state and sends {MBTRAIN.LINKSPEED error req} sideband message. If an {MBTRAIN.LINKSPEED error req} sideband message is received, the UCIe Module Partner must complete [Step 1](#) and [Step 2](#), evaluate the results and if not initiating an {MBTRAIN.LINKSPEED exit to phy retrain req} sideband message, the UCIe Module Partner enters electrical idle on its Receiver and sends the {MBTRAIN.LINKSPEED error resp} sideband message. If an {MBTRAIN.LINKSPEED exit to phy retrain req} sideband message is received, the UCIe Module must exit to PHYRETRAIN and send an {MBTRAIN.LINKSPEED exit to PHY retrain resp} sideband message; any outstanding messages are abandoned. Otherwise, after the {MBTRAIN.LINKSPEED error resp} sideband message is received, the PHY_IN_RETRAIN flag is cleared and the following rules apply:
 - a. Based on the number of Lanes encountering errors, the UCIe Module checks if the failing Lanes can be repaired (for Advanced package) or Width degraded (for standard package). If Lanes can be repaired (for Advanced package) or Width degraded (for standard package), the UCIe Module must send {MBTRAIN.LINKSPEED exit to repair req} to the UCIe Module Partner. The UCIe Module Partner, if not initiating a speed degrade, enters MBTRAIN.REPAIR and sends the sideband message {MBTRAIN.LINKSPEED exit to repair resp}. If {MBTRAIN.LINKSPEED exit to repair resp} is received in response to a {MBTRAIN.LINKSPEED exit to repair req}, the UCIe Module must exit to MBTRAIN.REPAIR. If a UCIe Module is initiating a speed degrade, it must not respond to {MBTRAIN.LINKSPEED exit to repair req}.
 - b. If the Lanes cannot be repaired (for Advanced package) or width degraded (for Standard package), the speed must be degraded. The UCIe Module sends {MBTRAIN.LINKSPEED exit to speed degrade req} sideband message and waits for a response from the remote Link partner. The UCIe Module Partner must respond with {MBTRAIN.LINKSPEED exit to speed degrade resp}. Following this handshake, the UCIe Module must exit to MBTRAIN.SPEEDIDLE to set data rate to next lower speed.
 - c. If the UCIe Module receives an {MBTRAIN.LINKSPEED exit to speed degrade req} any outstanding {MBTRAIN.LINKSPEED exit to repair req} must be abandoned and the UCIe Module must respond to {MBTRAIN.LINKSPEED exit to speed degrade req}.
 - d. Any outstanding {MBTRAIN.LINKSPEED done req} must be abandoned if a UCIe Module has received a {MBTRAIN.LINKSPEED error req}.
4. For single- or multi-module instantiations, if no errors are encountered, the UCIe Module must set the clock phase on its Transmitter to sample the data eye at the optimal point to maximize eye margins. If PHY_IN_RETRAIN is not set for single-module instantiations, proceed to [Step 6](#). If PHY_IN_RETRAIN is not set, for multi-module instantiations, the UCIe Module must send the {MBTRAIN.LINKSPEED done req} (if not waiting for Link match criteria as a Retimer) to the remote UCIe Module Partner and wait for multi-module PHY Logic (MMPL) resolution in [Step 5c](#). If the PHY_IN_RETRAIN variable is set, the following actions must be taken:
 - a. If a change is detected in Runtime Link Testing Control register relative to the values at previous PHYRETRAIN entry, the UCIe Module must send {MBTRAIN.LINKSPEED exit to phy retrain req} and wait for a response. Upon receiving this message, the UCIe Module Partner must exit to PHY retrain and send {MBTRAIN.LINKSPEED exit to PHY retrain resp}. Once this sideband message is received, the UCIe Module must exit to PHY retrain.
 - b. Else if no change is detected in the Runtime Link Testing Control register relative to the values at previous PHYRETRAIN entry, Busy bit in Runtime Link Testing Status and PHY_IN_RETRAIN variable must be cleared and the UCIe Module must proceed to [Step 6](#).

5. For multi-module instantiations, if errors are encountered, the UCIe Module sets its Transmitters to an electrical idle state and sends {MBTRAIN.LINKSPEED error req} sideband message. If an {MBTRAIN.LINKSPEED error req} sideband message is received, the UCIe Module Partner must complete [Step 1](#) and [Step 2](#), evaluate the results and if not initiating an {MBTRAIN.LINKSPEED exit to phy retrain req} sideband message, the UCIe Module Partner enters electrical idle on its Receiver, and sends the {MBTRAIN.LINKSPEED error resp} sideband message. If an {MBTRAIN.LINKSPEED exit to phy retrain req} sideband message is received, the UCIe Module must exit to PHYRETRAIN and send an {MBTRAIN.LINKSPEED exit to PHY retrain resp} sideband message; any outstanding messages are abandoned. Otherwise, after the {MBTRAIN.LINKSPEED error resp} sideband message is received, the PHY_IN_RETRAIN flag is cleared and the following rules apply:
 - a. Based on the number of Lanes encountering errors, the UCIe Module checks whether the failing Lanes can be repaired (for Advanced Package) or Width degraded (for Standard Package). If Lanes can be repaired (for Advanced Package) or Width degraded (for Standard Package), the UCIe Module must send {MBTRAIN.LINKSPEED exit to repair req} to the UCIe Module Partner.
 - b. If the Lanes cannot be repaired (for Advanced Package) or width degraded (for Standard Package), the speed must be degraded. The UCIe Module sends {MBTRAIN.LINKSPEED exit to speed degrade req}.
 - c. The UCIe Module informs MMPL of local and remote error requests, done requests, or speed degrade requests, and waits for resolution. It also informs MMPL of any prior width degrade (for example in MBINIT.REPAIRMB), and MMPL treats this as the corresponding module requesting width degrade from the full operational width.
 - d. Based on the resolution flow chart in [Section 4.7](#), MMPL directs each Module to send either the {MBTRAIN.LINKSPEED exit to repair resp} (indicating next state is REPAIR), {MBTRAIN.LINKSPEED exit to speed degrade resp} (indicating next state is SPEEDIDLE with target speed to next-lower speed), {MBTRAIN.LINKSPEED multi-module disable module resp} (indicating next state is TRAINERROR and eventually RESET), or {MBTRAIN.LINKSPEED done resp} (indicating next state is LINKINIT). This is done regardless of the module's original error request or done request, and indicates the result of the resolution and next state to each module. The UCIe Module transitions to next state once it has sent and received the sideband response message that matches the expected resolution. Any mismatch on received message vs. expected resolution must take all modules to TRAINERROR. For Retimer dies, the resolution must take into account any Link match requirements, and while resolving the target configuration with remote Retimer partner, each UCIe Module from the Retimer die must send {MBTRAIN.LINKSPEED done resp} with stall encoding every 4 ms. The UCIe Retimer must ensure that this stall is not perpetual, and an implementation-specific timeout must be included in the Retimer. If {MBTRAIN.LINKSPEED done resp} with stall encoding is received, it must reset timers for state transition as well as any outstanding handshakes for multi-module resolution.

6. If the UCIe die is not a Retimer, proceed to [Step 7](#). If the UCIe die is a Retimer, the following rules apply to achieve Link match (if required):
 - a. Retimer must not send {MBTRAIN.LINKSPEED done req} unless the target Link speed and width of the remote Retimer partner resolves to current Link and width. Proceed to [Step 7](#) if Link match is achieved or if it is not required.
 - b. While resolving the target Link speed and width with the remote Retimer partner, if a Retimer has received an {MBTRAIN.LINKSPEED done req}, it must send {MBTRAIN.LINKSPEED done resp} with stall encoding every 4 ms. UCIe Retimer must ensure that this stall is not perpetual, and an implementation specific timeout must be included in the Retimer.
 - c. If the local UCIe Link speed or width is greater than the remote Retimer UCIe Link, then it must treat this as an error condition, and perform [Step 3](#) or [Step 5](#) with repair or speed degrade (whichever is applicable).
7. The UCIe Module must send {MBTRAIN.LINKSPEED done req} sideband message. When {MBTRAIN.LINKSPEED done req} is received, the UCIe Module must respond with {MBTRAIN.LINKSPEED done resp} and when a UCIe Module has sent and received the {MBTRAIN.LINKSPEED done resp} sideband message, both Transmitters and Receivers are now enabled and idle and both devices exit to LINKINIT.

4.5.3.4.13 MBTRAIN.REPAIR

This state can be entered from PHYRETRAIN or from MBINIT.LINKSPEED. For Advanced package, this state will be used to apply repair and for Standard package, this state will be used for Link width degrade. Track, Data, and Valid Transmitters are held low. Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking).

For **Advanced Package**, if the number of repair resources currently available is greater than the number of Lanes encountering errors, repair must be applied:

1. The UCIe Module sends the sideband message {MBTRAIN.REPAIR init req} for its Transmitter and the UCIe Module Partner responds with {MBTRAIN.REPAIR init resp}.
2. If Lane repair is possible, the UCIe Module applies repair on its Transmitter Lanes as described in [Section 4.3.1](#) and sends {MBTRAIN.REPAIR Apply repair req} sideband message. The UCIe Module Partner applies repair as described in [Section 4.3.1](#) and responds with {MBTRAIN.REPAIR Apply repair resp} sideband message once the required repair is applied.
3. The UCIe Module must send {MBTRAIN.REPAIR end req} sideband message and waits for a response. The UCIe Module Partner must then respond with {MBTRAIN.REPAIR end resp}. When a UCIe Module has sent and received {MBTRAIN.REPAIR end resp}, it must exit to MBTRAIN.TXSELCAL.

For a **x16 Standard package**, if the Lanes encountering errors are all contained within the set of Lane 0 through Lane 7 or Lane 8 through Lane 15, the width must be degraded to a x8 Link using the set of Lanes without errors (Lane 0 ... Lane 7 OR Lane 8 ... Lane 15). Likewise, for a **x8 Standard package**, if the Lanes encountering errors are all contained within the set of Lane 0 through Lane 3 or Lane 4 through Lane 7, the width must be degraded to a x4 Link using the set of Lanes without errors (Lane 0 through Lane 3 or Lane 4 through Lane 7).

1. The UCIe Module sends the sideband message {MBTRAIN.REPAIR init req} and the Receiver responds with {MBTRAIN.REPAIR init resp}.

2. The UCIe Module must send the {MBTRAIN.REPAIR apply degrade req} sideband message, indicating the functional Lanes on its Transmitter using one of the logical Lane map encodings from [Table 4-9](#). Encodings 100b and 101b can be used only when the SPMW bit is set to 1 or the UCIe-S x8 bit was set to 1 in the transmitted or received {MBINIT.PARAM configuration req} sideband message. If the remote Link partner indicated a width degrade in the functional Lanes, the UCIe Module must apply the corresponding width degrade to its Receiver. If the remote Link partner indicated all Lanes are functional, the UCIe Module sets its Transmitter and Receiver to the logical lane map corresponding to the functional Lane encoding determined on its Transmitter. The UCIe Module sends the {MBTRAIN.REPAIR apply degrade resp} sideband message after setting its Transmitter and Receiver lanes to the relevant logical lane map and proceeds to [Step 3](#) if a degrade is possible or if all Lanes are functional. If a "Degrade not possible" encoding is sent or received in the {MBTRAIN.REPAIR apply degrade req} sideband message, the UCIe Module must exit to TRAINERROR after performing the TRAINERROR handshake.
3. The UCIe Module must send the {MBTRAIN.REPAIR end req} sideband message and wait for a response. The UCIe Module Partner must then respond with the {MBTRAIN.REPAIR end resp} sideband message. When UCIe Module has sent and received the {MBTRAIN.REPAIR end resp} sideband message, the UCIe Module must exit to MBTRAIN.TXSELCAL.

4.5.3.5 LINKINIT

This state is used to allow die to die adapter to complete initial Link management before entering Active state on RDI. See [Section 10.1.6](#) for more details on RDI bring up flow. Track, Data, and Valid Transmitters are held low. If Strobe mode was advertised by the UCIe Module partner, then the Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking). If continuous clock mode was advertised by the UCIe Module partner, then the Clock Transmitters are providing the free-running forwarded clock. Clock Receivers are enabled.

Once RDI is in Active state, the PHY will clear its copy of "Start UCIe Link training" bit from UCIe Link control register.

The LFSR must be RESET upon entering this state.

This state is common for **Advanced Package** and **Standard Package** configurations.

4.5.3.6 ACTIVE

Physical layer initialization is complete, RDI is in Active state and packets from upper layers can be exchanged between the two dies.

All data in this state is scrambled using the scrambler LFSR described in [Section 4.4.1](#). Clock gating rules as described in [Section 5.11](#) apply.

This state is common for **Advanced Package** and **Standard Package** configurations.

4.5.3.7 PHYRETRAIN

A die can enter PHY retrain for a number of reasons. Track, Data, and Valid Transmitters are held low. Clock Transmitters are held differential low (for differential clocking) or simultaneous low (for Quadrature clocking). The trigger for PHY to enter PHY retrain is one of the following scenarios:

- Adapter directed PHY retrain: Adapter can direct the PHY to retrain for any reason it deems necessary (see [Section 10.3.3.4](#) Retrain State rules for more details and examples of Adapter-initiated Retrain requests).
- PHY initiated PHY retrain: Local PHY must initiate retrain on detecting a Valid framing error.

- Remote die requested PHY retrain: Local PHY must enter PHY retrain on receiving a request from the remote die.
- If a change is detected in Runtime Link Testing Control register during MBTRAIN.LINKSPEED.

A variable PHY_IN_RETRAIN must be set when entering PHYRETRAIN. For a multi-module configuration, the Retrain encoding (and hence the Retrain exit resolution) must be the same for all the modules which are part of the same multi-module Link. This is required because in a multi-module Link all modules must operate at the same speed and width (however, for Advanced package, it is possible for a module not needing repair to go through the Repair state and send the "No Repair" encoding in the corresponding {*apply repair*} messages).

Table 4-10. Runtime Link Test Status Register based Retrain encoding

Link Test Status Register		Retrain Encoding Resolution
Busy bit value	Repair Required	
0b	N/A	TXSELF CAL
1b	Repair needed	REPAIR (If repair resources are available)
		SPEEDIDLE (if unrepairable)
1b	No Repair	TXSELF CAL

Table 4-11. Retrain encoding

Retrain Encoding	State	Retain Condition
001b	TXSELF CAL	No Lane errors (Valid framing errors detected by PHY)
010b	SPEEDIDLE	Lane errors & faulty Lanes cannot be repaired
100b	REPAIR	Lane errors & faulty Lanes are repairable

Table 4-12. Retrain exit state resolution

Retrain reg Condition		Retrain Request Encoding		Resolved State Encoding		Exit
Die 0	Die 1	Die 0	Die 1	Die 0	Die 1	Both Dies
No Lane Errors	No Lane Errors	001b	001b	001b	001b	MBTRAIN.TXSELF CAL
No Lane Errors	Repair	001b	100b	100b	100b	MBTRAIN.REPAIR
No Lane Errors	Speed Degrade	001b	010b	010b	010b	MBTRAIN.SPEEDIDLE
Repair	No Lane Errors	100b	001b	100b	100b	MBTRAIN.REPAIR
Repair	Repair	100b	100b	100b	100b	MBTRAIN.REPAIR
Repair	Speed Degrade	100b	010b	010b	010b	MBTRAIN.SPEEDIDLE
Speed Degrade	No Lane Errors	010b	001b	010b	010b	MBTRAIN.SPEEDIDLE
Speed Degrade	Repair	010b	100b	010b	010b	MBTRAIN.SPEEDIDLE
Speed Degrade	Speed Degrade	010b	010b	010b	010b	MBTRAIN.SPEEDIDLE

4.5.3.7.1 Adapter initiated PHY retrain

Following is the sequence of steps for an Adapter initiated PHY retrain:

1. UCIe Module receives retrain request from the local Adapter (RDI state req moved to Retrain). Following this, the UCIe Module must complete the stall Req/Ack (`p1_stallreq; lp_stallack`) hand shake on RDI as described in [Chapter 10.0](#).
2. After completion of stall Req/Ack handshake and transmitting any pending data over mainband, UCIe Module must send sideband message {LinkMgmt.RDI.Req.Retrain} to UCIe Module Partner.
3. The UCIe Module Partner on receiving the sideband message {LinkMgmt.RDI.Req.Retrain} must transition its RDI state to Retrain after completion of stall Req/Ack (`p1_stallreq; lp_stallack`) handshake on its RDI and there is no mainband data pending in the Receiver pipeline. After completion of stall Req/Ack handshake and transmitting any pending data over mainband, the UCIe Module Partner responds with {LinkMgmt.RDI.Rsp.Retrain}.
4. Once {LinkMgmt.RDI.Rsp.Retrain} is received and there is no mainband data pending in the receiver pipeline, the UCIe Module must transition its RDI to Retrain.
5. UCIe Module must send {PHYRETRAIN.retrain start req} with retrain encoding reflecting the contents of Runtime Link Test Control register except the Start bit (if the Busy bit in Runtime Link Test Status Register is set). Following this, the UCIe Module Partner compares the received retrain encoding with the local retrain encoding. If received retrain encoding is the same as the local retrain encoding, the UCIe Module Partner must respond with {PHYRETRAIN.retrain start resp}. If the retrain encodings do not match, the UCIe Module Partner must resolve according to Retrain encodings and resolutions shown in [Table 4-10](#), [Table 4-11](#), and [Table 4-12](#) and then send {PHYRETRAIN.retrain start resp} with the resolved retrain encoding.
6. Once UCIe Module sends and receives the sideband message {PHYRETRAIN.retrain start resp}, it must exit to corresponding training state according to the resolved retrain register encoding.

4.5.3.7.2 PHY initiated PHY retrain

Following is the sequence of steps for PHY initiated PHY retrain:

1. On detecting a valid framing error, the UCIe Module must assert `p1_error` when transmitting that flit (or flit chunk) on RDI. Following this the UCIe Module (PHY) must complete the stall Req/Ack (`p1_stallreq; lp_stallack`) handshake on RDI.
2. The UCIe Module must send sideband message {LinkMgmt.RDI.Req.Retrain}.
3. The UCIe Module Partner on receiving the sideband message {LinkMgmt.RDI.Req.Retrain} must transition its RDI to retrain after completion of stall Req/Ack (`p1_stallreq; lp_stallack`) handshake on its RDI. Following that the UCIe Module Partner responds with {LinkMgmt.RDI.Rsp.Retrain}.
4. Once {LinkMgmt.RDI.Rsp.Retrain} is received, the UCIe Module must transition its RDI to Retrain.
5. UCIe Module must send {PHYRETRAIN.retrain start req} with retrain encoding reflecting the contents of Runtime Link Test Control register except the Start bit (if the Busy bit in Runtime Link Test Status Register is set). Following this, the UCIe Module Partner compares the received retrain encoding with the local retrain encoding. If received retrain encoding is the same as the local retrain encoding, the UCIe Module Partner must respond with {PHYRETRAIN.retrain start resp}. If the retrain encodings do not match, the UCIe Module Partner must resolve according to Retrain encodings and resolutions shown in [Table 4-10](#), [Table 4-11](#), and [Table 4-12](#) and then send {PHYRETRAIN.retrain start resp} with the resolved retrain encoding.
6. Once a UCIe Module has sent and received the sideband message {PHYRETRAIN.retrain start resp}, it must exit to corresponding training state according to the resolved retrain encoding.

4.5.3.7.3 Remote Die requested PHY retrain

1. On receiving {LinkMgmt.RDI.Req.Retrain}, the UCIe Module must transition local RDI to retrain after completion of stall Req/Ack (`pl_stallreq`; `lp_stallack`) handshake on its RDI. Following that the UCIe Module responds with {LinkMgmt.RDI.Rsp.Retrain}.
2. Once {LinkMgmt.RDI.Rsp.Retrain} is received, the UCIe Module Partner must transition its RDI to retrain.
3. UCIe Module must send {PHYRETRAIN.retrain start req} with retrain encoding reflecting the contents of Runtime Link Test Control register except the Start bit (if the Busy bit in Runtime Link Test Status Register is set). Following this, the UCIe Module Partner compares the received retrain encoding with the local retrain encoding. If received retrain encoding is the same as the local retrain encoding, the UCIe Module Partner responds with {PHYRETRAIN.retrain start resp}. If the retrain encodings do not match, the UCIe Module Partner must resolve according to Retrain encodings and resolutions shown in [Table 4-10](#), [Table 4-11](#), and [Table 4-12](#) and then send {PHYRETRAIN.retrain start resp} with the resolved retrain encoding.
4. Once a die has sent and received the sideband message {PHYRETRAIN.retrain start resp}, it must exit to corresponding training state according to the resolved retrain encoding.

4.5.3.7.4 PHY retrain from LINKSPEED

1. The UCIe Module must send {PHYRETRAIN.retrain start req} with retrain encoding reflecting the contents of Runtime Link Test Control register except the Start bit (if the Busy bit in Runtime Link Test Status Register is set). Following this, the UCIe Module Partner compares the received retrain encoding with the local retrain encoding. If received retrain encoding is the same as the local retrain encoding, the UCIe Module Partner must respond with {PHYRETRAIN.retrain start resp}. If the retrain encodings do not match, the UCIe Module Partner must resolve according to Retrain encodings and resolutions shown in [Table 4-10](#), [Table 4-11](#), and [Table 4-12](#) and then send {PHYRETRAIN.retrain start resp} with the resolved retrain encoding.
2. Once a die has sent and received the sideband message {PHYRETRAIN.retrain start resp}, it must exit to corresponding training state according to the resolved retrain encoding.

4.5.3.8 TRAINERROR

This state used as a transitional state due to any fatal or non-fatal events that need to bring the state machine back to RESET state. This can happen during initialization and training or if "Start UCIe Link training" bit from UCIe Link control register is set when state machine is not in RESET. It is also used for any events that transition the Link from a Link Up to a Link Down condition. Data, Valid, Clock, and Track transmitters are tri-stated, and their receivers are permitted to be disabled.

The exit from TRAINERROR to RESET is implementation specific. For cases when there is no error escalation (i.e., RDI is not in LinkError), it is recommended to exit TRAINERROR as soon as possible. For cases when there is error escalation (i.e., RDI is in LinkError), it is required for Physical Layer to be in TRAINERROR as long as RDI is in LinkError. To avoid problems with entering RESET while transmitting sideband packets, any in-progress sideband packets must finish transmission before entering RESET state.

See [Chapter 10.0](#) for correctable, non-fatal, and fatal error escalation on RDI.

This state is common for **Advanced Package** and **Standard Package** configurations.

If sideband is Active, a sideband handshake must be performed to enter TRAINERROR state from any state other than SINIT. The following is defined as the TRAINERROR handshake:

- The UCIe Module requesting exit to TRAINERROR must send {TRAINERROR Entry req} sideband message and wait for a response. The UCIe Module Partner must exit to TRAINERROR and

respond with {TRAINERROR Entry resp}. Once {TRAINERROR Entry resp} sideband message is received, the UCIe Module must exit to TRAINERROR. If no response is received for 8 ms, the LTSM transitions to TRAINERROR.

4.5.3.9 L1/L2

PM state allows a lower power state than dynamic clock gating in ACTIVE. Data, Valid, Clock, and Track transmitters are tri-stated, and their receivers are permitted to be disabled.

- This state is entered when RDI has transitioned to PM state as described in [Chapter 10.0](#). The PHY power saving features in this state are implementation specific.
- When local Adapter requests Active on RDI or remote Link partner requests L1 exit the PHY must exit to MBTRAIN.SPEEDIDLE. L1 exit is coordinated with the corresponding L1 state exit transitions on RDI.
- When local Adapter requests Active on RDI or remote Link partner requests L2 exit the PHY must exit to RESET. L2 exit is coordinated with the corresponding L2 state exit transitions on RDI.

4.6 Runtime Recalibration

Track signal can be used by the Receiver to perform periodic runtime calibration while in ACTIVE. Mainband data must continued to be sampled and processed (when accompanied by correct valid framing) during Runtime Recalibration. For unterminated Link, when not sending the required pattern the Track signal must alternate between being held low and held high (for anti aging) across consecutive Track recalibration iterations. For a terminated Link, when not sending the required pattern, the Track transmitter must go to Hi-Z.

The following sequence is used to request track pattern:

1. The UCIe Module enables the Track signal buffers on its Receiver and sends a {RECAL.track pattern init req} sideband message, and then waits for a response.
2. The UCIe Module Partner sends {RECAL.track pattern init resp} and enables its Track signal Transmitter (is preconditioned to drive low if needed). Following this, the UCIe Module Partner's Track Transmitter starts sending the pattern described in [Section 5.5.1](#), along with the forwarded clock. If the link is in Clock-gated mode, the UCIe Module Partner should enable the clock and manage whether the Link should return to Clock-gated mode after the Track update is complete.
3. The UCIe Module on its Receiver performs the required recalibration and sends the {RECAL.track pattern done req} sideband message.
4. Upon receiving this message, the UCIe Module Partner's Track Transmitter stops sending the pattern and sends the {RECAL.track pattern done resp} sideband message.
5. The UCIe Module is permitted to disable the Track Receiver upon receiving the {RECAL.track pattern done resp} sideband message.

4.7 Multi-module Link

As described in [Chapter 1.0](#), the permitted configurations for a multi-module Link are one-, two-, and four-module configurations. In a multi-module Link, each module is assigned a dedicated Module Identifier (Module ID), which is advertised to the remote Link partner during MBINIT.PARAM. [Chapter 5.0](#) defines the permitted combinations of Module ID assignments for the different scenarios of Multi-module instantiations that must be supported by multi-module implementations.

4.7.1 Multi-module initialization

Each module in a multi-module configuration must initialize and train independently, using its sideband. If two or four modules are used, a separate multi-module PHY logic block coordinates across the modules, as described in [Section 1.2.2](#). The MMPL is responsible for orchestrating data transfer and any associated byte swizzling for the Transmitters across the multiple modules such that the remote Link partner's Receivers observe the correct byte-to-Lane mapping (i.e., for any valid transfer, bytes are laid out from LSB to MSB in ascending order of Module ID and Lane ID across all the active Lanes). [Figure 4-42](#), [Figure 4-43](#), [Figure 4-44](#), and [Figure 4-45](#) illustrate examples of the aforementioned byte swizzling for some of the Standard Package configurations. M0, M1, M2, and M3 in the figures correspond to Module ID 0, Module ID 1, Module ID 2, and Module ID 3, respectively. The figures provide the RDI byte-to-Module mapping. [Figure 4-42](#) shows the scenario in which the remote Link partner's Module ID is the same. [Figure 4-43](#) shows an example of a scenario in which the remote Link partner's Module ID is different. [Figure 4-44](#) shows an example of width degradation for Standard package in which the remote Link partner's Module ID is different (note that the bytes are laid out in ascending order of Module ID and Lane ID across all the active Lanes at the Receiver in all cases). [Figure 4-45](#) shows a scenario in which two modules are disabled. This corresponds to a case outlined in [Chapter 5.0](#) in which a stacked configuration is connected with an unstacked configuration and the M0 and M2 modules are disabled; the remaining bytes of RDI are sent over subsequent 8-UI intervals such that M1 on the remote Link partner receives the least significant bytes.

Figure 4-42. Example of Byte Mapping for Matching Module IDs

Module Name	M1	M0	M2	M3				
Data Bytes	RDI B16 --- RDI B31	RDI B16 --- RDI B31	RDI B0 --- RDI B15	RDI B0 --- RDI B15	RDI B32 --- RDI B47	RDI B32 --- RDI B47	RDI B48--- RDI B63	RDI B48--- RDI B63
Tx/Rx	Rx (x16)	Tx (x16)	Rx (x16)	Tx (x16)	Rx (x16)	Tx (x16)	Rx (x16)	Tx (x16)
Tx/Rx	Tx (x16)	Rx(x16)	Tx (x16)	Rx(x16)	Tx (x16)	Rx(x16)	Tx (x16)	Rx(x16)
Data Bytes	RDI B16 --- RDI B31	RDI B16 --- RDI B31	RDI B0 --- RDI B15	RDI B0 --- RDI B15	RDI B32 --- RDI B47	RDI B32 --- RDI B47	RDI B48--- RDI B63	RDI B48--- RDI B63
Module Name	M1	M0	M2	M3				

Figure 4-43. Example of Byte Mapping for Differing Module IDs

Module Name	M1	M0	M2	M3				
Data Bytes	RDI B16 --- RDI B31	RDI B48 --- RDI B63	RDI B0 --- RDI B15	RDI B32 --- RDI B47	RDI B32 --- RDI B47	RDI B0 --- RDI B15	RDI B48--- RDI B63	RDI B16--- RDI B31
Tx/Rx	Rx (x16)	Tx (x16)	Rx (x16)	Tx (x16)	Rx (x16)	Tx (x16)	Rx (x16)	Tx (x16)
Tx/Rx	Tx (x16)	Rx(x16)	Tx (x16)	Rx(x16)	Tx (x16)	Rx(x16)	Tx (x16)	Rx(x16)
Data Bytes	RDI B16 --- RDI B31	RDI B48 --- RDI B63	RDI B0 --- RDI B15	RDI B32 --- RDI B47	RDI B32 --- RDI B47	RDI B0 --- RDI B15	RDI B48--- RDI B63	RDI B16 --- RDI B31
Module Name	M3	M2	M0	M1				

Figure 4-44. Example of Width Degradation with Byte Mapping for Differing Module IDs

Module Name	M1	M0	M2	M3				
Data Bytes UI 15-8	RDI B40 --- RDI B47	RDI B57-- RDI B63	RDI B32 --- RDI B39	RDI B48 --- RDI B55	RDI B48 --- RDI B55	RDI B32 --- RDI B39	RDI B57-- RDI B63	RDI B40 --- RDI B47
Data Bytes UI 7-0	RDI B8 --- RDI B15	RDI B24-- RDI B31	RDI B0 -- RDI B7	RDI B16 --- RDI B23	RDI B16 --- RDI B23	RDI B0 -- RDI B7	RDI B24-- RDI B31	RDI B8 -- RDI B15
Tx/Rx	Rx (x8)	Tx (x8)	Rx (x8)	Tx (x8)	Rx (x8)	Tx (x8)	Rx (x8)	Tx (x8)
Tx/Rx	Tx (x8)	Rx (x8)	Tx (x8)	Rx (x8)	Tx (x8)	Rx (x8)	Tx (x8)	Rx (x8)
Data Bytes UI 7-0	RDI B8 -- RDI B15	RDI B24 -- RDI B31	RDI B0 -- RDI B7	RDI B16 --- RDI B23	RDI B16 --- RDI B23	RDI B0 -- RDI B7	RDI B24-- RDI B31	RDI B8 -- RDI B15
Data Bytes UI 15-8	RDI B40 -- RDI B47	RDI B57-- RDI B63	RDI B32 --- RDI B39	RDI B48 --- RDI B55	RDI B48 --- RDI B55	RDI B32 --- RDI B39	RDI B57-- RDI B63	RDI B40 -- RDI B47
Module Name	M3	M2	M0	M1				

Figure 4-45. Example of Byte Mapping with Module Disable

Module Name	M1	M0	M2	M3		
Data Bytes	RDI B0 --- RDI B15	RDI B16 --- RDI B31			RDI B16 --- RDI B31	RDI B0 --- RDI B15
Tx/Rx	Rx (x16)	Tx (x16)	Rx (x16)	Tx (x16)	Rx (x16)	Tx (x16)
			Disabled		Disabled	
Tx/Rx	Tx (x16)	Rx(x16)	Tx (x16)	Rx(x16)	Tx (x16)	Rx(x16)
Data Bytes	RDI B0 -- RDI B15	RDI B16 -- RDI B31			RDI B16 -- RDI B31	RDI B0 -- RDI B15
Module Name	M3	M2	M0	M1		

Each module in a multi-module Link must operate at the same width and speed. During initialization or retraining, if any module failed to train, the MMPL must ensure that the multi-module configuration degrades to the next permitted configuration for width or speed degrade (see [Figure 4-46](#) and [Figure 4-47](#)). Subsequently, any differences in width and speed between the different modules must be resolved using the following rules:

1. For Standard package multi-module configuration, if width degrade is reported for any of the modules:
 - a. If less than or equal to half the number of modules report width degrade at the current Link speed, the corresponding Modules must be disabled. The MMPL must ensure that the multi-module configuration degrades to the next permitted configuration for width or speed degrade (see [Figure 4-46](#) and [Figure 4-47](#)). For example, if three out of four modules are active, the MMPL must degrade the Link to a two-module configuration.
 - b. If the majority of modules report width degrade at the current Link speed, see the pseudo code below:

```

CLS: Current Link Speed
CLS-1: Next lower allowed Link Speed
M is number of active Modules
Aggregate Raw BW(M, CLS) = Common Minimum Link width * M * CLS
If modules report Width degrade:
  If CLS = 4 GT/s
    Apply Width degrade for all modules
  Else If Aggregate Raw BW(M, (CLS-1)) > Aggregate raw BW (M/2, CLS):
    Attempt Speed Degrade
  Else:
    Apply Width degrade for all modules
  
```

2. For Advanced or Standard package multi-module configuration, if any Module reports speed difference, see the pseudo code below:

```

IF modules report speed difference:
    CMLS: Common Maximum Link Speed
    HMLS: Highest Maximum Link Speed of next lower configuration
    IF HMLS/2 > CMLS:
        Modules degrade to next lower configuration
    Else:
        Speed for all modules degrades to CMLS

```

Figure 4-46 and Figure 4-47 provide a consolidated view of the above two rules as a flow chart that Advanced Package and Standard Package implementations, respectively, must follow. Note that the "Yes" condition for $HMLS/2 > CMLS$ question is there to cover the base case of 4 GT/s. In other words, if some module(s) passed MBINIT but failed 4 GT/s in LinkSpeed, then the "Yes" arc will result in module disable instead of TrainError (because CMLS will be 0 for that) and provide the opportunity to remain operational at 4 GT/s for the modules that were still operational at 4 GT/s.

Figure 4-46. Decision Flow Chart for Multi-module Advanced Package

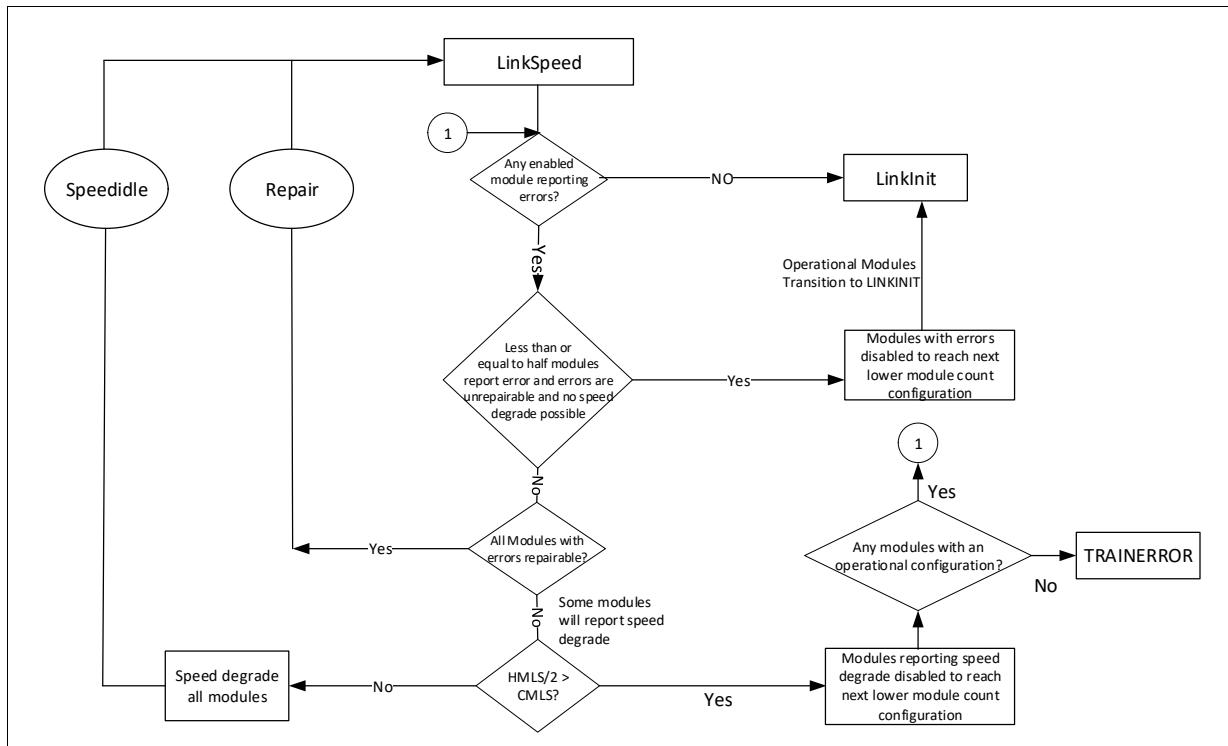
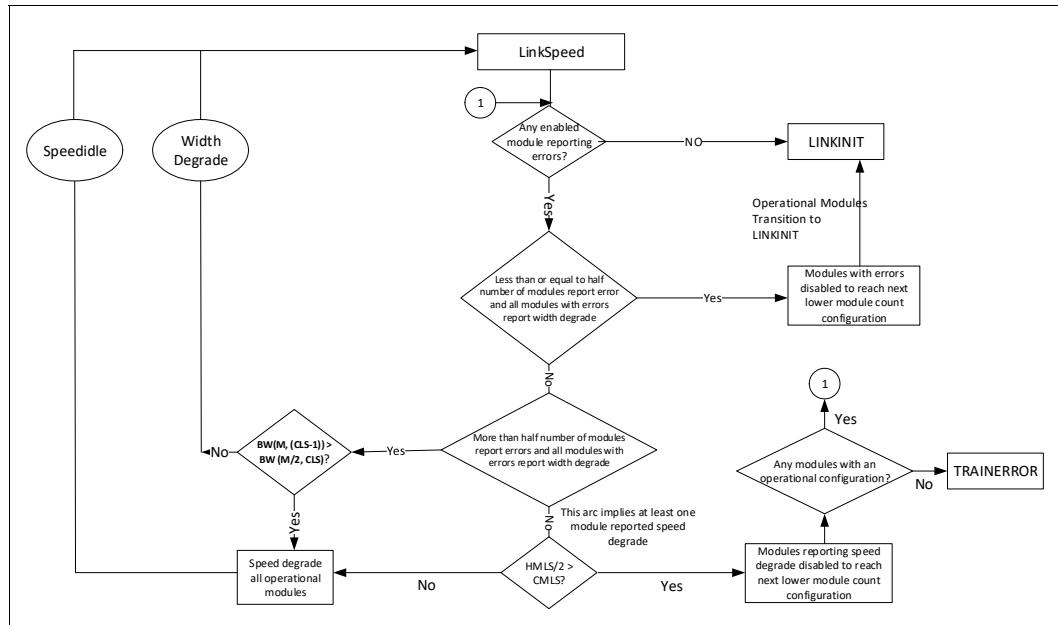


Figure 4-47. Decision Flow Chart for Multi-module Standard Package

4.7.1.1 Sideband Assignment and Retimer Credits for Multi-module Configurations

During Link initialization, training and retraining (see [Section 4.5](#)) certain sideband packets are sent on individual module sideband interfaces. These include all the messages in [Table 7-9](#) and [Table 7-11](#), and any vendor-defined messages that were defined as such.

Other sideband packets use a single sideband to send sideband packets. These include Register Access packets (requests as well as completions), all the non-vendor defined messages in [Table 7-8](#) and [Table 7-10](#), and any vendor-defined messages that were defined as such. A device must send these sideband packets on the sideband interface of the numerically least Module ID whose LTSM is not in RESET or SBINIT. A packet sent on a given Module ID could be received on a different Module ID on the sideband Receiver.

Similarly, Retimer credits are returned on the Valid signal of the numerically least Module ID whose LTSM is in Active state. Credits sent on a given Module ID could be received on a different Module ID on the remote Link partner.

4.7.1.2 Examples of MMPL Synchronization

When a module is part of a multi-module UCIe Link, it performs individual training steps through [Step 2](#) of MBTRAIN.LINKSPEED independent of other modules using its own sideband Link. This is true for both Link Initialization and Link Retraining. The common Link parameters exchanged in MBINIT.PARAM are the same for all modules that are part of the multi-module Link (for example the "Maximum Data Rate"). Thus, when in MBTRAIN.LINKSPEED, all modules of a multi-module Link are operating at the same data rate.

The synchronization orchestration by MMPL happens in the MBTRAIN.LINKSPEED state based on the rules outlined in [Section 4.7.1](#). As outlined in [Section 4.5.3.4.12](#), after [Step 2](#) of MBTRAIN.LINKSPEED has completed and PHY_IN_RETRAIN is not set:

- If no errors are encountered for a module, an {MBTRAIN.LINKSPEED done req} is sent to the remote Link partner module
- If errors are encountered for a module, an {MBTRAIN.LINKSPEED error req}/ {MBTRAIN.LINKSPEED error resp} handshake is performed followed by sending an {MBTRAIN.LINKSPEED exit to repair req} or an {MBTRAIN.LINKSPEED exit to speed degrade req} on that module's sideband.

The individual modules notify the MMPL of the sent and received information from these sideband messages. MMPL collects this information from all the modules which are operational in the Link and determines the next state based on resolution of the rules outlined in [Section 4.7.1](#). Of course, the case without errors is when all modules sent and received the {MBTRAIN.LINKSPEED done req} message with no change to Link width. In this scenario, the MMPL directs them to proceed to [Step 6](#) of MBTRAIN.LINKSPEED.

The following sections cover a few examples of this resolution for a Link with four modules and Standard Package configuration where errors were encountered.

4.7.1.2.1 Example 1: MMPL Resolution results in a Width Degrade per Module

In this example, the four modules of the UCIe Link are in MBTRAIN.LINKSPEED at 8 GT/s. [Table 4-13](#) shows the exchanged messages for one die (in the case where errors are encountered, it is assumed that the {MBTRAIN.LINKSPEED error req}/ {MBTRAIN.LINKSPEED error resp} has completed before the messages shown). Because the resolution is consistent in using the sent and received messages, both die of the Link will reach the same resolution.

Table 4-13. Messages exchanged that are used to determine resolution for Example 1

Module Identifier	Sent Message	Received Message
Module 0	{MBTRAIN.LINKSPEED done req}	{MBTRAIN.LINKSPEED done req}
Module 1	{MBTRAIN.LINKSPEED exit to repair req}	{MBTRAIN.LINKSPEED done req}
Module 2	{MBTRAIN.LINKSPEED done req}	{MBTRAIN.LINKSPEED exit to repair req}
Module 3	{MBTRAIN.LINKSPEED exit to repair req}	{MBTRAIN.LINKSPEED exit to repair req}

In this example, 3 out of the 4 modules have either sent or received the {MBTRAIN.LINKSPEED exit to repair req} message which indicate a width degrade for Standard Package configurations. The value of "CLS" (Current Link Speed) is 8 GT/s, and the value of "CLS-1" is 4 GT/s. The value of "M" (Number of Active Modules) is 4. Because BW (4 Links at 4 GT/s) is not greater than BW (2 Links at 8 GT/s), the flow chart in [Figure 4-46](#) would result in the MMPL notifying all the modules to proceed with a width degrade by moving to MBTRAIN.REPAIR as the next state (i.e., {MBTRAIN.LINKSPEED exit to repair resp} will be sent on each Module). Note that "CLS" and "M" are re-computed using the updated information every time the Link is MBTRAIN.LINKSPEED and there is a corresponding MMPL resolution. Width degrade is applied per module following the steps in MBTRAIN.REPAIR for every module in this UCIe Link. For the module where no errors were encountered, the transmitter is permitted to pick either of Lanes 0 to 7 or Lanes 8 to 15 as the operational Lanes when transmitting the {MBTRAIN.REPAIR apply degrade req} to the remote Link partner module. Following the exit from MBTRAIN.REPAIR, the training continues through the substates of MBTRAIN and in the next iteration of MBTRAIN.LINKSPEED, if no errors are encountered, MMPL will direct the modules to proceed to [Step 6](#) of MBTRAIN.LINKSPEED.

Note that this example is covering the case where errors occurred during the LINKSPEED state. If the width of a module is already lower from the rest of the operational modules that are part of a multi-module Link (e.g., if a module had degraded width during MBINIT.REPAIRMB itself), it may have sent and received {MBTRAIN.LINKSPEED done req} during LINKSPEED. However, from a MMPL resolution

perspective, MMPL must treat this as a Module reporting errors requiring width degrade. This is because a multi-module Link requires all modules to operate at the same width and speed.

4.7.1.2.2 Example 2: MMPL Resolution results in a Speed Degrade

In this example, the four modules of the UCIe Link are in MBTRAIN.LINKSPEED at 16 GT/s. Table 4-14 shows the exchanged messages for one die (in the case where errors are encountered, it is assumed that the {MBTRAIN.LINKSPEED error req}/ {MBTRAIN.LINKSPEED error resp} has completed before the messages shown). Because the resolution is consistent in using the sent and received messages, both die of the Link will reach the same resolution.

Table 4-14. Messages exchanged that are used to determine resolution for Example 2

Module Identifier	Sent Message	Received Message
Module 0	{MBTRAIN.LINKSPEED exit to repair req}	{MBTRAIN.LINKSPEED done req}
Module 1	{MBTRAIN.LINKSPEED done req}	{MBTRAIN.LINKSPEED done req}
Module 2	{MBTRAIN.LINKSPEED done req}	{MBTRAIN.LINKSPEED done req}
Module 3	{MBTRAIN.LINKSPEED done req}	{MBTRAIN.LINKSPEED exit to speed degrade req}

In this example, Module 3 has received a message indicating that the remote partner wants to speed degrade. "CMLS" always maps to the next degraded Link speed and so in this case "CMLS" is 12 GT/s. "HMLS" always ends up mapping to current Link speed and so in this case it is 16 GT/s. Because Module 3 received a speed degrade request, following the flow chart in Figure 4-47, this would result in MMPL notifying all the modules to proceed with a speed degrade by moving to MBTRAIN.SPEEDIDLE (i.e. {MBTRAIN.LINKSPEED exit to speed degrade resp} will be sent on each Module). Following the exit from MBTRAIN.SPEEDIDLE, the training continues through the substates of MBTRAIN and in the next iteration of MBTRAIN.LINKSPEED, if no errors are encountered, MMPL will direct the modules to proceed to Step 6 of MBTRAIN.LINKSPEED. Note that "CMLS" and "HMLS" are using the updated information every time the Link is MBTRAIN.LINKSPEED and there is a corresponding MMPL resolution. In this example, for the next iteration, CMLS will be 8 GT/s and HMLS will be 12 GT/s.

4.7.1.2.3 Example 3: MMPL Resolution results in a Module Disable

In this example, the four modules of the UCIe Link are in MBTRAIN.LINKSPEED at 16 GT/s. Table 4-15 shows the exchanged messages for one die (in the case where errors are encountered, it is assumed that the {MBTRAIN.LINKSPEED error req}/ {MBTRAIN.LINKSPEED error resp} has completed before the messages shown). Because the resolution is consistent in using the sent and received messages, both die of the Link will reach the same resolution.

Table 4-15. Messages exchanged that are used to determine resolution for Example 3

Module Identifier	Sent Message	Received Message
Module 0	{MBTRAIN.LINKSPEED done req}	{MBTRAIN.LINKSPEED done req}
Module 1	{MBTRAIN.LINKSPEED exit to repair req}	{MBTRAIN.LINKSPEED done req}
Module 2	{MBTRAIN.LINKSPEED done req}	{MBTRAIN.LINKSPEED done req}
Module 3	{MBTRAIN.LINKSPEED done req}	{MBTRAIN.LINKSPEED done req}

Because less than half of the modules are reporting errors and requesting a width degrade, as per the flow chart in Figure 4-47, MMPL would take the configuration to a two module configuration. As per the rules in Section 5.7.3.4.1, Module 0 and Module 1 would send the {MBTRAIN.LINKSPEED multi-module disable module resp} to take these modules to TRAINERROR and RESET. Module 2 and Module 3 would send the {MBTRAIN.LINKSPEED done resp} to take them to LINKINIT.

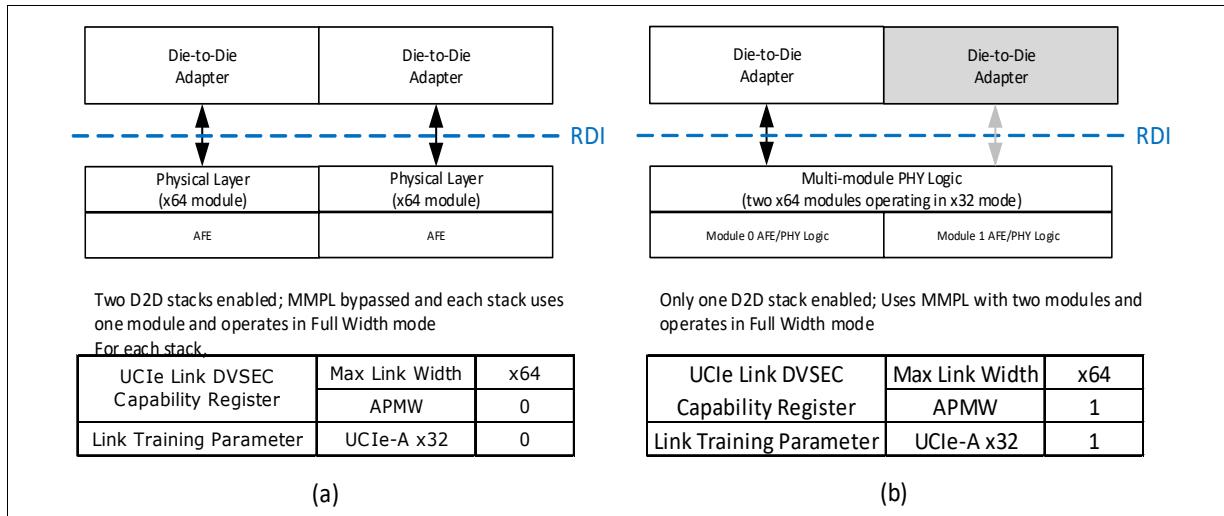
4.7.2 Multi-module Interoperability between x64 and x32 Advanced Packages

MMPL is responsible for the appropriate byte swizzling and width adjustment when a multi-module x64 Advanced Package module is connected to a corresponding multi-module x32 Advanced Package module. All the modules in a multi-module configuration must be of the same type (in this context, all the modules within a multi-module set must be x64 Advanced or x32 Advanced). All the rules related to module naming conventions and disabled configurations apply to x32 Advanced Package Modules as well.

One example of interoperation between UCIe-A x64 and UCIe-A x32 is when the UCIe-A x64 Stack (including RDI and FDI maximum throughput) is bandwidth-matched (Full Width Mode) by the remote Link partner's maximum throughput for a given interface. [Figure 4-48](#) shows an example of two x64 modules that are capable of operating as two independent UCIe stacks with independent Adapters and Protocol Layers (bypass MMPL logic in configuration (a) in [Figure 4-48](#)) and is also capable of operating as a multi-module configuration when connected to a corresponding multi-module configuration of x32 Advanced Package to achieve the equivalent bandwidth of a single x64 module (configuration (b) in [Figure 4-48](#)). In the latter configuration, one of the Adapters (shown in gray) is disabled.

Software and firmware are permitted to use UCIe DVSEC Link Capability and Control registers to determine within which configuration to train the link.

Figure 4-48. Implementation Example Showing Two Different Operating Modes of the Same Hardware Implementation



Another example of interoperation between UCIe-A x64 and UCIe-A x32 is when the UCIe-A x64 Stack degrades bandwidth (Degraded Width Mode) to match the remote Link partner's maximum throughput. [Figure 4-49](#) shows an example of RDI byte-to-module assignments for a four-module set of x64 Advanced Package modules interoperating with a four-module set of x32 Advanced Package modules. The example is for a 256B RDI width on the x64 set, and a 128B RDI on the x32 set. On the Transmitter side of x64 modules, the MMPL throttles RDI, as required, because the MMPL can only send half the bytes over 8 UI; and on the Receiver side, the MMPL accumulates 16 UI worth of data before forwarding it over RDI (assumes data transfers are in chunks of 256B OR appropriate pause of data stream indications are applied and detected by MMPLs/Adapters within the data stream).

Figure 4-49. RDI Byte-to-Module Assignment Example for x64 Interop with x32

	M1	M0	M2	M3
UI 8 - 15	RDI B160 --- RDI B191 RDI B224 --- RDI B255 RDI B128 --- RDI B159 RDI B192 --- RDI B223 RDI B192 --- RDI B159	RDI B192 --- RDI B223 RDI B128 --- RDI B159 RDI B64 --- RDI B95 RDI B64 --- RDI B95	RDI B192 --- RDI B223 RDI B128 --- RDI B159 RDI B64 --- RDI B95 RDI B64 --- RDI B95	RDI B224 --- RDI B255 RDI B160 --- RDI B191 RDI B96 --- RDI B127 RDI B32 --- RDI B63
UI 0 - 7	RDI B32 --- RDI B63 RDI B96 --- RDI B127 Rx (x32)	RDI B0 --- RDI B31 RDI B64 --- RDI B95 Rx (x32)	RDI B64 --- RDI B95 RDI B0 --- RDI B31 Rx (x32)	RDI B96 --- RDI B127 RDI B32 --- RDI B63 Tx (x32)
	Tx (x64)	Rx(x64)	Tx (x64)	Rx(x64)
UI 0 - 7	RDI B32 --- RDI B63 RDI B96 --- RDI B127 RDI B0 --- RDI B31 RDI B64 --- RDI B95 RDI B64 --- RDI B95	RDI B160 --- RDI B191 RDI B224 --- RDI B255 RDI B128 --- RDI B159 RDI B192 --- RDI B223 RDI B192 --- RDI B223	RDI B192 --- RDI B223 RDI B128 --- RDI B159 RDI B64 --- RDI B95 RDI B64 --- RDI B95	RDI B224 --- RDI B255 RDI B160 --- RDI B191 RDI B96 --- RDI B127 RDI B32 --- RDI B63
UI 8 - 15	RDI B160 --- RDI B191 RDI B224 --- RDI B255 RDI B128 --- RDI B159 RDI B192 --- RDI B223 RDI B192 --- RDI B159	M3	M2	M0
				M1

For the example shown in [Figure 4-49](#), a single Adapter is operating with all four Modules. The D2D stack uses MMPL with 4 modules, with each of the x64 modules operating in Degraded Width Mode, and only 32 lanes routed per module. The corresponding values in the capability register and Link Training parameter are as listed in [Table 4-16](#).

Table 4-16. Capability Register and Link Training Parameter Values for RDI Byte-to-Module Assignment Example for x64 Interop with x32

UCIE Link DVSEC Capability Register	Max Link Width	x128
	APMW	1
Link Training Parameter	UCIE-A x32	1

See [Section 5.7.2.4](#) for comprehensive rules of interoperation between x64 and x32 Advanced Package modules.

4.8 Sideband PHY Arbitration between MPMs and Link Management Packets

While the exact arbitration policy is implementation specific, care should be taken to avoid delaying transmitting any pending link management packets for extended lengths of time potentially causing timeouts. See [Section 8.2.5.1.2](#) for length restriction on MPMs with Data that allows for PHY arbitration to provide an upper bound on the amount of delay to send a link management packet or a higher-priority MPM packet that might be waiting behind an MPM with Data packet. Additionally, PHY transmitter must fully complete transmission of a MPM with Data within 512 UI max (when Sideband Performant Mode Operation is negotiated) and 768 UI max (when Sideband Performant Mode

Operation is not negotiated). See [Section 4.1.5](#) for details of sideband transmission UI notation. See [Section 4.1.5.1](#) for Performant Mode Operation (PMO) details.

[Figure 4-50](#) and [Figure 4-51](#) show the arbitration at the PHY.

Figure 4-50. Example of Encapsulated MTPs Transmitted on Sideband Link without Sideband PMO

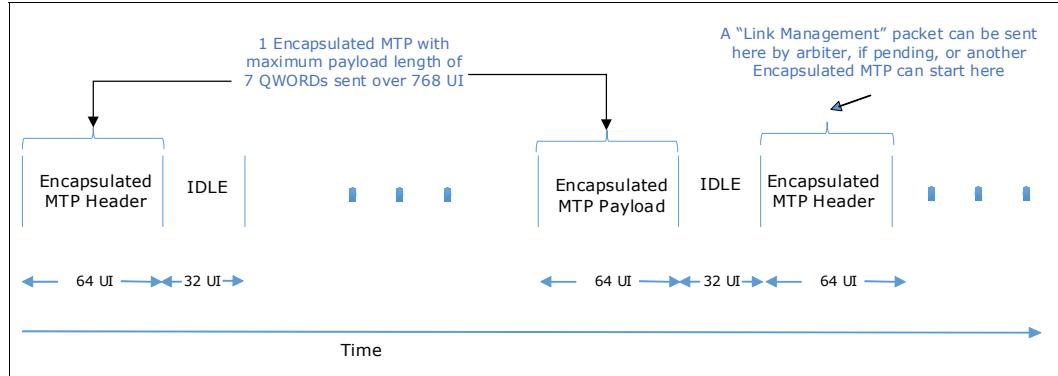
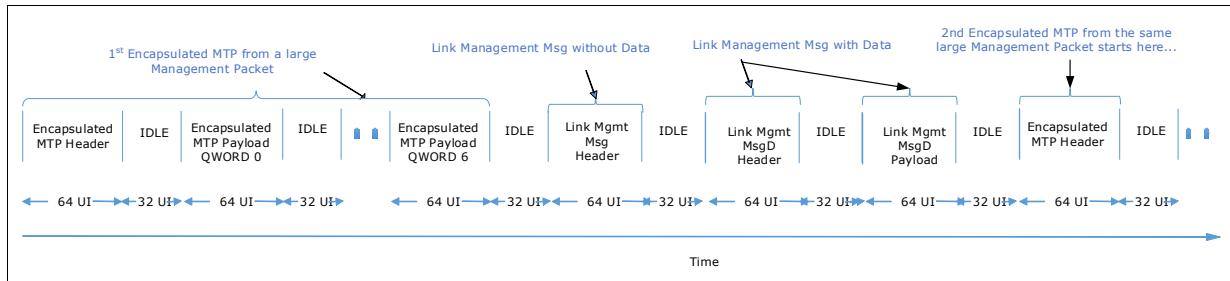


Figure 4-51. Example of a Large Management Packet Split into Two Encapsulated MTPs, with No Segmentation, No Sideband PMO, and with Two Link Management Packets between the Two Encapsulated MTPs



§ §

5.0 Electrical Layer (2D and 2.5D)

Key attributes of electrical specification include:

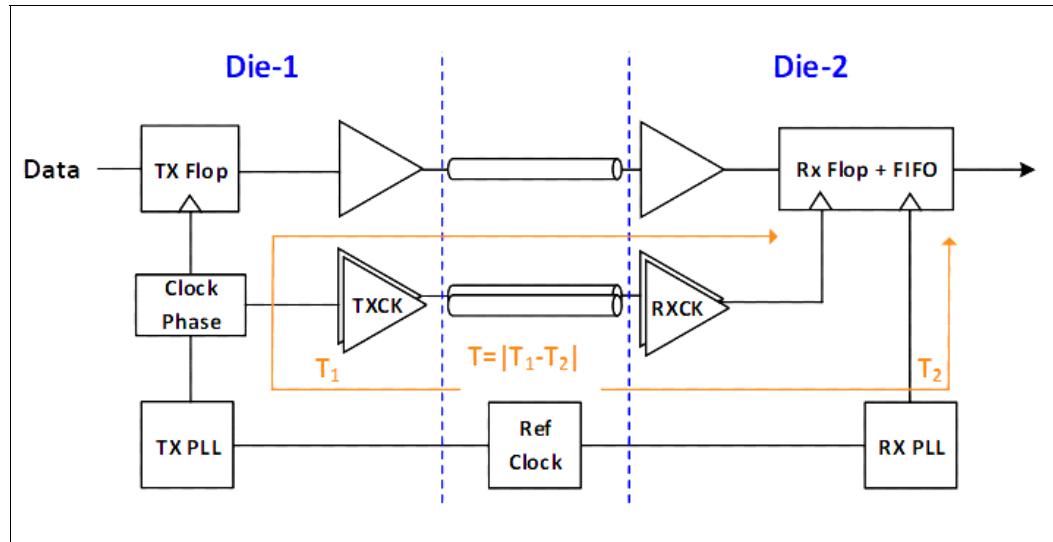
- Support for 4, 8, 12, 16, 24, and 32 GT/s data rates
- Support for Advanced and Standard package interconnects
- Support for clock and power gating mechanisms
- Single-ended unidirectional data signaling
- DC coupled point-to-point interconnect
- Forwarded clock for transmit jitter tracking
- Matched length interconnect design within a module
- Tx driver strength control and unterminated Rx for Advanced Package
- Tx termination and data rate and channel reach dependent Rx termination for Standard Package

5.1 Interoperability

5.1.1 Data rates

A device must support 4 GT/s and all the data rates data rates between 4 GT/s and the highest supported data rate. For example, a device supporting 16 GT/s must also support 4, 8, and 12 GT/s Data rates.

Spread-Spectrum Clocking (SSC) is permitted. Common reference clock (REFCLK) is required between a UCIe Link Transmitter and the corresponding UCIe Link partner's Receiver with a transport delay difference less than 5 ns to limit the FIFO depth and minimize the latency impact. For the retimer use case, the "Local UCIe Link connection" shall use common REFCLK, while the "Off-Package Link connection" is not required to use or share the common REFCLK. [Figure 5-1](#) shows the transport delay difference and is symmetrical for both directions of a Die's UCIe Link connection. The transport delay represents the delay difference between the Transmitter data to the Receiver data latch and the clock as seen at the receiver's FIFO output data latch. See [Section 5.1.2](#) for REFCLK details.

Figure 5-1. Example Common Reference Clock**IMPLEMENTATION NOTE**

In typical implementations, the LCLK for PCIe Link Transmitter and LCLK for the corresponding link partner Receiver, are both generated from the common reference clock. In the example implementation of Figure 5-1, the LCLK for Transmitter in Die-1 can be generated from TX PLL and the LCLK for Receiver in Die-2 can be generated from the RX PLL.

5.1.2 Reference Clock (REFCLK)

Common reference clock (REFCLK) uses a single source that is distributed to both the Transmitter and the Receiver. The clock can be supplied from a package pin or be forwarded by another die on the package. In either case, the reference clock used by both dies on the same link must be from the same clock source. Although other reference clocks are possible, it is recommended that every chiplet use a 100-MHz reference clock, including both dies having different reference clock values from the same clock source. Table 5-1 lists the permitted reference clock frequency range. The minimum and maximum frequencies listed in the table indicate the limits, and do not indicate a requirement to support the entire frequency range. It is required for implementations to generate precise I/O clock frequencies for the supported data rates that use the reference clock. Note that this is possible if the I/O clock frequency is an exact integer multiple of the reference clock frequency (if different from 100 MHz). The reference clock may be disabled in low-power states (such as is done in other Standards and Specifications).

Table 5-1. REFCLK Frequency PPMs and SSC PPMs (Sheet 1 of 2)

Symbol	Description	Limits			Unit	Notes
		Min	Rec	Max		
F _{REFCLK}	REFCLK Frequency	25	100	200	MHz	
F _{REFCLKDEVIATION}	REFCLK Frequency Deviation	-300		300	ppm	Maximum deviation allowed from ideal target frequency.
F _{SSC}	SSC Modulation Frequency	30		33	kHz	

Table 5-1. REFCLK Frequency PPMs and SSC PPMs (Sheet 2 of 2)

Symbol	Description	Limits			Unit	Notes
		Min	Rec	Max		
T _{SSC-FREQ-DEVIATION}	SSC Deviation	-0.5		0	%	Tracks for different frequencies.
T _{TRANSPORT-DELAY}	Tx-to-Rx Transport Delay			5	ns	
T _{SSC-MAX-FREQ-SLEW}	SSC df/dt			1250	ppm/us	

5.2 Overview

5.2.1 Interface Overview

High-level block diagrams of UCIe PHY are shown in [Figure 5-2](#) and [Figure 5-3](#). The UCIe physical interface consists of building blocks called Modules. A Module that uses advanced packaging technology (e.g., EMIB, CoWoS) called “Advanced Package Module” consists of a pair of clocks, 64 or 32 single-ended data Lanes for x64 or x32 Advanced Package Module, respectively, a data valid Lane each direction (transmit and receive) and a Track Lane. There is a low-speed sideband bus for initialization, Link training, and configuration reads/writes. The sideband consists of a single-ended sideband data Lane and single-ended sideband clock Lane in both directions (transmit and receive).

The x16 or x8 “Standard Package Module” uses a traditional Standard packaging with larger pitch. A Standard Package Module consists of a pair of clocks, 16 or 8 single-ended data Lanes, a data valid Lane and Track Lane in each direction (transmit and receive). There is a low-speed sideband bus for initialization, Link training, and configuration reads/writes. The sideband consists of a single-ended sideband data Lane and single-ended sideband clock Lane in both directions (transmit and receive).

For some applications, multiple modules (2 or 4) can be aggregated to deliver additional bandwidth.

To avoid reliability issues, it is recommended to limit the Transmitter output high (V_{OH}) to a maximum of 100 mV above the receiving chiplet’s Receiver front-end circuit power supply rail. An over-stress protection circuit may be implemented in the Receiver when V_{OH} is more than 100 mV above the Receiver power supply rail.

Figure 5-2. x64 or x32 Advanced Package Module

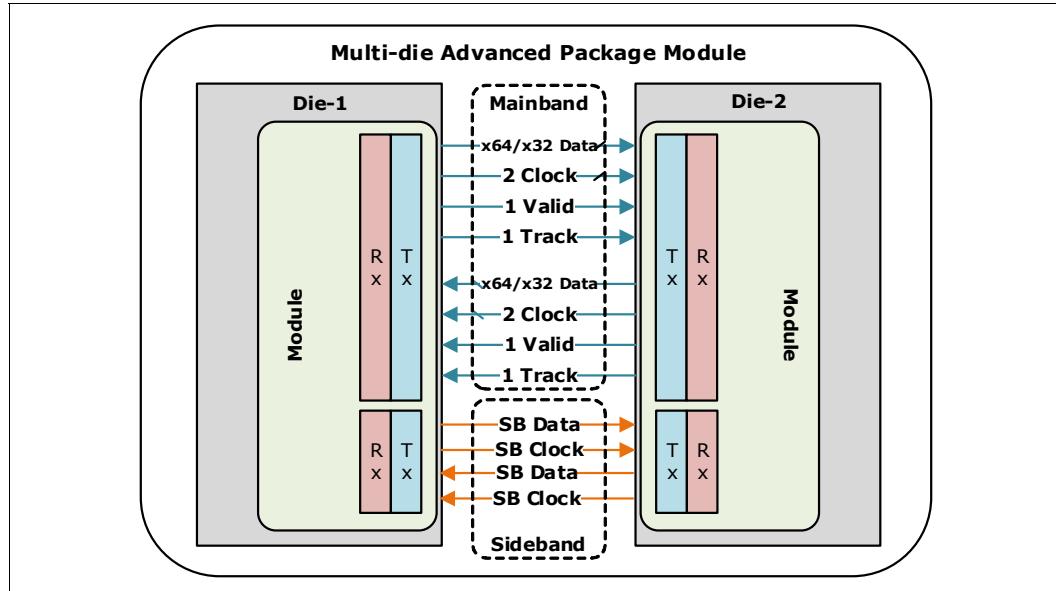
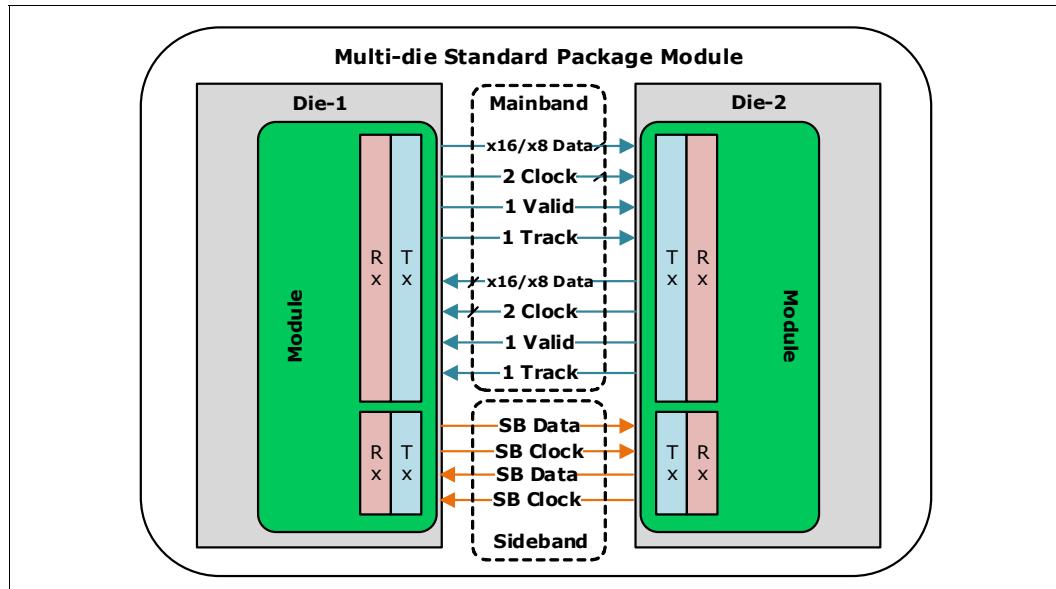


Figure 5-3. x16 or x8 Standard Package Module



5.2.2 Electrical summary

Table 5-2 defines the PHY electrical characteristics of a UCIe device.

Table 5-2. Electrical summary

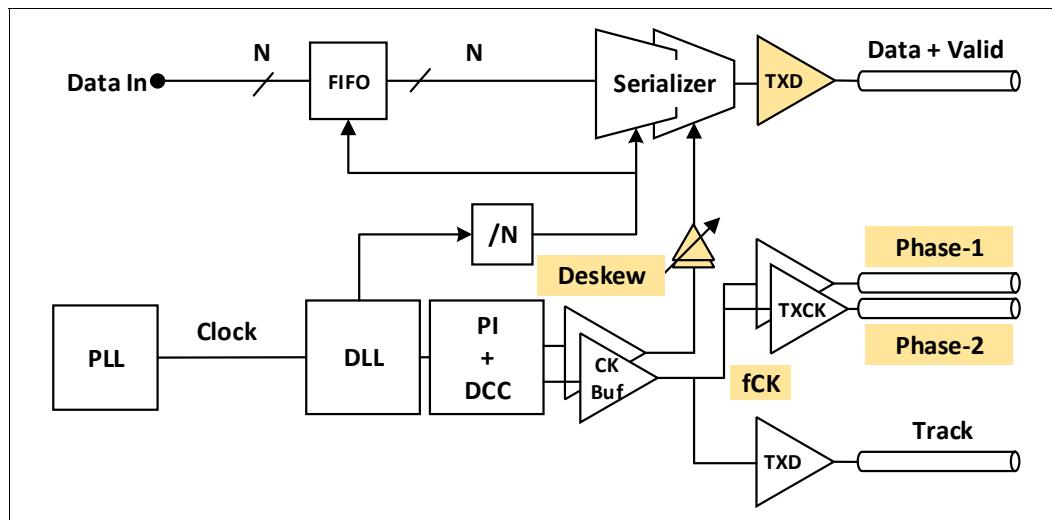
Parameter	Advanced Package (x64)			Standard Package			
Data Width (per module)	64	64	64	16	16	16	16
Data Rate (GT/s)	4/8/12	16	24/32	4-16	4/8/12	16	24/32
Power Efficiency Target (pJ/b)	See Table 1-4						
Latency Target (TX+RX) (UI) ^a (Target upper bound)	12	12	16	12	12	12	16
Idle Exit/Entry Latency (ns) (target upper bound)	0.5	1	1	0.5	0.5	1	1
Idle Power (% of peak power) (target upper bound)	15	15	15	15	15	15	15
Channel Reach (mm)	2	2	2	2-10	25	25	25
Die Edge Bandwidth Density (GB/s/mm) ^b	See Table 1-4						
Bandwidth area density (GB/s/mm ²)	158/316/473	631	710/947	21-85	21/42/64	85	109/145
PHY dimension width per module (um) ^c	388.8	388.8	388.8	571.5 ^d	571.5 ^d	571.5 ^d	571.5 ^d
PHY dimension Depth (um) ^e	1043	1043		1320	1320	1320	1540
ESD ^f	30V CDM (Anticipating going to 5-10V in Future.)						

- a. Electrical PHY latency target. For overall latency target, see Table 1-4.
- b. See Table 1-4.
- c. For compatibility, PHY dimension width must match spec for Advanced Package. Tolerance of PHY dimension width for Standard Package can be higher because there is more routing flexibility. For best channel performance, it's recommended for width to be close to spec.
- d. Standard Package PHY dimension width is the effective width of one (x16) module based on x32 interface (see Figure 5-42 and Figure 5-43).
- e. PHY dimension depth is an informative parameter and depends on bump pitch. Number in the table is based on 45-um bump pitch for 10-column x64 Advanced Package and 100-um bump pitch for Standard Package. See Section 5.7.2 for informative values of PHY dimension depth for combinations of the x64 and x32 Advanced Package modules in 10-column, 16-column, and 8-column bump matrix construction.
- f. Reference (Industry Council on ESD Target Levels): White Paper 2: A Case for Lowering Component-level CDM ESD Specifications and Requirements.

5.3 Transmitter Specification

The Transmitter topology is shown in Figure 5-4. Each data module consists of N single-ended data Transmitters plus a Valid signal. N is 68 (64 Data + 4 Redundant Data) for a x64 Advanced Package Module. N is 34 (32 Data + 2 Redundant Data) for a x32 Advanced Package Module. N is 16 for a x16 Standard Package Module. N is 8 for a x8 Standard Package Module. There is a pair of Transmitters for clocking and a Track signal in each module. The clock rates and phases are discussed in detail in Section 5.5.

Figure 5-4. Transmitter



The Valid signal is used to gate the clock distribution to all data Lanes to enable fast idle exit and entry. The signal also serves the purpose of Valid framing, see [Section 4.1.2](#) for details. The Transmitter implementation for Valid signal is expected to be the same as for regular Data.

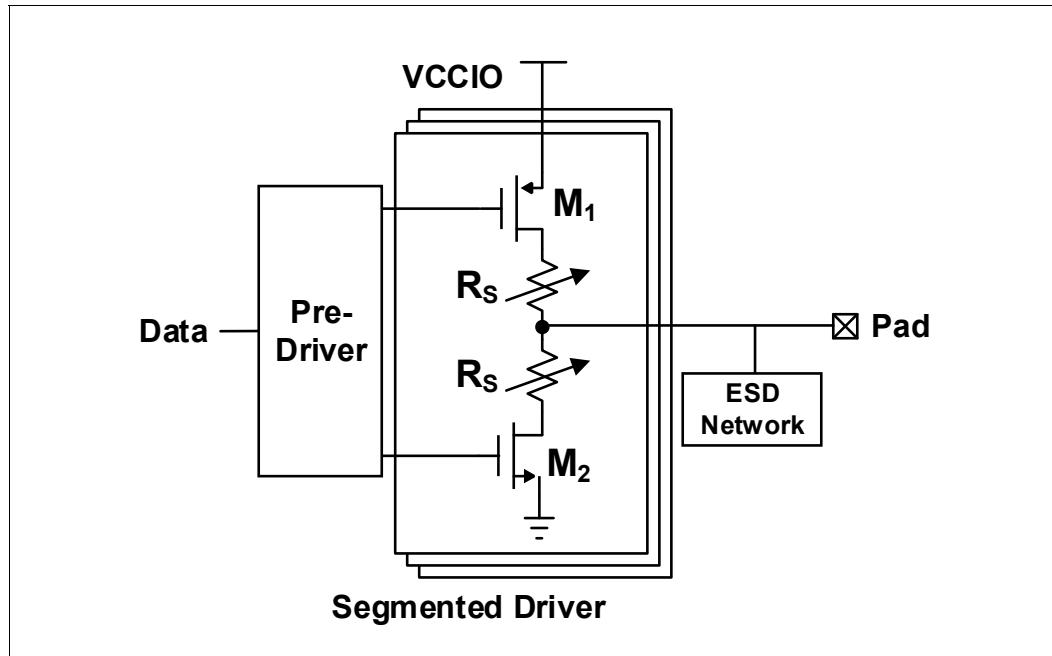
The Track signal can be used for PHY to compensate for slow-changing variables such as voltage or temperature. Track is a unidirectional signal similar to a data bit. The UCIE Module sends a clock pattern (1010...) aligned with Phase-1 of the forwarded clock signal on its Track Transmitter when requested over the sideband by the UCIE Module Partner for its Track Receiver. See [Section 4.6](#) for more details on Runtime Recalibration steps and [Section 5.5.1](#) for Track usage.

5.3.1 Driver Topology

The Transmitter is optimized for simplicity and low power operation. An example of a low power Transmitter driver is shown in Figure 5-5. Separate pull-up and pull-down network strengths are permitted to achieve optimal performance across different channel configurations.

A control loop or training is recommended to adjust output impedance to compensate for the process, voltage and temperature variations. Control loop and training are implementation specific and beyond the scope of this specification. In low power states, the implementation must be capable of tri-stating the output.

It is recommended to optimize the ESD network to minimize pad capacitance. Inductive peaking technique such as T-coil may be needed at higher data rates.

Figure 5-5. Transmitter driver example circuit

5.3.2 Transmitter Electrical parameters

Table 5-3 defines the Transmitter electrical parameters.

Table 5-3. Transmitter Electrical Parameters (Sheet 1 of 2)

Parameter	Min	Typ	Max	Unit
Data Lane TX Swing ^a	0.4			V
Fwd Clock Tx Swing (single ended)	0.4			V
Incoming Clock Rise/Fall time ^b	0.1	0.22	0.25	UI
Incoming Differential Clock Overlap ^b	-	-	30	mUI
Incoming Data Rise/Fall time ^b	-	0.35	-	UI
Driver Pull-up/Pull-down Impedance for Advanced Package ^c	22	25	28	Ohms
Impedance Step Size for Advanced Package ^d			0.5	Ohms
Driver Pull-up/Pull-down Impedance for Standard Package ^e	27	30	33	Ohms
Impedance Step Size for Standard Package ^d	-	-	0.5	Ohms
1-UI Total Jitter ^f	-	-	96/113	mUI pk-pk
1-UI Deterministic Jitter (Dual Dirac) ^g	-	-	48	mUI pk-pk
Tx Data/clock Differential Jitter (Divergent Path) ^h			60	mUI pk-pk
Duty Cycle Error ⁱ	-0.02	-	0.02	UI
Lane-to-Lane Skew Correction Range (up to 16 GT/s) ^j	-0.1	-	0.1	UI
Lane-to-Lane Skew Correction Range (up to 32 GT/s) ^j	-0.15	-	0.15	UI
Lane-to-Lane Skew Correction Range (up to 16 GT/s) ^k	-0.14	-	0.14	UI
Lane-to-Lane Skew Correction Range (up to 32 GT/s) ^k	-0.22	-	0.22	UI
Lane-to-Lane Skew ⁱ	-0.02	-	0.02	UI

Table 5-3. Transmitter Electrical Parameters (Sheet 2 of 2)

Parameter	Min	Typ	Max	Unit
Clock to Mean Data Training Accuracy ^l	-0.07	-	0.07	UI
Phase Adjustment Step ^m	-	-	16	mUI
TX Pad Capacitance (for all speeds) ⁿ	-	-	250	fF
TX Pad Capacitance (8 GT/s capable design) ¹⁴	-	-	300	fF
TX Pad Capacitance (16 GT/s capable design) ^o	-	-	200	fF
TX Pad Capacitance (32 GT/s capable design) ^o	-	-	125	fF

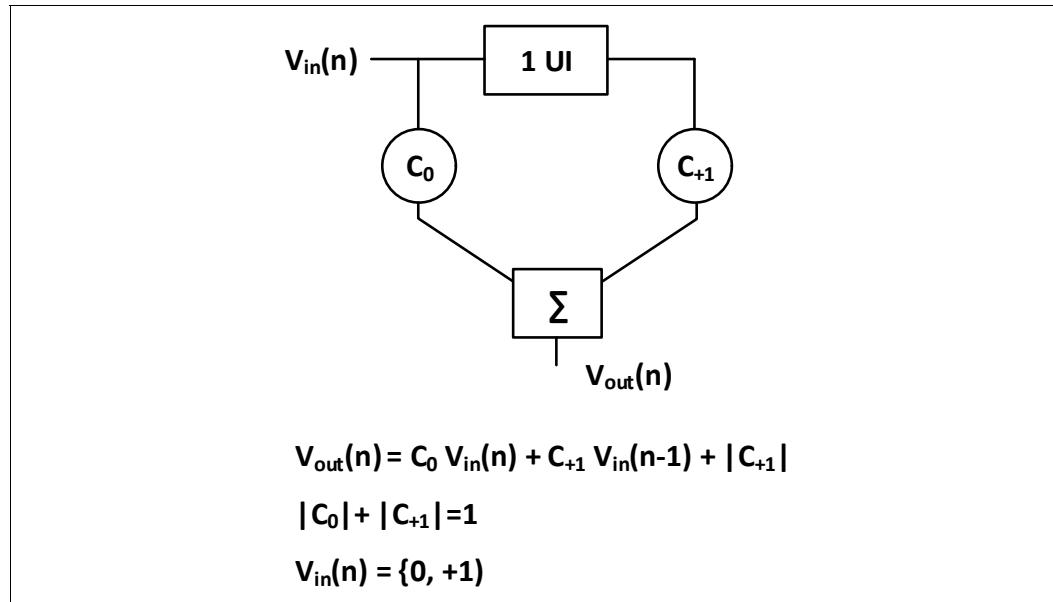
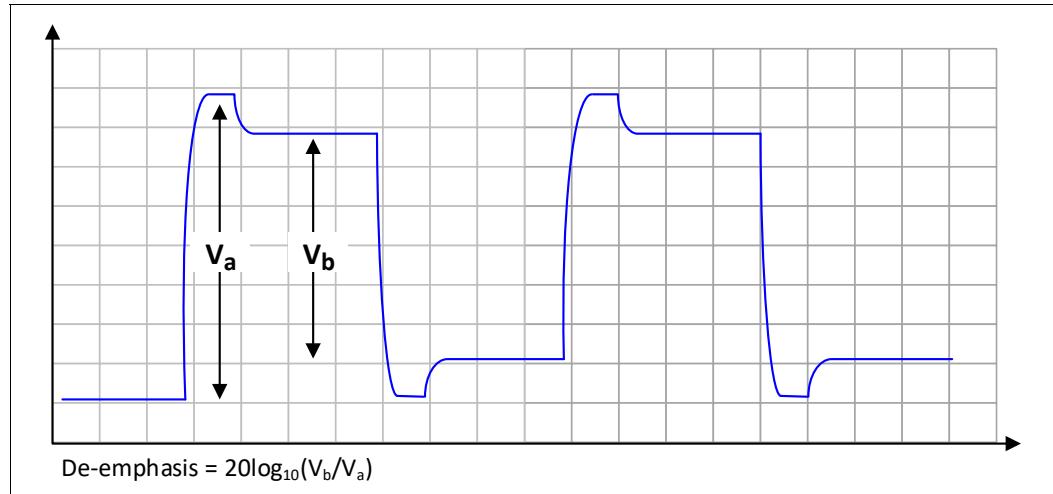
- a. For recommended maximum Transmitter voltage, see [Section 1.5](#).
- b. Expected input (informative). Measured 20% to 80%. Differential clock overlap is deviation from the ideal differential phase (180 degrees apart).
- c. Driver pull-up/down impedance is calibrated at midpoint of Transmitter signal swing.
- d. Impedance step size is an informative parameter and can be implementation specific to meet Driver pull-up/pull-down impedance.
- e. Driver pull-up/pull-down impedance is calibrated at midpoint of Transmitter signal swing (with nominal Rx termination when applicable).
- f. At BER 1E-15/1E-27.
- g. Data dependent jitter excluding Duty Cycle Error.
- h. Includes absolute random jitter and untracked deterministic jitter of the divergent path due to delay mismatch (in the matched architecture).
- i. Post correction.
- j. Advanced Package.
- k. Standard Package.
- l. Includes static and tracking error.
- m. Informative parameter. Phase adjustment step size must be chosen to meet other timing parameters, including Clock-to-Mean Data Training Accuracy, Lane-to-Lane skew, and Duty cycle error (if applicable).
- n. Effective pad capacitance Advanced Package.
- o. Effective pad capacitance Standard Package.

5.3.3 24 GT/s and 32 GT/s Transmitter Equalization

Transmitter equalization is recommended for 16 GT/s and must be supported at 24 GT/s and 32 GT/s data rates to mitigate the channel ISI impact. Tx equalization is de-emphasis only for all applicable Data rates.

Tx equalization coefficients for 24 GT/s and 32 GT/s are based on the FIR filter shown in [Figure 5-6](#). Equalization coefficient is subject to maximum unity swing constraint.

The Transmitter must support the equalization settings shown in [Table 5-4](#). Determination of de-emphasis setting is based on initial configuration or training sequence, where the value with larger eye opening will be selected.

Figure 5-6. Transmitter de-emphasis**Figure 5-7.** Transmitter de-emphasis waveform**Table 5-4.** Transmitter de-emphasis values

Setting	De-emphasis	Accuracy	C_{+1}	V_b/V_a
1	0.0 dB	-	0.000	1.000
2	-2.2 dB	+/- 0.5 dB	-0.112	0.776

5.4 Receiver Specification

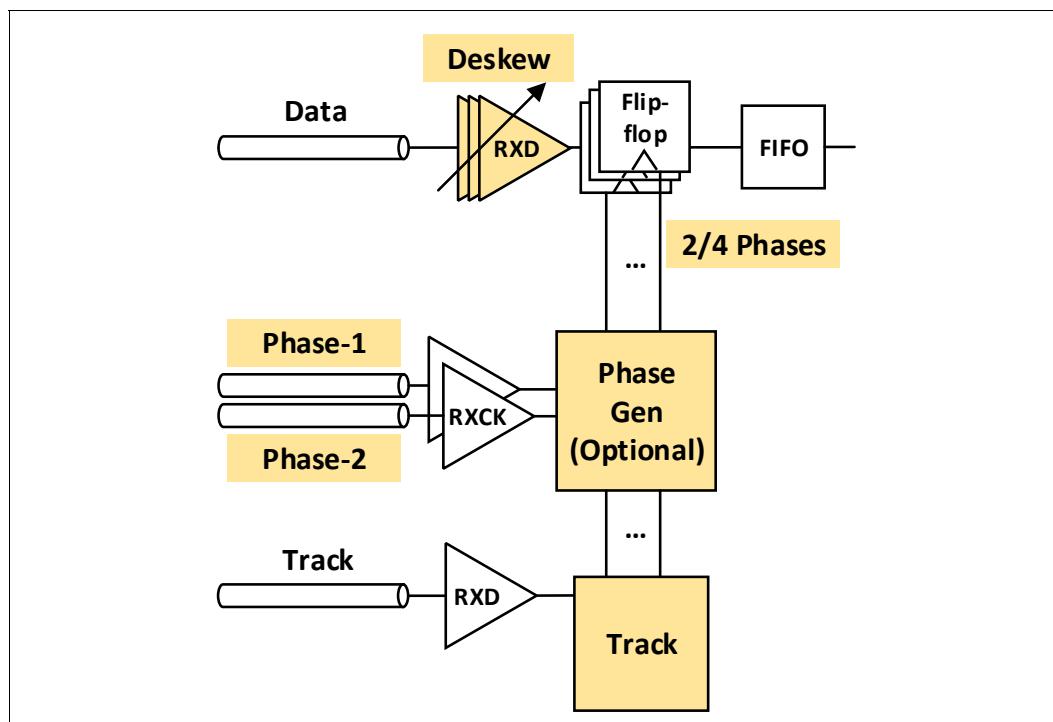
The Receiver topology is illustrated in Figure 5-8. Each Module (Advanced Package and Standard Package) consists of clocks Receivers, data Receivers, and Track Receiver.

The received clock is used to sample the incoming data. The Receiver must match the delays between the clock path and the data/valid path to the sampler. This is to minimize the impact of power supply noise induced jitter. The data Receivers may be implemented as 2-way or 4-way interleaved. For 4-way interleaved implementation the Receiver needs to generate required phases internally from the two phase of the forwarded clock. This may require duty cycle correction capability on the Receiver. The supported forwarded clock frequencies and phases are described in [Section 5.5](#).

At higher data rates, deskew capability may be needed in the receiver to achieve the matching requirements between the data Lanes. Receiver Deskew, when applicable, can be performed during mainband training. More details are provided in [Section 4.5](#).

The UCIe Module, upon requesting the Track signal, receives a clock pattern (1010...) aligned with Phase-1 of the forwarded clock signal on its Track Receiver from the UCIe Module Partner's Track Transmitter and may use the Track signal to track the impact of slow varying voltage and temperature changes on sampling phase.

Figure 5-8. Receiver topology



5.4.1 Receiver Electrical Parameters

The specified Receiver electrical parameters are shown in [Table 5-5](#).

Table 5-5. Receiver Electrical parameters

Parameter	Min	Typ	Max	Unit
RX Input Impedance ^a	45	50	55	Ohms
Impedance Step Size ^a	-	-	1	Ohms
Data/Clock Total Differential Jitter ^{b c}	-	-	60	mUI pk-pk
Lane-to-Lane skew (up to 16 GT/s) ^d	-0.07	-	0.07	UI
Lane-to Lane skew (> 16 GT/s) ^d	-0.12	-	0.12	UI
Phase error ^e (Including Duty cycle error and in-phase quadrature mismatch)	-0.04	-	0.04	UI
Per-Lane deskew adjustment step ^f	-	-	16	mUI
Output Rise Time ^g	-	-	0.1	UI
Output Fall Time ^g	-	-	0.1	UI
RX Pad Capacitance ^h	-	-	200	fF
RX Pad Capacitance (up to 8 GT/s) ^a	-	-	300	fF
RX Pad Capacitance (up to 16 GT/s) ^{a i}	-	-	200	fF
RX Pad Capacitance (24 and 32 GT/s) ^{a i}	-	-	125	fF
Rx Voltage sensitivity	-	-	40	mV

a. Standard Package mode with termination. Impedance step size is an informative parameter and can be implementation specific to meet Rx Input Impedance.

b. Based on matched architecture.

c. Includes absolute random jitter and untracked deterministic jitter of the divergent path due to delay mismatch (in the matched architecture).

d. Require Rx per-Lane deskew if limit is exceeded.

e. Residual error post training and correction.

f. When applicable (informative).

g. Expected output (informative). Measured 20% to 80%.

h. Advanced Package.

i. Effective Pad capacitance.

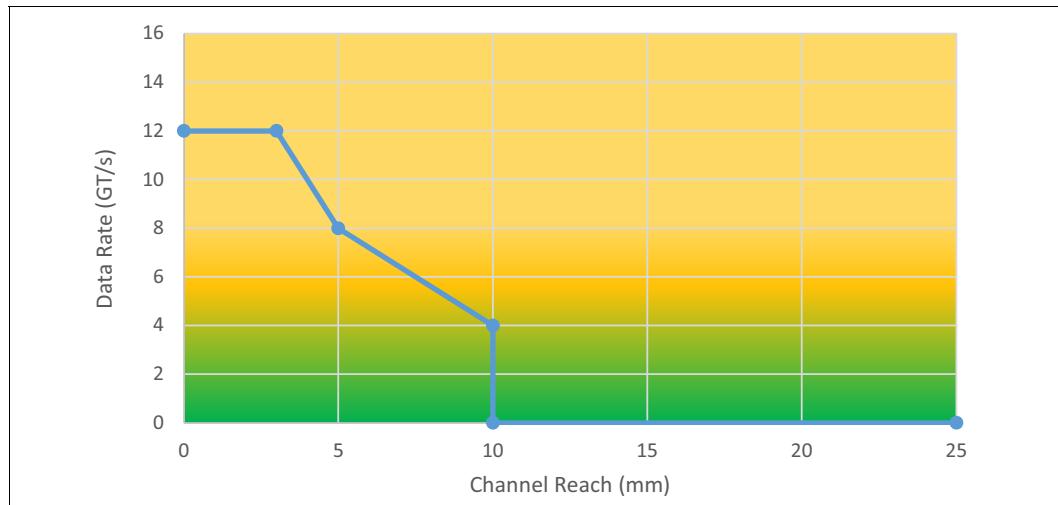
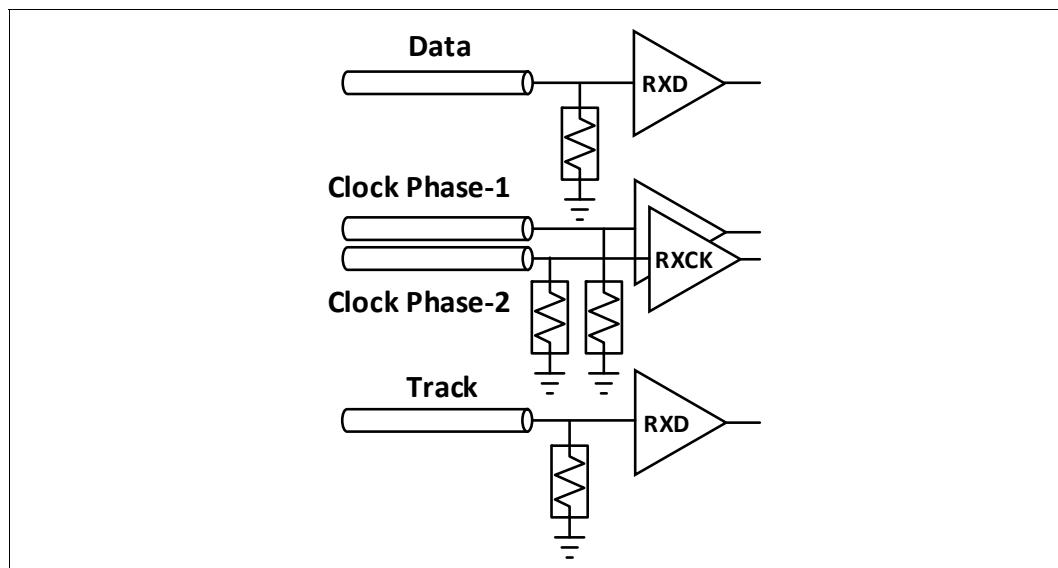
5.4.2 Rx Termination

Rx termination is applicable only to Standard Package modules. All Receivers on Advanced Package modules must be unterminated.

Receiver termination on Standard Package is data rate and channel dependent. [Table 5-6](#) shows the maximum data rate and channel reach combinations for which the Receivers in Standard Package Modules are recommended to remain unterminated for a minimally compliant Transmitter. [Figure 5-9](#) shows an alternate representation of termination requirement. The area below the curve in [Figure 5-9](#) shows the speed and channel-reach combinations for which the Receivers in Standard Package Modules are recommended to remain unterminated. Termination is required for all other combinations. Receivers must be ground-terminated when applicable, as shown in [Figure 5-10](#).

Table 5-6. Maximum channel reach for unterminated Receiver (Tx Swing = 0.4 V)

Data Rate (GT/s)	Channel Reach (mm)
12	3
8	5
4	10

Figure 5-9. Receiver Termination Map for Table 5-6 (Tx Swing = 0.4 V)**Figure 5-10.** Receiver termination

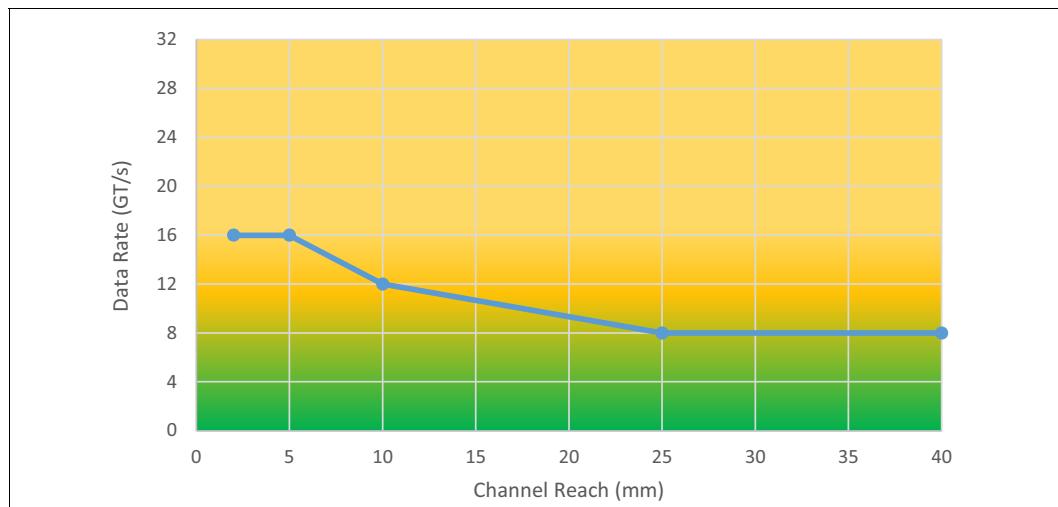
For higher Transmitter swing, unterminated Receiver can be extended to longer channel and high data rate. [Table 5-7](#) shows the maximum data rate and channel reach combinations for Transmitter swing and 0.85 V (maximum recommended swing). [Figure 5-11](#) shows an alternate representation of termination requirement. The area below the curve in [Figure 5-11](#) shows the speed and channel reach

combinations for which the Receivers in Standard Package Modules are recommended to remain unterminated.

Table 5-7. Maximum Channel reach for unterminated Receiver (TX swing = 0.85V)

Data Rate (GT/s)	Channel Reach (mm)
16	5
12	10
8 and below	All supported Lengths

Figure 5-11. Receiver termination map for Table 5-7 (TX Swing = 0.85 V)



IMPLEMENTATION NOTE

When the Transmitter is tri-stated and the Receiver is not required to be enabled (e.g., SBINIT, and some MBINIT states):

- Disabled Receivers must be tolerant of a floating input pad
- Receivers are permitted to enable weak-termination directly on the input pad to prevent crowbar current in the receiver and to lower noise sensitivity at the receiver trip point

When the Transmitter is tri-stated and the Receiver is required to be enabled (e.g., REPAIRCLK and REPAIRVAL states for Advanced Package):

- Enabled Receivers for (CLKP, CLKN, CLKRD, TRK, VLD, VLDRD) must be tolerant of a floating input signal on the pad
- Receivers are permitted to enable weak-termination directly on the input pad to prevent crowbar current in the receiver and to lower noise sensitivity at the receiver trip point

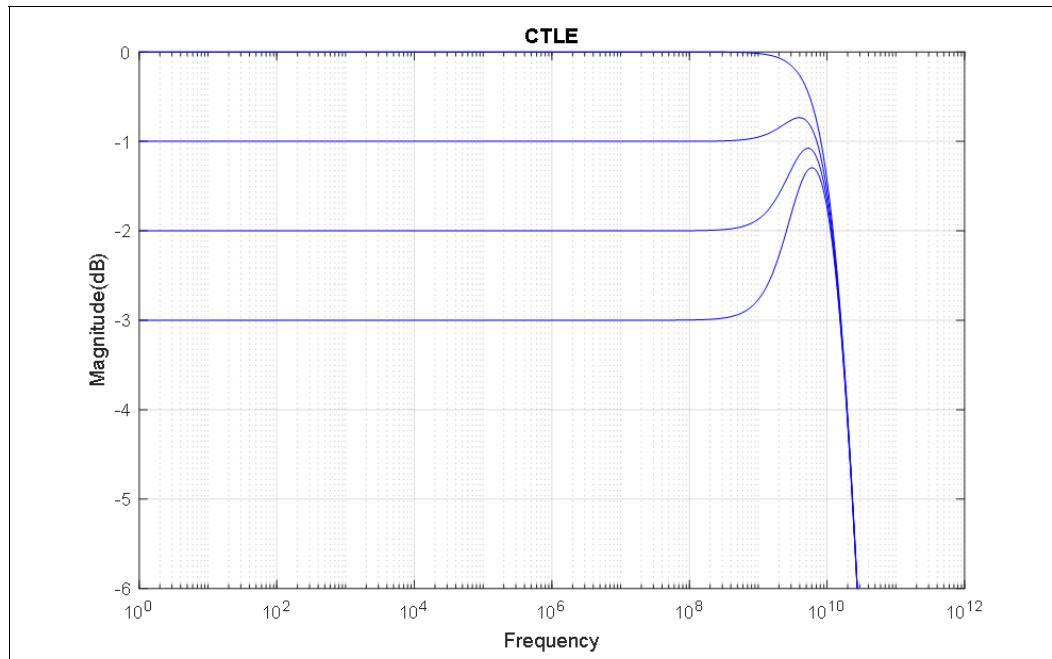
5.4.3 24 and 32 GT/s Receiver Equalization

Receiver equalization may be implemented at 24 GT/s and 32 GT/s data rates. This enables Link operation even when TX equalization is not available. Implementation can be CTLE, inductive peaking, 1-tap DFE, or others. Expected RX equalization capability is equivalent of 1st order CTLE. Example transfer function curves of a first order CTLE are shown in [Figure 5-12](#) and the corresponding equation is shown below:

$$H(s) = \omega_{p2} \left(\frac{s + A_{DC}\omega_{p1}}{(s + \omega_{p1})(s + \omega_{p2})} \right)$$

where, $\omega_{p2} = 2\pi \times \text{DataRate}$, $\omega_{p1} = 2\pi \times \text{DataRate} / 4$, and A_{DC} is the DC gain.

Figure 5-12. Example CTLE



5.5 Clocking

[Figure 5-13](#) shows the forwarded clocking architecture. Each module supports a two-phase forwarded clock. It is critical to maintain matching between all data Lanes and valid signal within the module. The Receiver must provide matched delays between the Receiver clock distribution and Data/Valid Receiver path. This is to minimize the impact of power supply noise-induced jitter on Link performance. Phase adjustment is performed on the Transmitter as shown in [Figure 5-13](#). Link training is required to set the position of phase adjustment to maximize the Link margin.

At higher data rates, Receiver eye margins may be small and any skew between the data Lanes (including Valid) may further degrade Link performance. Per-Lane deskew must be supported on the Transmitter at high data rates.

This specification supports quarter-rate clock frequencies at data rates (24 GT/s and 32 GT/s). The forwarded clock Transmitter must support quadrature phases in addition to differential clock at these data rates (to enable either quarter-rate or half-rate Receiver implementations). [Table 5-8](#) shows the clock frequencies and phases that must be supported at different data rates. Forwarded Clock Phase is negotiated during Link Initialization and Training (see [Section 4.5.3.3.1](#)). At 24 GT/s and 32 GT/s,

Receiver has the options to support differential clock or quadrature clock. The capability register is defined in [Table 9-47](#), and advertised at the beginning of link negotiation. Note that to achieve interoperability with designs of lower max data rate, differential clock must always be used at 16 GT/s and below, independent of the choice at 24 GT/s and 32 GT/s.

5.5.1 Track

Track signal can be used to perform Runtime Recalibration to adjust the Receiver clock path against slow varying voltage, temperature and transistor aging conditions.

When requested by the UCIe Module, the UCIe Module Partner sends a clock pattern (1010...) aligned with Phase-1 of the forwarded clock on its Track Transmitter, as shown in [Figure 5-13](#).

Figure 5-13. Clocking architecture

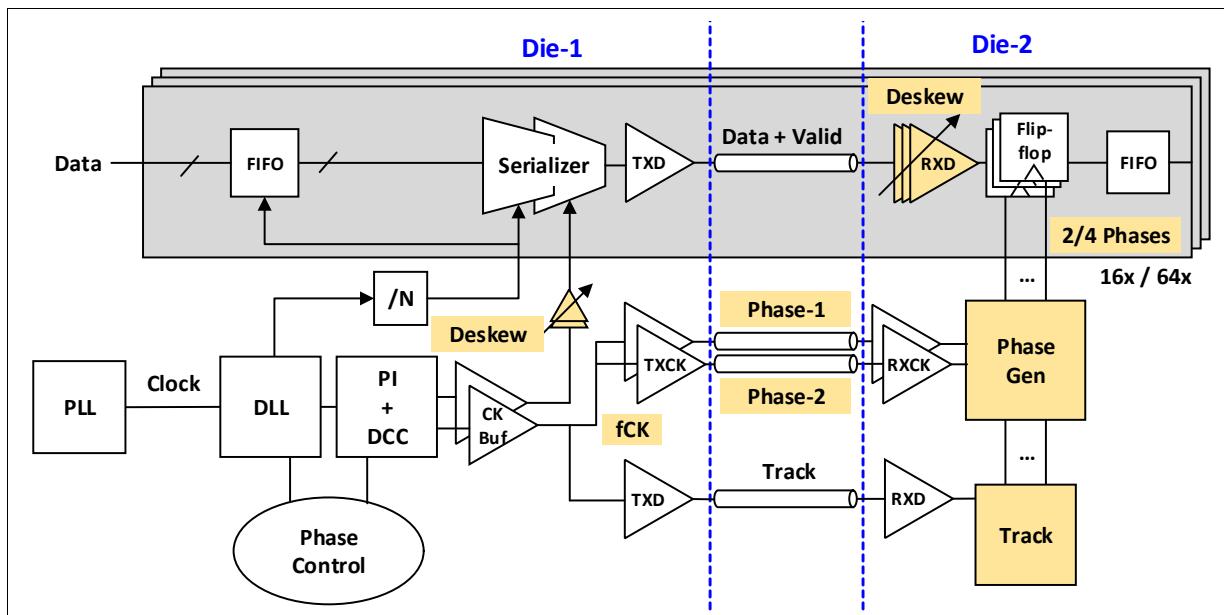


Table 5-8. Forwarded clock frequency and phase

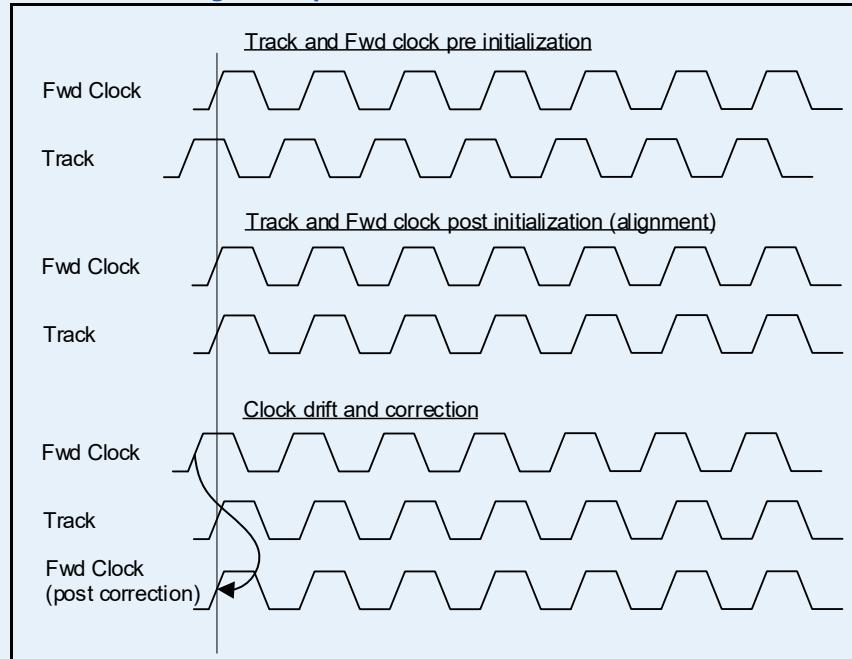
Data rate (GT/s)	Clock freq. (fCK) (GHz)	Phase-1	Phase-2	Deskew (Req/Opt)
32	16	90	270	Required
	8	45	135	Required
24	12	90	270	Required
	6	45	135	Required
16	8	90	270	Required
12	6	90	270	Required
8	4	90	270	Optional
4	2	90	270	Optional

IMPLEMENTATION NOTE

This implementation note provides an example usage for Track signal to calibrate out slow varying temperature- and voltage-related delay drift between Data and Clock on the Receiver.

Track uses the same type of Tx driver and Rx receiver as Data (see [Figure 5-13](#)). A clock pattern aligned with Phase-1 of the forwarded clock is sent from Track Transmitter and received on the Track Receiver. Any initial skew can be calibrated out during initialization and training (MBTRAIN.RXCLKCAL) on the Receiver side. During run-time, any drift between Data and the forwarded clock can be detected. One method for detecting the drift is to sample Track with the forwarded clock. An implementation-specific number of samples can be collected, averaged if needed, and used for drift detection. This drift can then be corrected on the forwarded clock (if needed).

Figure 5-14. Track Usage Example



5.6 Supply noise and clock skew

I/O Vcc noise and the clock skew between data modules shall be within the range specified in Table 5-9.

Table 5-9. I/O Noise and Clock Skew

Parameter	Min	Nom	Max	Unit
I/O Vcc noise for 4 GT/s and 8 GT/s ^a	-	-	80	mVpp
I/O Vcc noise for 12 GT/s ^a	-	-	50	mVpp
I/O Vcc noise for 16 GT/s	-	-	40	mVpp
I/O Vcc noise for 24 GT/s and 32 GT/s ^a	-	-	30	mVpp
Module to module clock skew ^b	-	-	60	ps

- a. I/O VCC noise includes all noise at the I/O supply bumps relative to VSS bumps. This noise includes all DC and AC fluctuations at all applicable frequencies.
- b. Applies only to multi-module instantiations.

IMPLEMENTATION NOTE

Due to different micro bump max current capacity and power delivery requirements, PHY in Advanced Package may have TX providing I/O power supply to RX circuits.

Due to low current draw, sideband supply voltage is strongly recommended to be on an always-on power domain.

5.7 Ball-out and Channel Specification

UCIE interconnect channel needs to meet the requirement of minimum rectangular eye open as specified in [Table 5-10](#) under channel compliance simulation conditions with noiseless and jitter-less behavioral TX and RX models.

Figure 5-15. Example Eye diagram

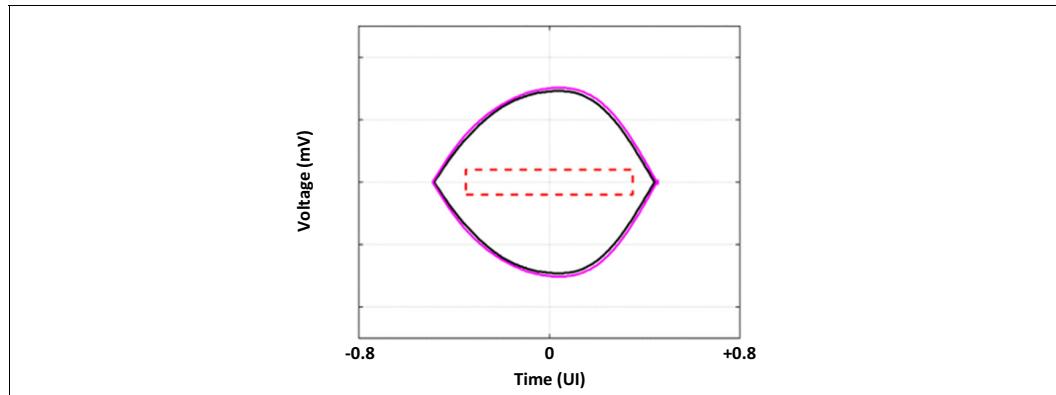


Table 5-10. Eye requirements

Data Rate (GT/s)	Eye Height (mV)	Eye width (UI)
4, 8, 12, 16 ^{a c}	40	0.75
24, 32 ^{a b c}	40	0.65

- a. Rectangular mask.
- b. With equalization enabled.
- c. Based on minimum Tx swing specification.

IMPLEMENTATION NOTE

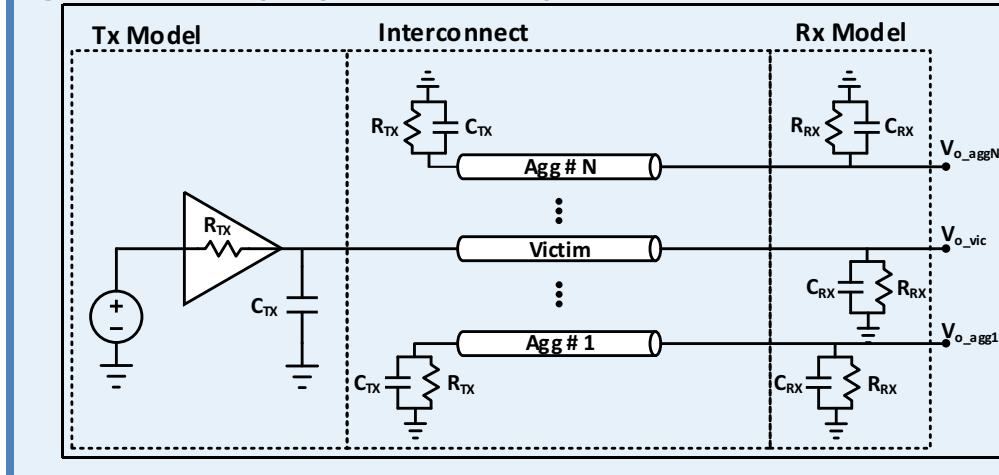
Figure 5-16 shows an example circuit setup that can be used to generate the statistical eye diagram shown in Figure 5-15. RTX is the Transmitter impedance and RRX represents the Receiver termination. CTX, CRX represent effective Transmitter and Receiver capacitance, respectively. For crosstalk, the 19-largest aggressors need to be included. Transmitter equalization (TXEQ) is enabled at 24 GT/s and 32 GT/s.

The eye diagram was generated using a two-step process.

1. Generate ISI and XTALK channel step response using circuit setup shown in Figure 5-16.
2. Use the generated channel response in a signal-integrity or channel-simulation tool to generate a statistical eye diagram (see Figure 5-15)

Other equivalent methods may be used, depending on the signal-integrity or channel-simulation tool.

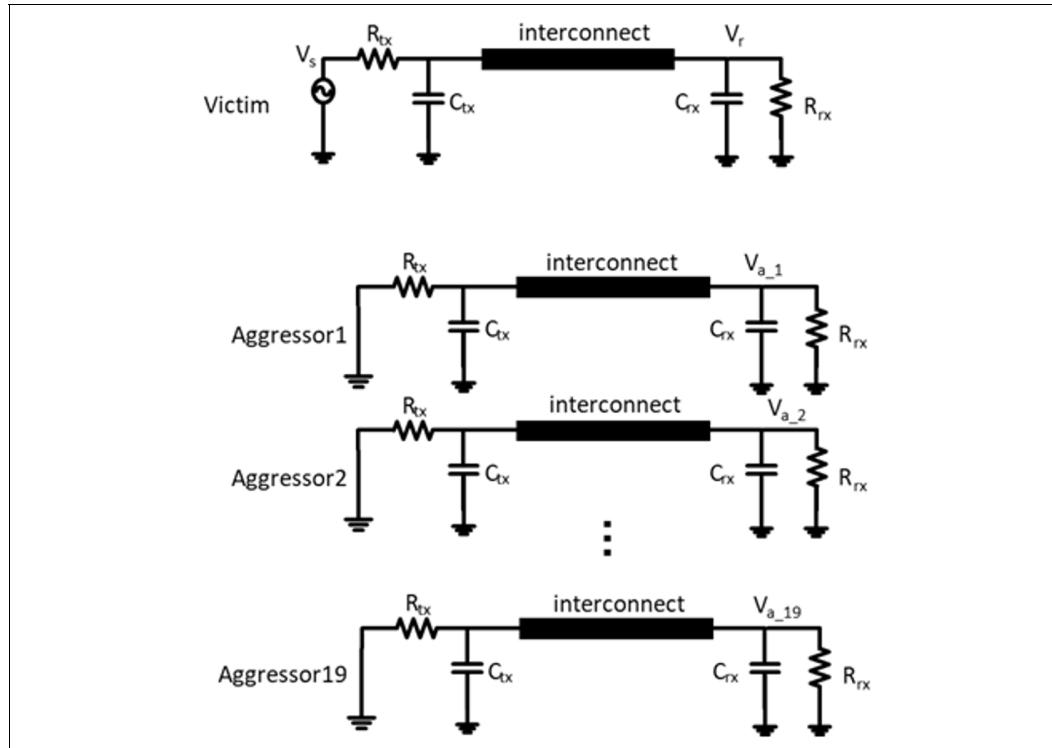
Figure 5-16. Example Eye Simulation Setup



5.7.1 Voltage Transfer Function

Voltage Transfer Function (VTF) based metrics are used to define insertion loss and crosstalk. VTF metrics incorporate both resistive and capacitive components of TX and RX terminations. [Figure 5-17](#) shows the circuit diagram for VTF calculations.

Figure 5-17. Circuit for VTF calculation



VTF loss is defined as the ratio of the Receiver voltage and the Source voltage, as shown in [Equation 5-1](#) and [Equation 5-2](#).

Equation 5-1.

$$L(f) = 20\log_{10}\left|\frac{V_r(f)}{V_s(f)}\right|$$

Equation 5-2.

$$L(0) = 20\log_{10}\left(\frac{R_{rx}}{R_{tx} + R_{channel} + R_{rx}}\right)$$

$L(f)$ is the frequency dependent loss and $L(0)$ is the DC loss. For unterminated channel, $L(0)$ is effectively 0.

VTF crosstalk is defined as the power sum of the ratios of the aggressor Receiver voltage to the source voltage. 19 aggressors are included in the calculation. Based on crosstalk reciprocity, VTF crosstalk can be expressed as shown in [Equation 5-3](#).

Equation 5-3.

$$XT(f) = 10\log_{10} \left(\sum_{i=1}^{19} \left| \frac{V_{ai}(f)}{V_s(f)} \right|^2 \right)$$

5.7.2 Advanced Package**Table 5-11. Channel Characteristics**

Data Rate	4-16 GT/s	24, 32 GT/s
VTF Loss (dB)	$L(f_N) > -3$	$L(f_N) > -5$
VTF Crosstalk (dB) ^a	$XT(f_N) < 1.5 L(f_N) - 21.5$ and $XT(f_N) < -23$	$XT(f_N) < 1.5 L(f_N) - 19$ and $XT(f_N) < -24$

a. Based on Voltage Transfer Function Method (Tx: 25 ohm / 0.25 pF; Rx: 0.2 pF).

f_N is the Nyquist frequency. The equations in the table form a segmented line in the loss-crosstalk coordinate plane, defining the pass/fail region.

Table 5-12. x64 Advanced Package Module Signal List (Sheet 1 of 2)^a

Signal Name	Count	Description
Data		
TXDATA[63:0]	64	Transmit Data
TXVLD	1	Transmit Data Valid; Enables clocking in corresponding module
TXTRK	1	Transmit Track signal
TXCKP	1	Transmit Clock Phase-1
TXCKN	1	Transmit Clock Phase-2
TXCKRD	1	Redundant for Clock and Track Lane repair
TXDATARD[3:0]	4	Redundant for Data Lane repair
TXVLDRD	1	Redundant for Valid
RXDATA[63:0]	64	Receive Data
RXVLD	1	Receive Data Valid; Enables clocking in corresponding module
RXTRK	1	Receive Track.
RXCKP	1	Receive Clock Phase-1
RXCKN	1	Receive Clock Phase-2
RXDATARD[3:0]	4	Redundant for Data Lane repair
RXCKRD	1	Redundant for Clock Lane repair
RXVLDRD	1	Redundant for Valid
Sideband		
TXDATASB	1	Sideband Transmit Data
RXDATASB	1	Sideband Receiver Data
TXCKSB	1	Sideband Transmit Clock
RXCKSB	1	Sideband Receive Clock
TXDATASBRD	1	Redundant Sideband Transmit Data

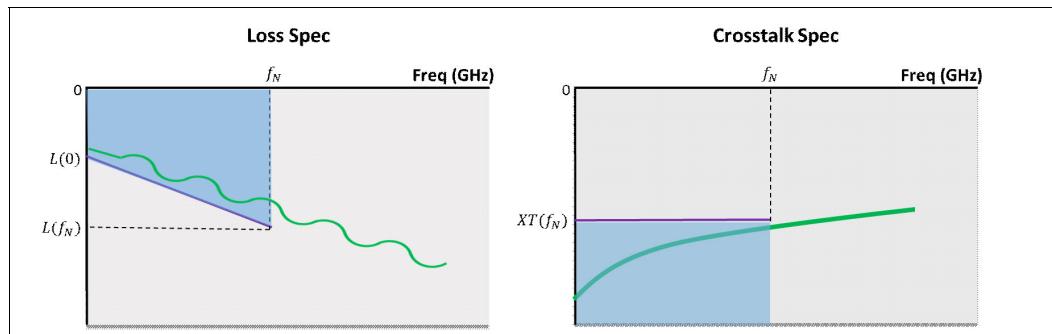
Table 5-12. x64 Advanced Package Module Signal List (Sheet 2 of 2)^a

Signal Name	Count	Description
RXDATASBRD	1	Redundant Sideband Receiver Data
TXCKSBRD	1	Redundant Sideband Transmit Clock
RXCKSBRD	1	Redundant Sideband Receive Clock
Power and Voltage		
VSS		Ground Reference
VCCIO		I/O supply
VCCFWDIO		Forwarded power supply from remote Transmitter supply to local Receiver AFE (see Tightly Coupled mode in Section 5.8)
VCCAON		Always on Aux supply (sideband)

a. For x32 Advanced Package module, the **TXDATA[63:32]**, **TXRD[3:2]**, **RXDATA[63:32]**, and **RXRD[3:2]** signals do not apply. All other signals are the same as the x64 Advanced Package Module signals.

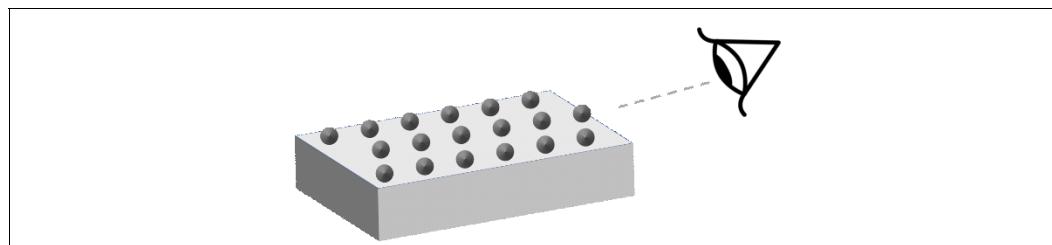
5.7.2.1 Loss and Crosstalk Mask

Loss and crosstalk are specified by a mask defined by the $L(f_N)$ and $XT(f_N)$ at Nyquist frequency. It is a linear mask from DC to f_N for loss and flat mask for crosstalk, illustrated by [Figure 5-18](#). Loss from DC to f_N needs to be above the spec line. Crosstalk from DC to f_N needs to be below the spec line. The green line in [Figure 5-18](#) is a representative passing signal.

Figure 5-18. Loss and Crosstalk Mask

5.7.2.2 x64 Advanced Package Module Bump Map

All bump matrices in this section and hereinafter are defined with “dead bug” view which means the viewer is looking directly at the UCIe micro bumps facing up, with the die flipped like a “dead bug” as illustrated in [Figure 5-19](#).

Figure 5-19. Viewer Orientation Looking at the Defined UCIe Bump Matrix

[Figure 5-20](#), [Figure 5-21](#), and [Figure 5-22](#) show the reference bump matrix for the 10-column, 16-column, and 8-column x64 Advanced Package Modules, respectively. The lower left corner of the bump map will be considered “origin” of a bump matrix and the leftmost column is Column 0.

It is strongly recommended to follow the bump matrices provided in [Figure 5-20](#), [Figure 5-21](#), and [Figure 5-22](#) for x64 Advanced Package interfaces.

The 10-column bump matrix is optimal for bump pitch range of 38 to 50 um. To achieve optimal area scaling with different bump pitches, the optional 16-column and 8-column bump matrices are defined for bump ranges of 25 to 37 um and 51 to 55 um, respectively, which will result in optimal Module depth while maintaining Module width of 388.8 um, as shown in [Figure 5-21](#) and [Figure 5-22](#), respectively.

The following rule must be followed for the 10-column x64 Advanced Package bump matrix:

- The signal order within a column must be preserved. For example, Column 0 must contain the signals: `txdataRD0`, `txdata0`, `txdata1`, `txdata2`, `txdata3`, `txdata4`, ..., `rxdata59`, `rxdata60`, `rxdata61`, `rxdata62`, `rxdata63`, `rxdataRD3`, and `txdatasbRD`. Similarly, 16-column and 8-column x64 Advanced Packages must preserve the signal order within a column of the respective bump matrices.

It is strongly recommended to follow the supply and **ground** pattern shown in the bump matrices. It must be ensured that sufficient supply and **ground** bumps are provided to meet channel characteristics (FEXT and NEXT) and power-delivery requirements.

The following rules must be followed when instantiating multiple modules of Advanced Package bump matrix:

- Modules must be stepped in the same orientation and abutted.
- Horizontal or vertical mirroring is not permitted.
- Module stacking is not permitted.

Additionally, in multi-module instantiations it is strongly recommended to add one column of **vss** bumps on each outside edge of the multi-module instantiation.

Mirror die implementation may necessitate a jog or additional metal layers for proper connectivity.

Figure 5-20. 10-column x64 Advanced Package Bump Map

Column0	Column1	Column2	Column3	Column4	Column5	Column6	Column7	Column8	Column9
VSS		VSS		VCCIO		VCCIO		VSS	
	VSS		VCCIO		VCCIO		VSS		VSS
VSS		VSS		VCCIO		VCCIO		VSS	
	rxcksbRD		rxcksb		VCCIO		rxdataSB		rxdataSBRD
txdataSBRD		txdataSB		VCCIO		txcksB		txcksBRD	
	rxdata50		rxdata35		rxdata29		rxdata14		rxdataRDO
rxdataRD3		rxdata49		rxdata34		rxdata28		rxdata13	
rxdata63		rxdata51		rxdata36		rxdata30		rxdata15	
	rxdata52		VCCIO		rxdata33		VCCIO		rxdata12
VSS			VSS		rxdata31		VSS		rxdata0
	rxdata53		rxdata48		rxdata32		rxdata27		rxdata11
rxdata62		rxdata54		rxdata37		rxdataRD1		rxdata16	
	rxdata55		rxdata46		rxdata38		VSS		rxdata10
rxdata61		rxdata55		rxdata39		rxckRD		rxdata17	
	rxdata56		rxdata45		rxvldRD		rxdata24		rxdata9
rxdata60		rxdata46		VSS		rxckn		rxdata18	
	rxdata57		rxdata44		rxvld		VSS		rxdata8
rxdata59		rxdata43		rxdata40		rxckp		rxdata19	
	rxdata58		rxdata43		rxtrk		rxdata23		rxdata7
VSS		rxdata42		VCCIO		VSS		rxdata20	
VCCIO		VCCFWDIO		VCCFWDIO		VCCFWDIO		rxdata6	
	txdata5		txdata21		VCCIO		txdata41		VCCFWDIO
txdata4		txdata5		txdata22		VSS		txdata42	
	txdata6		txdata20		txckp		txdata40		VSS
VSS		txdata6		txdata23		txckn		txdata43	
	txdata7		txdata19		txdata24		VSS		txdata59
txdata3		txdata7		VSS		txckRD		txdata44	
	txdata8		txdata18		txdata25		txvldRD		txdata56
txdata2		txdata8		txdata26		VCCIO		txdata45	
	txdata9		txdata17		VCCIO		txdata39		txdata60
VCCIO		txdata9		VCCIO		VSS		txdata46	
	txdata10		txdata16		VCCIO		VCCIO		txdata61
txdata1		txdata10		txdata27		txdataRD1		txdata47	
	txdata11		VSS		txdata26		txdata37		txdata62
txdata0		txdata11		VSS		txdata32		txdata48	
	txdata12		VSS		txdata31		VSS		txdata53
VSS		txdata12		VSS		txdata33		txdata49	
	txdata13		VSS		txdata28		VSS		txdata52
txdataRDO		txdata13		VCCIO		txdata34		txdata50	
	VCCIO		VCCIO		VCCIO		VCCIO		VCCIO
VCCIO		VCCIO		VCCIO		VCCIO		VCCIO	
Die Edge									

Note:

In Figure 5-20, at 45-um pitch, the module depth of the 10-column reference bump matrix as shown is approximately 1043 um.

Figure 5-21. 16-column x64 Advanced Package Bump Map

Column0	Column1	Column2	Column3	Column4	Column5	Column6	Column7	Column8	Column9	Column10	Column11	Column12	Column13	Column14	Column15
vss	vss	vss	vccio	vccio	vccio	vss	vccio	vccio	vccio	vss	vss	vss	vss	vss	vss
vss	vss	rxcksbRD	txdatasbRD	rxcksb	rxdata35	vss	rxdata36	rxdata34	rxdataRD1	rxdata30	rxdata28	rxdata13	vss	vss	vss
vss	rxdata54	rxdata50	rxdata55	rxdata49	rxdata51	rxdata48	rxdata37	rxdata32	rxckRD	rxdata31	rxdata27	rxdata15	rxdata10	rxdata9	rxdataRDO
rxdataRD3	rxdata55	rxdata53	rxdata51	rxdata48	rxdata36	rxdata33	vss	rxvldRD	rxdata25	rxdata23	rxdata16	rxdata12	rxdata8	rxdata7	rxdata0
rxdata63	rxdata61	vss	rxdata56	rxdata47	rxdata46	rxdata45	rxdata42	rxdata39	rxckp	rxdata22	rxdata20	rxdata18	rxdata6	rxdata5	rxdata2
vss	rxdata60	rxdata57	rxdata59	rxdata45	rxdata43	rxdata40	rxvld	rxckn	vss	rxdata21	rxdata20	rxdata19	rxdata6	rxdata5	rxdata1
rxdata62	rxdata58	rxdata44	rxdata44	rxdata41	rxdata41	rxdata40	rxtrk	rxtrk	rxdata24	rxdata21	rxdata20	rxdata19	rxdata4	rxdata3	rxdata2
vccfwdio	vccfwdio	vccfwdio	vccfwdio	vccfwdio	vccfwdio	vccfwdio	vccfwdio	vccfwdio	vccfwdio	vccfwdio	vccfwdio	vccfwdio	vccfwdio	vccfwdio	vccfwdio
vss	vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio
vss	txdata5	txdata21	txdata4	txdata19	txdata20	txdata22	txdata24	txdata25	txtrk	txdata39	txdata41	txdata44	vss	vss	vccio
txdata1	txdata6	txdata7	txdata18	txdata17	txdata23	txdata26	txdata20	txdata25	txckp	txdata32	txdata40	txdata42	txdata45	txdata58	txdata62
vss	txdata3	txdata7	txdata18	vss	txdata23	txdata25	txdata23	txdata25	txckn	txdata38	txdata43	txdata46	txdata57	vss	vss
txdata0	txdata2	vss	txdata17	txdata16	txdata16	txdata31	txdata12	txdata27	txckRD	txdata33	txdata37	txdata48	txdata49	txdata56	txdata63
vss	txdata8	txdata10	txdata15	txdata15	txdata15	txdata30	txdata28	txdata30	txdataRD2	txdata34	txdata36	txdata49	txdata51	txdata53	txdataRDO
txdataRDO	txdata9	txdata11	txdata13	txdata14	txdata14	txdata29	txdata29	txdata29	vss	txdata35	txdata35	txdata50	txdata52	txdata54	vss
vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio	vccio
Die Edge															

Note:

In Figure 5-21, at 25-um pitch, the module depth of the 16-column reference bump matrix as shown is approximately 388 um.

Figure 5-22. 8-column x64 Advanced Package Bump Map

Column0	Column1	Column2	Column3	Column4	Column5	Column6	Column7
vss		vccio		vccio		vss	
	vss		vccio		vccio		vss
vss		vccio		vccio		vss	
	rxcksbRD		rxcksb	txcksb	rxdatasb		rxdatasbRD
txdatasbRD		txdatasb		txcksb		txcksbRD	
	rxdata50		rxdata36		rxdata14		rxdataRD0
rxdataRD3		rxdata49		rxdata27		rxdata13	
	rxdata51		rxdata35		rxdata15		rxdata0
vss		rxdata48		vss		rxdata12	
	rxdata52		rxdata34		rxdata16		vss
rxdata63		vss		rxdata28		rxdata11	
	rxdata53		rxdata33		rxdata17		rxdata1
rxdata62		rxdata47		rxdata29		rxdata10	
	vccio		vccio		vccio		vccio
rxdata61		rxdata46		rxdata30		vss	
	vss		vss		rxdata26		rxdata2
rxdata60		rxdata37		rxdata31		rxdata9	
	rxdata54		rxdata32		rxdata25		rxdata3
rxdata59		rxdata38		rxdataRD1		rxdata8	
	rxdata55		rxdataRD2		rxdata24		rxdata4
vss		rxdata39		vss		vss	
	rxdata45		vss		rxdata23		rxdata7
rxdata56		rxdata40		rxckRD		rxdata18	
	vss		rxvldRD		rxdata22		vss
rxdata57		rxdata41		rxckn		rxdata19	
	rxdata44		rxvld		vss		rxdata6
rxdata58		vss		rxckp		rxdata20	
	rxdata43		rxtrk		rxdata21		rxdata5
vss		rxdata42		vss		vss	
	vccfwdio		vccfwdio		vccfwdio		vccfwdio
vccio		vccio		vccio		vccio	
	vss		vss		txdata42		vss
txdata5		txdata21		txtrk		txdata43	
	txdata20		txckp		vss		txdata58
txdata6		vss		txvld		txdata44	
	txdata19		txckn		txdata41		txdata57
vss		txdata22		txvldRD		vss	
	txdata18		txckRD		txdata40		txdata56
txdata7		txdata23		vss		txdata45	
	vss		txdata30		txdata39		vss
txdata4		txdata24		txdataRD2		txdata55	
	txdata8		txdataRD1		txdata38		txdata59
txdata3		txdata25		txdata32		txdata54	
	txdata9		txdata31		txdata37		txdata60
txdata2		txdata26		vss		vss	
	vss		txdata30		txdata46		txdata61
vccio		vccio		vccio		vccio	
	txdata10		txdata29		txdata47		txdata62
txdata1		txdata17		txdata33		txdata53	
	txdata11		txdata28		vss		txdata63
vss		txdata16		txdata34		txdata52	
	txdata12		vss		txdata48		vss
txdata0		txdata15		txdata35		txdata51	
	txdataRD0		txdata27		txdata49		txdataRD3
	vccio		vccio		vccio		vccio
	vccio		vccio		vccio		vccio
					Die Edge		

Note:

In Figure 5-22, at 55-um pitch, the module depth of the 8-column reference bump matrix as shown is approximately 1,585 um.

Figure 5-23 shows the signal exit order for the 10-column x64 Advanced Package bump map.

Figure 5-23. 10-column x64 Advanced Package Bump map: Signal exit order

		Left to Right																
		txdataR00	txdata0	txdata1	txdata2	txdata3	txdata4	txdata5	txdata6	txdata7	txdata8	txdata9	txdata10	txdata11	txdata12	txdata13	Cont...	
Tx Breakout		txdata14	txdata15	txdata16	txdata17	txdata18	txdata19	txdata20	txdata21	txdata22	txdata23	txdata24	txdata25	txdata26	txdata27	txdata28	Cont...	
Cont...		txdata29	txdata30	txdata31	txdataRD1	txckRD	txckn	txrk	txvld	txvldRD	txdataRD2	txdata32	txdata33	txdata34	txdata35	txdata36	txdata37	Cont...
Cont2...		txdata36	txdata37	txdata38	txdata39	txdata40	txdata41	txdata42	txdata43	txdata44	txdata45	txdata46	txdata47	txdata48	txdata49	txdata50	txdata51	Cont3...
Cont3...		txdata51	txdata52	txdata53	txdata54	txdata55	txdata56	txdata57	txdata58	txdata59	txdata60	txdata61	txdata62	txdata63	txdataRD3			
		Left to Right																
		rxdataRD3	rxdata63	rxdata62	rxdata61	rxdata60	rxdata59	rxdata58	rxdata57	rxdata56	rxdata55	rxdata54	rxdata53	rxdata52	rxdata51	rxdata50	rxdata51	Cont...
Rx Breakout		rxdata49	rxdata48	rxdata47	rxdata46	rxdata45	rxdata44	rxdata43	rxdata42	rxdata41	rxdata40	rxdata39	rxdata38	rxdata37	rxdata36	rxdata35	rxdata36	Cont1...
Cont...		rxdata34	rxdata33	rxdata32	rxdataRD2	rxvldRD	rxvld	rxrk	rxckp	rxckn	rxckRD	rxdataRD1	rxdata31	rxdata30	rxdata29	rxdata28	rxdata29	Cont2...
Cont2...		rxdata27	rxdata26	rxdata25	rxdata24	rxdata23	rxdata22	rxdata21	rxdata20	rxdata19	rxdata18	rxdata17	rxdata16	rxdata15	rxdata14	rxdata13	rxdata14	Cont3...
Cont3...		rxdata12	rxdata11	rxdata10	rxdata9	rxdata8	rxdata7	rxdata6	rxdata5	rxdata4	rxdata3	rxdata2	rxdata1	rxdata0	rxdata0	rxdata0	rxdata0	

IMPLEMENTATION NOTE — x64 BUMP MAPS FOR MAX SPEED

Three reference bump maps in Figure 5-20, Figure 5-21, and Figure 5-22 are recommended for different ranges of bump pitch, while PHY implementations have the flexibility to adjust the power and ground bumps to meet channel characteristics and power delivery requirements, which largely depend on the target speed and the advanced packaging technology capabilities.

At higher speeds, the PHY circuits draw larger current through the bumps and require better signal and power integrity of the packaging solution. This typically requires adding power and ground bumps and optimizing the distribution of them, but the implementation also needs to minimize the lane-to-lane length skew and preserve the assignment and relative order of the signals in each column to comply with the bump matrix rules in Section 5.7.2.2.

Table 5-13. Bump Map Options and the Recommended Bump Pitch Range and Max Speed

Bump Map	Bump Pitch (um)	Max Speed (GT/s)
16 column	25-30	12
	31-37	16
10 column	38-44	24
	45-50	32
8 column	51-55	32

This Implementation Note is formulated to provide PHY implementations a set of reference x64 bump maps to encompass the max speed specified. Table 5-13 summarizes the corresponding max speed for these bump map options and their recommended bump pitch ranges.

Bump maps in Figure 5-24, Figure 5-25, and Figure 5-26 are the x64 implementation references for the corresponding max speed with an enhancement of the power and ground bumps. They all comply with the bump matrix rules in Section 5.7.2.2, and they maintain the backward compatibility in terms of signal exit order. These reference examples have been optimized for signal integrity, power integrity, lane-to-lane skew, electro-migration stress and bump area based on most of the advanced packaging technologies in the industry. Please note that technology requirements vary, and it is still required to verify the bump map with the technology provider for actual implementation requirements and performance targets.

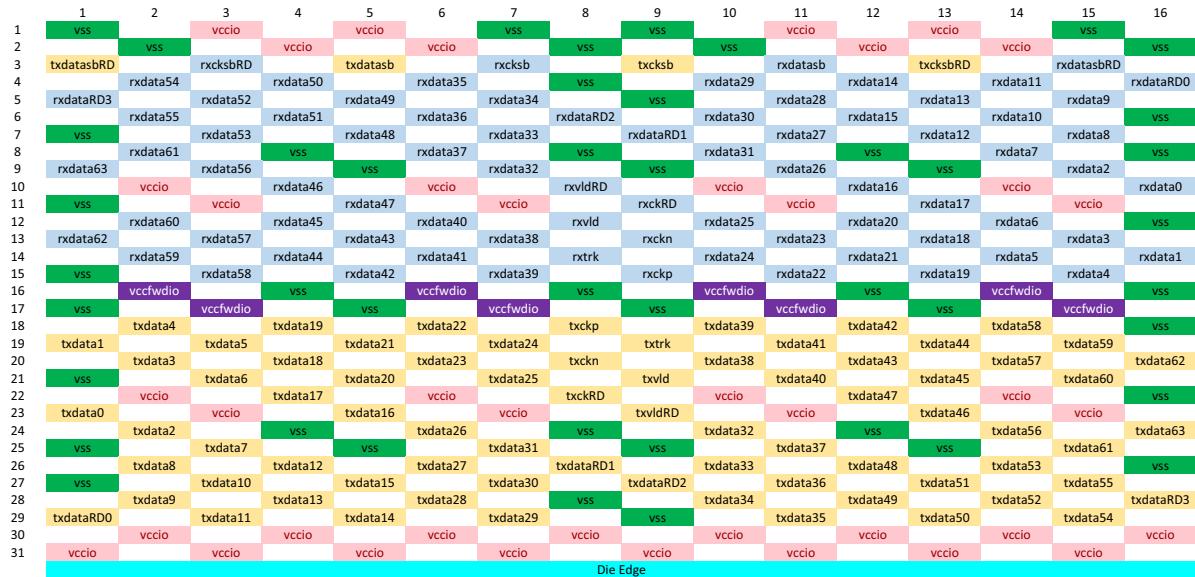
Figure 5-24. 10-column x64 Advanced Package Bump Map Example for 32 GT/s Implementation

	1	2	3	4	5	6	7	8	9	10
1	vss		vss		vccio		vccio		vss	
2		vss		vccio		vccio		vss		vss
3	vss		vss		vccio		vccio		vss	
4		rxcksbRD		rxcksb		vss		rxdatasb		rxdatasbRD
5	txdatasbRD		txdatasb		rxdata35		rxdata29		txcksb	
6		rxdata50		rxdata49		rxdata34		rxdata28		rxdata14
7	rxdataRD3			rxdata51		vccio			rxdata13	
8					vss	rxdata33		vccio		vccio
9	vccio					rxdata36		rxdata30		rxdata12
10				rxdata52			rxdata36		rxdata15	
11	vss				rxdata48		vss		rxdata27	
12				rxdata53		rxdata37		rxdata31		rxdata16
13	rxdata63				rxdata47		rxdata32		rxdata26	
14		vccio			vccio		vccio		vccio	
15	rxdata62			rxdata46		rxdataRD2		rxdata25		rxdata9
16					rxdata38		rxdataRD1		rxdata17	
17	vss			vss		vss		vss		rxdata1
18				rxdata55		rxdata39		vccio		rxdata18
19	rxdata61			rxdata45		rxvldRD		rxdata24		rxdata8
20		vccio			vccio		vccio		vccio	
21	rxdata60			rxdata44		rxvld		rxdata23		rxdata7
22				rxdata56		rxdata40		rxckRD		rxdata19
23	vss			vss		vss		vss		vss
24				rxdata57		rxdata41		rxckn		rxdata20
25	rxdata59				rxdata43		rxtrk		rxdata22	
26				rxdata58		vss		rxckp		rxdata6
27	vss			rxdata42		vss		vss		rxdata5
28		vccfwdio		vccfwdio			vccfwdio		vccfwdio	
29	vss				vss			vss		
30		txdata5		vccio		vccio		txdata42		vccio
31	vccio			txdata21		txckp		vccio		txdata58
32		txdata6			txdata22		txtrk		txdata43	
33	txdata4			txdata20		txckn		txdata41		txdata57
34		vccio			vccio		vccio		vccio	
35	txdata3			txdata19		txckRD		txdata40		txdata56
36		txdata7			txdata23		txvld		txdata44	
37	vss			vss		vss		vss		
38		txdata8			txdata24		txvldRD		txdata45	
39	txdata2			txdata18		vss		txdata39		txdata55
40		vccio			vccio		vccio		vccio	
41	txdata1			txdata17		txdataRD1		txdata38		txdata54
42		txdata9			txdata25		txdataRD2		txdata46	
43	vss			vss		vss		vss		
44		txdata10			txdata26		txdata32		txdata47	
45	txdata0			txdata16		txdata31		txdata37		txdata53
46		txdata11			txdata27		vccio		txdata48	
47	vccio			txdata15		txdata30		txdata36		txdata52
48		txdata12		vccio			txdata33		vccio	
49	vss			vss		vss		vss		
50		txdata13			txdata28		txdata34		txdata49	
51	txdataRD0			vss		txdata29		txdata35		txdata50
52		vccio			vss		vss		vccio	
53	vccio			vccio		vccio		vccio		vccio
							Die Edge			

Note:

In Figure 5-24, at 45-um pitch, the module depth of the 10-column bump map as shown is approximately 1225 um. Rows 1, 2, and 53 are required for packaging solutions using floating bridges without through-silicon vias (TSVs). They can be optional for packaging solutions with TSVs.

Figure 5-25. 16-column x64 Advanced Package Bump Map Example for 16 GT/s Implementation



Note:

In Figure 5-25, at 25-um pitch, the module depth of the 16-column bump map as shown is approximately 400 um. Rows 1 and 31 are required for packaging solutions using floating bridges without TSVs. They can be optional for packaging solutions with TSVs.

Figure 5-26. 8-column x64 Advanced Package Bump Map Example for 32 GT/s Implementation

	1	2	3	4	5	6	7	8
1	vss		vccio		vccio		vss	
2		vss		vccio		vccio		vss
3	vss		vccio		vccio		vss	
4		rxcksbRD		rxcksb		rxdatab		rxdatabRD
5	txdatabRD		txdatab		txcksb		txcksbRD	
6		rxdata50		vss		rxdata14		rxdataRD0
7	rxdataRD3		rxdata49		rxdata27		rxdata13	
8		vccio		rxdata36		vss		vccio
9	rxdata63		rxdata48		rxdata28		rxdata12	
10		rxdata51		rxdata35		rxdata15		rxdata0
11	vss		vss		vss		vss	
12		rxdata52		rxdata34		rxdata16		rxdata1
13	rxdata62		rxdata47		rxdata29		rxdata11	
14		vccio		vccio		vccio		vccio
15	rxdata61		rxdata46		rxdata30		rxdata10	
16		rxdata53		rxdata33		rxdata17		rxdata2
17	rxdata60		rxdata37		rxdata31		rxdata9	
18		rxdata54		rxdata32		rxdata26		rxdata3
19	vss		vss		vss		vss	
20		rxdata55		rxdataRD2		rxdata25		rxdata4
21	rxdata59		rxdata38		rxdataRD1		rxdata8	
22		vccio		vccio		rxdata24		vccio
23	rxdata56		rxdata39		vccio		rxdata18	
24		rxdata45		rxvldRD		rxdata23		rxdata7
25	vss		rxdata40		vss		vss	
26		vccio		rxvld		rxdata22		rxdata6
27	rxdata57		vss		rxckRD		rxdata19	
28		rxdata44		vccio		vccio		vccio
29	rxdata58		rxdata41		rxckn		rxdata20	
30		rxdata43		rxtrk		rxdata21		rxdata5
31	vss		rxdata42		rxckp		vss	
32		vccfdio		vccfdio		vccfdio		vccfdio
33	vss		vss		vss		vss	
34		vccio		txckp		txdata42		vccio
35	txdata5		txdata21		txtrk		txdata43	
36		txdata20		txckn		txdata41		txdata58
37	vss		vss		vss		txdata44	
38		txdata19		txckRD		vccio		txdata57
39	txdata6		txdata22		txvld		txdata40	
40		vccio		vccio		txdata45		vccio
41	txdata7		txdata23		txvldRD		txdata45	
42		txdata18		vss		txdata39		txdata56
43	vss		txdata24		vss		vss	
44		txdata8		txdataRD1		txdata38		txdata59
45	txdata4		txdata25		txdataRD2		txdata55	
46		vccio		vccio		vccio		vccio
47	txdata3		txdata26		txdata32		txdata54	
48		txdata9		txdata31		txdata37		txdata60
49	txdata2		txdata17		txdata33		txdata53	
50		txdata10		txdata30		txdata46		txdata61
51	vss		vss		vss		vss	
52		txdata11		txdata29		txdata47		txdata62
53	txdata1		txdata16		txdata34		txdata52	
54		vccio		vccio		vccio		vccio
55	txdata0		txdata15		txdata35		txdata51	
56		txdata12		txdata28		txdata48		txdata63
57	vss		vss		txdata36		vss	
58		txdata13		txdata27		txdata49		txdataRD3
59	txdataRD0		txdata14		vss		txdata50	
60		vccio		vccio		vccio		vccio
61	vccio		vccio		vccio		vccio	
					Die Edge			

Note: In [Figure 5-26](#), at 55-um pitch, the module depth of the 8-column bump map as shown is approximately 1705 um. Rows 1, 2, and 61 are required for packaging solutions using floating bridges without TSVs. They can be optional for packaging solutions with TSVs.

5.7.2.3 x32 Advanced Package Module Bump Map

UCIE also defines a x32 Advanced Package Module that supports 32 Tx and 32 Rx data signals and two redundant bumps each for Tx and two for Rx (total of four) for lane-repair functions. All other signals, including the sidebands, are the same as those of the x64 Advanced Package.

[Figure 5-27](#), [Figure 5-28](#), and [Figure 5-29](#) show the reference bump matrix for the 10-column, 16-column, and 8-column x32 Advanced Package Modules, respectively. The lower left corner of the bump map will be considered "origin" of a bump matrix and the leftmost column is Column 0.

It is strongly recommended to follow the bump matrices provided in [Figure 5-27](#), [Figure 5-28](#), and [Figure 5-29](#) for x32 Advanced Package Modules.

The following rule must be followed for the 10-column x32 Advanced Package bump matrix:

- The signals order within a column must be preserved. For example, Column 0 must contain the signals: `txdataRD0`, `txdata0`, `txdata1`, `txdata2`, `txdata3`, `txdata4`, and `txdatasbRD`. Similarly, 16-column and 8-column x32 Advanced Packages must preserve the signal order within a column of the respective bump matrices.

It is strongly recommended to follow the supply and **ground** pattern shown in the bump matrices. It must be ensured that sufficient supply and **ground** bumps are provided to meet channel characteristics (FEXT and NEXT) and power-delivery requirements.

When instantiating multiple x32 Advanced Package Modules, the same rules as defined in [Section 5.7.2.2](#) must be followed.

Figure 5-27. 10-column x32 Advanced Package Bump Map

Column0	Column1	Column2	Column3	Column4	Column5	Column6	Column7	Column8	Column9
vss		vss		vccio		vccio		vss	
	vss		vccio		vccio		vss		vss
vss		vss		vccio		vccio		vss	
	rxcksbRD		rxcksb		vccio		rxdatasb		rxdatasbRD
txdatasbRD		txdatasb		vss		txcksb		txcksbRD	
	vss		txdata22		rxdata31		vccio		vccio
vss		txdata21		txckp		rxdata30		rxdata13	
	txdata5		txdata23		vss		rxdata14		vccio
vccio		txdata20		txckn		rxdata29		rxdata12	
	txdata6		vss		rxckRD		rxdata15		rxdataRDO
txdata4		vss		txckRD		rxvldRD		rxdata11	
	txdata7		txdata24		txtrk		rxdata27		vss
vss		txdata19		txtrk		rxvld		rxdata10	
	txdata8		txdata25		rxvld		rxdata16		rxdata0
txdata3		txdata18		vss		rxdata26		vss	
	vss		txdata26		vss		rxdata17		rxdata1
txdata2		txdata17		txvld		rxdata25		rxdata9	
	txdata9		vss		rxtrk		rxdata18		vss
vccio		vccio		vccio		vccfwdio		vccfwdio	
	txdata10		txdata27		rxckRD		rxdata19		rxdata2
txdata1		txdata16		txvldRD		rxdata24		rxdata8	
	txdata11		txdata28		rxckn		rxdata20		rxdata3
txdata0		vss		vss		vss		rxdata7	
	txdata12		txdata29		rxckp		vss		vss
vss		txdata15		txdataRD1		rxdata23		rxdata6	
	txdata13		txdata30		vss		rxdata21		rxdata4
txdataRDO		txdata14		txdata31		rxdata22		rxdata5	
	vccio		vccio		vccfwdio		vccfwdio		vccfwdio
vccio		vccio		vccio		vccfwdio		vccfwdio	

Note: In Figure 5-27, at 45- μm pitch, the module depth of the 10-column reference bump matrix as shown is approximately 680.5 μm .

Figure 5-28. 16-column x32 Advanced Package Bump Map

Note: In Figure 5-28, at 25- μm pitch, the module depth of the 16-column reference bump matrix as shown is approximately 237.5 μm .

Figure 5-29. 8-column x32 Advanced Package Bump Map

Column0	Column1	Column2	Column3	Column4	Column5	Column6	Column7
vss	vss	vss	vss	vss	vss	vss	vss
vccio		vccio	vccio		vccio	vccio	vccio
rxcksbRD	rxcksb	txdatasb	txdatasb	txcksb	rxdatasbRD	txcksbRD	
txdata5	txdata23	rxdata30		rxdata13			
txdata6	txdata22	txckp		rxdata14		rxdata12	rxdataRDO
txdata7	txdata21	txckn		rxdata15		rxdata11	
txdata8	txdata20	txdata24	vss	rxdata16		rxdata10	rxdata0
vss	txdata19	txdata25	txckRD	rxdata17		rxdata9	rxdata1
txdata4	vss	txdata18	txtrk	rxdata29		rxdata8	
txdata3	txdata17	txdata26	rxvldRD	rxdata28		rxdata7	rxdata2
vss	txdata27	vss	rxvld	rxdata27	vss		rxdata3
vccio	txdata8	vccio	vcfcwdio	rxdata26	vcfcwdio		vss
txdata2	txdata9	txvldRD		rxdata17			
txdata10	txdata28	rxtrk				rxdata4	
vss	txdata11	vss	rxckRD	vss		rxdata18	
txdata11	txdata16	txdataRD1	vss	rxdata25		rxdata19	
txdata0	vss	txdata15	txdata31	rxckn		rxdata24	vss
txdata12	txdata14	vss	rxckp	vss		rxdata21	rxdata6
vss	txdata13	txdata30	rxckp	rxdata23		rxdata20	rxdata5
txdataRDO	vccio	vccio	vss	vcfcwdio	vcfcwdio		vcfcwdio
vccio	vccio	vcfcwdio	vcfcwdio	vcfcwdio	vcfcwdio		
Die Edge							

Note:

In [Figure 5-29](#), at 55-um pitch, the module depth of the 8-column reference bump matrix as shown is approximately 962.5 um.

[Figure 5-30](#) shows the signal exit order for the 10-column x32 Advanced Package bump map.

Figure 5-30. 10-column x32 Advanced Package Bump Map: Signal Exit Order

Tx Breakout		Left to Right											
		txdataRDO	txdata0	txdata1	txdata2	txdata3	txdata4	txdata5	txdata6	txdata7	txdata8	Cont...	
Cont...		txdata9	txdata10	txdata11	txdata12	txdata13	txdata14	txdata15	txdata16	txdata17	txdata18	Cont1...	
Cont1...		txdata19	txdata20	txdata21	txdata22	txdata23	txdata24	txdata25	txdata26	txdata27	txdata28	Cont2...	
Cont2...		txdata29	txdata30	txdata31	txdataRD1	txvldRD	txvld	txtrk	txckRD	txckn	txckp		
Rx Breakout		Left to Right											
		rxckp	rxckn	rxckRD	rxtrk	rxvld	rxvldRD	rxdataRD1	rxdata31	rxdata30	rxdata29	Cont...	
Cont...		rxdata28	rxdata27	rxdata26	rxdata25	rxdata24	rxdata23	rxdata22	rxdata21	rxdata20	rxdata19	Cont1...	
Cont1...		rxdata18	rxdata17	rxdata16	rxdata15	rxdata14	rxdata13	rxdata12	rxdata11	rxdata10	rxdata9	Cont2...	
Cont2...		rxdata8	rxdata7	rxdata6	rxdata5	rxdata4	rxdata3	rxdata2	rxdata1	rxdata0	rxdataRDO		

IMPLEMENTATION NOTE — x32 BUMP MAPS FOR MAX SPEED

This Implementation Note is formulated to provide PHY implementations a set of reference x32 bump maps to encompass the max speed specified.

Bump maps in [Figure 5-31](#), [Figure 5-32](#), and [Figure 5-33](#) are the x32 implementation references for the corresponding max speed with an enhancement of the power and ground bumps. They all comply with the bump matrix rules in [Section 5.7.2.3](#), and they maintain the backward compatibility in terms of signal exit order. These reference examples have been optimized for signal integrity, power integrity, lane-to-lane skew, electro-migration stress and bump area based on most of the advanced packaging technologies in the industry. Please note that technology requirements vary, and it is still required to verify the bump map with the technology provider for actual implementation requirements and performance targets.

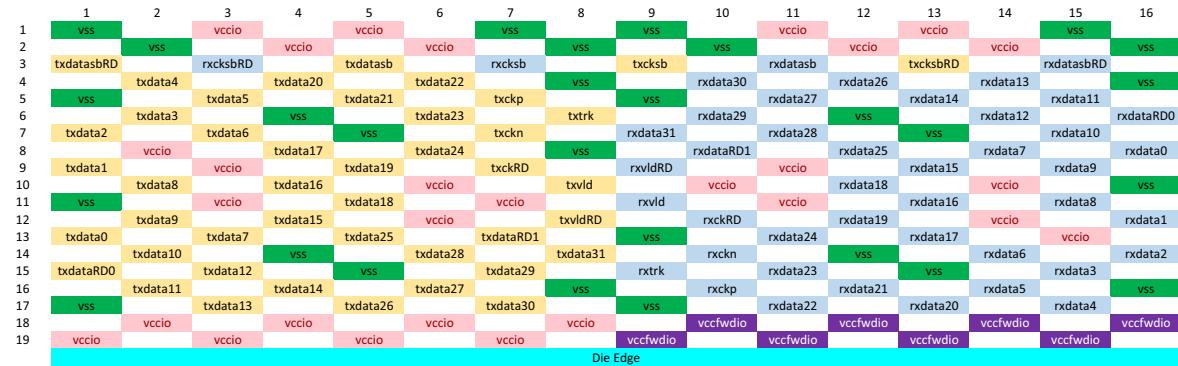
Figure 5-31. 10-column x32 Advanced Package Bump Map Example for 32 GT/s Implementation

	1	2	3	4	5	6	7	8	9	10
1	vss		vss		vccio		vccio		vss	
2		vss		vccio		vccio		vss		vss
3	vss		vss		vccio		vccio		vss	
4		rxcksbRD		rxcksb		vss		rxdata14		rxdata1RD0
5	txdatasbRD		txdatasb			txckp		rxdata14		txcksbRD
6	txdata5		vccio		vss		rxdata30		rxdata13	
7	txdata4		txdata21		txckp		rxdata30		rxdata12	
8	txdata6		txdata22		vccio		vccio		rxdata11	
9	vss		vss		txckn		rxdata29		rxdata10	
10		txdata7		txdata23		rxckRD		rxdata15		vccio
11	vccio		txdata20		txckRD		rxdata28		vss	
12		vccio		vccio		vccio		rxdata16		rxdata0
13	txdata3		txdata19		txtrk		rxdata27		rxdata11	
14		txdata8		txdata24		rxdataRD1		vccio		rxdata1
15	vss		vss		vss		vss		rxdata10	
16		txdata9		txdata25		rxvldRD		rxdata17		vss
17	txdata2		txdata18		txvld		rxvld		vss	
18		vccio		txdata26		rxvldRD		rxdata18		rxdata2
19	vccio		txdata17		txvldRD		rxdata25		rxdata9	
20		txdata10		vccio		vccio		vccio		vccio
21	txdata1		vss		txdataRD1		rxdata24		rxdata8	
22		txdata11		txdata27		rxtrk		rxdata19		rxdata3
23	txdata0		txdata16		vss		vccfwdio		vccfwdio	
24		vccio		txdata28		rxckRD		rxdata20		vss
25	vss		txdata15		txdata31		rxckn		rxdata7	
26		txdata12		txdata29		rxckn		vss		vss
27	txdataRD0		vss		vss		rxdata22		rxdata6	
28		txdata13		txdata30		rxckp		rxdata21		rxdata4
29	vss		txdata14		vss		vss		rxdata5	
30		vccio		vccio		vccfwdio		vccfwdio		vccfwdio
31	vccio		vccio		vccio		vccfwdio		vccfwdio	
						Die Edge				

Note:

In [Figure 5-31](#), at 45-um pitch, the module depth of the 10-column bump map as shown is approximately 725 um. Rows 1, 2, and 31 are required for packaging solutions using floating bridges without through-silicon vias (TSVs). They can be optional for packaging solutions with TSVs. The vccfwdio bumps are required for the tightly coupled mode up to 16 GT/s. For higher speeds, the vccfwdio bumps may be connected to the vccio bumps in package.

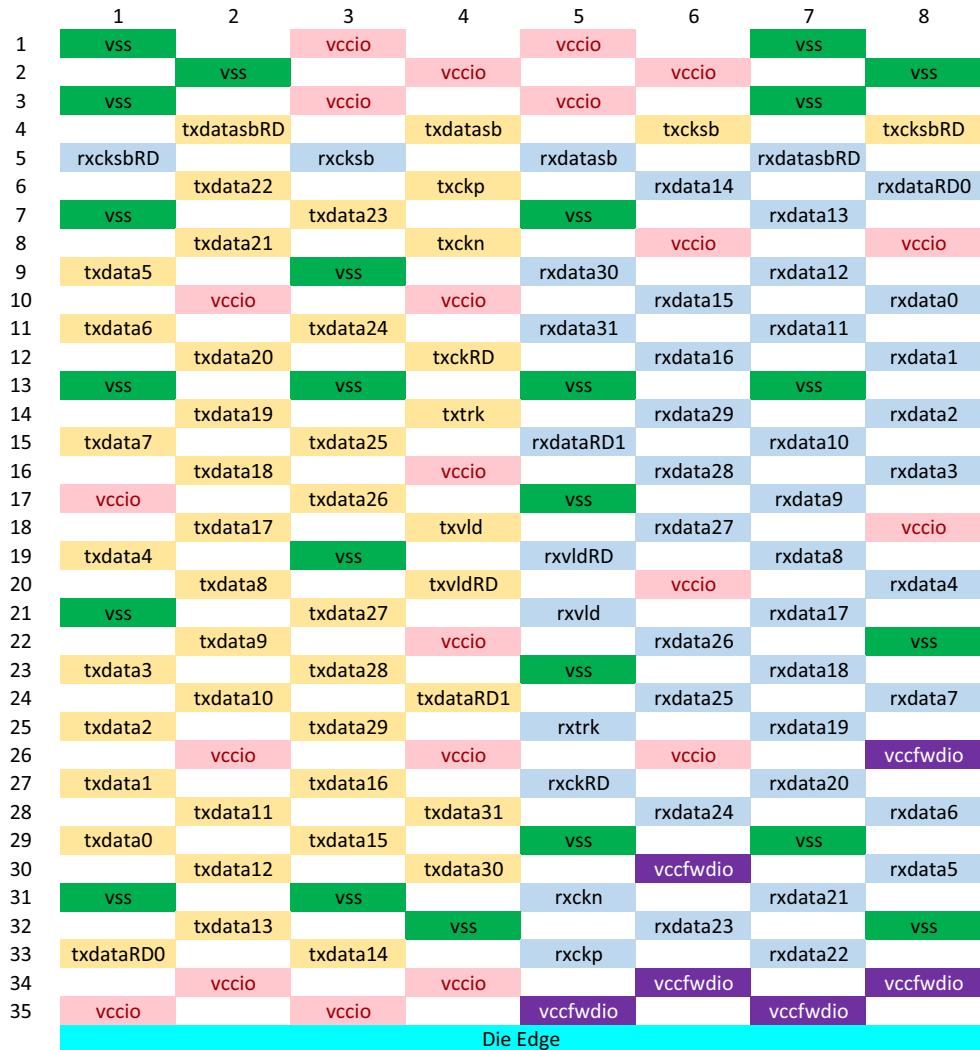
Figure 5-32. 16-column x32 Advanced Package Bump Map Example for 16 GT/s Implementation



Note:

In Figure 5-32, at 25-um pitch, the module depth of the 16-column bump map as shown is approximately 250 um. Rows 1 and 19 are required for packaging solutions using floating bridges without TSVs. They can be optional for packaging solutions with TSVs. The vccfwdio bumps are required for the tightly coupled mode up to 16 GT/s. For higher speeds, the vccfwdio bumps may be connected to the vccio bumps in package.

Figure 5-33. 8-column x32 Advanced Package Bump Map Example for 32 GT/s Implementation



Note:

In Figure 5-33, at 55-um pitch, the module depth of the 8-column bump map as shown is approximately 990 um. Rows 1, 2, and 35 are required for packaging solutions using floating bridges without TSVs. They can be optional for packaging solutions with TSVs. The vccfdio bumps are required for the tightly coupled mode up to 16 GT/s. For higher speeds, the vccfdio bumps may be connected to the vccio bumps in package.

These bump maps have been optimized to minimize the lane to-lane routing mismatch, which is not avoidable when two different bumps at different bump pitches interoperate. Table 5-14 summarizes the max skew due to bump locations for the representative cases. As a rule of thumb, each 150-um mismatch causes about 1-ps timing skew. This skew can be reduced or eliminated by the length matching effort in package channel layout design.

Table 5-14. Maximum Systematic Lane-to-lane Length Mismatch in um between the Reference Bump Maps in the Implementation Note

Rx	16-column x64 at 25 um	16-column x32 at 25 um	10-column x64 at 45 um	10-column x32 at 45 um	8-column x64 at 55 um	8-column x32 at 55 um
Tx						
16-column x64 at 25 um	0	125	351	399	560	605
16-column x32 at 25 um		0	351	393	563	618
10-column x64 at 45 um			0	159	351	463
10-column x32 at 45 um				0	428	398
8-column x64 at 55 um					0	468
8-column x32 at 55 um						0

5.7.2.4 x64 and x32 Advanced Package Module Interoperability

x64 and x32 Advanced Package Module bump maps enable interoperability between all Tx and Rx combinations of x64 or x32, 10-column, 16-column, or 8-column Modules, in both Normal-to-Normal module orientation or Normal-to-Mirrored module orientation.

However, if x64 to x32 modules or x32 to x32 modules have normal and mirrored orientation as shown in [Figure 5-34](#) and [Figure 5-35](#), respectively, signal traces between the TX half and RX half will crisscross and require swizzling technique which refers to rearranging the physical connections between signal bumps of two chiplets to optimize the layout and routing on the interposer or substrate. It involves changing the order of the connections or route on different layers without altering the netlist or the electrical functionality of the design. Moreover, connections between 8-column, 16-column, and 10-column modules may need to be routed to adjacent columns (swizzle and go across). In all cases, the electrical spec must be met for all these connections.

It is optional for a x64 Module to support interoperability with a x32 Module. The following requirements apply when a x64 module supports x32 interoperability:

- When a x64 module connects to x32 module, the connection shall always be contained to the lower half of the x64 module. This must be followed even with x32 lane reversal described below.
- Electrical specifications must be met for combinations that require signal-routing swizzling.
- Lane reversal will not be permitted on CKP-, CKN-, CKRD-, VLD-, VLDRD-, TRK-, and sideband-related pins. These pins need to be connected appropriately. Swizzling for these connections is acceptable.
- x64 module must support a lane-reversal mode in a x32 manner (i.e., TD_P[31:0] = TD_L[0:31]. When a x64 module is connected to a x32 module, in either Normal or Mirrored orientation, the upper 32 bits are not used and should be disabled.
- It is not permitted for a single module of larger width to simultaneously interop with two or more modules of a lower width. For example, a x64 Advanced Package module physically connected to two x32 Advanced Package modules is prohibited.

Additional technological capabilities or layers may be needed to accomplish swizzling on data/auxiliary signals.

Table 5-15 summarizes the connections between combinations of x64 and x32 modules in both Normal-to-Normal and Normal-to-Mirrored module orientations. The table applies to all combinations of 10-column, 16-column, or 8-column modules on either side of the Link.

Table 5-15. x64 and x32 Advanced Package Connectivity Matrix

-			Normal Module		Mirrored Module	
			Rx			
			x64	x32	x64	x32
Normal Module	Tx	x64	TX[63:0] – RX[63:0] ^a	TX[31:0] – RX[31:0] ^b	rTX[63:0] – RX[0:63] ^{c d}	rTX[31:0] – RX[0:31] ^{c e}
		x32	TX[31:0] – RX[31:0] ^b	TX[31:0] – RX[31:0] ^b	rTX[31:0] – RX[0:31] ^{c e}	rTX[31:0] – RX[0:31] ^{c e}

- a. Entry "TX[63:0] – RX[63:0]" is for Normal Module connections between two x64 modules without lane reversal. This applies to x64-to-x64 combination.
- b. Entry "TX[31:0] – RX[31:0]" is for Normal Module connections between lower 32-bit half without lane reversal. This applies to x64-to-x32, x32-to-x64, and x32-to-x32 combinations.
- c. The prefix "r" means lane reversal is enabled on the Transmitter lanes, and:
 - "rTX[63:0]" means TD_P[63:0] = TD_L[0:63], to be connected with RD_P[0:63]
 - "rTX[31:0]" means TD_P[31:0] = TD_L[0:31], to be connected with RD_P[0:31]
- d. Entry "rTX[63:0] – RX[0:63]" = Normal-to-Mirrored Module connections between two x64 modules with TX lane reversal. This applies to x64-to-x64 Normal-to-Mirrored combinations.
- e. Entry "rTX[31:0] – RX[0:31]" = Normal-to-Mirrored Module connections between lower 32-bit half with TX lane reversal. This applies to x64-to-x32, x32-to-x64, and x32-to-x32 Normal-to-Mirrored combinations.

The defined bump matrices can achieve optimal skew between bump matrices of differing depths, and the worst-case trace-reach skews are expected to be within the maximum lane-to-lane skew limit for the corresponding data rates as defined in [Section 5.3](#) and [Section 5.4](#).

[Figure 5-34](#) and [Figure 5-35](#) show examples of normal and mirrored x64-to-x32 and x32-to-x32 Advanced Package Module connections, respectively.

Figure 5-34. Example of Normal and Mirrored x64-to-x32 Advanced Package Module Connection

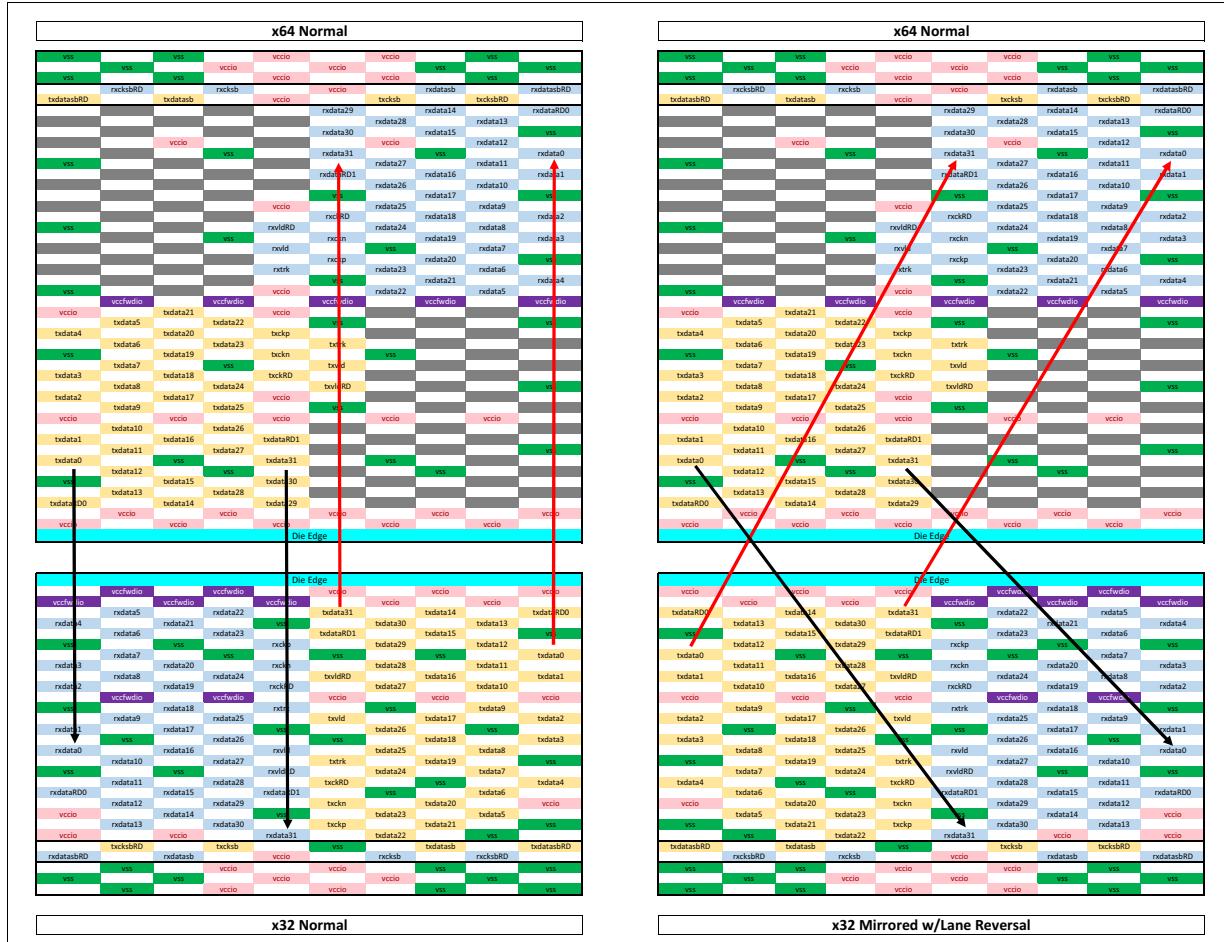
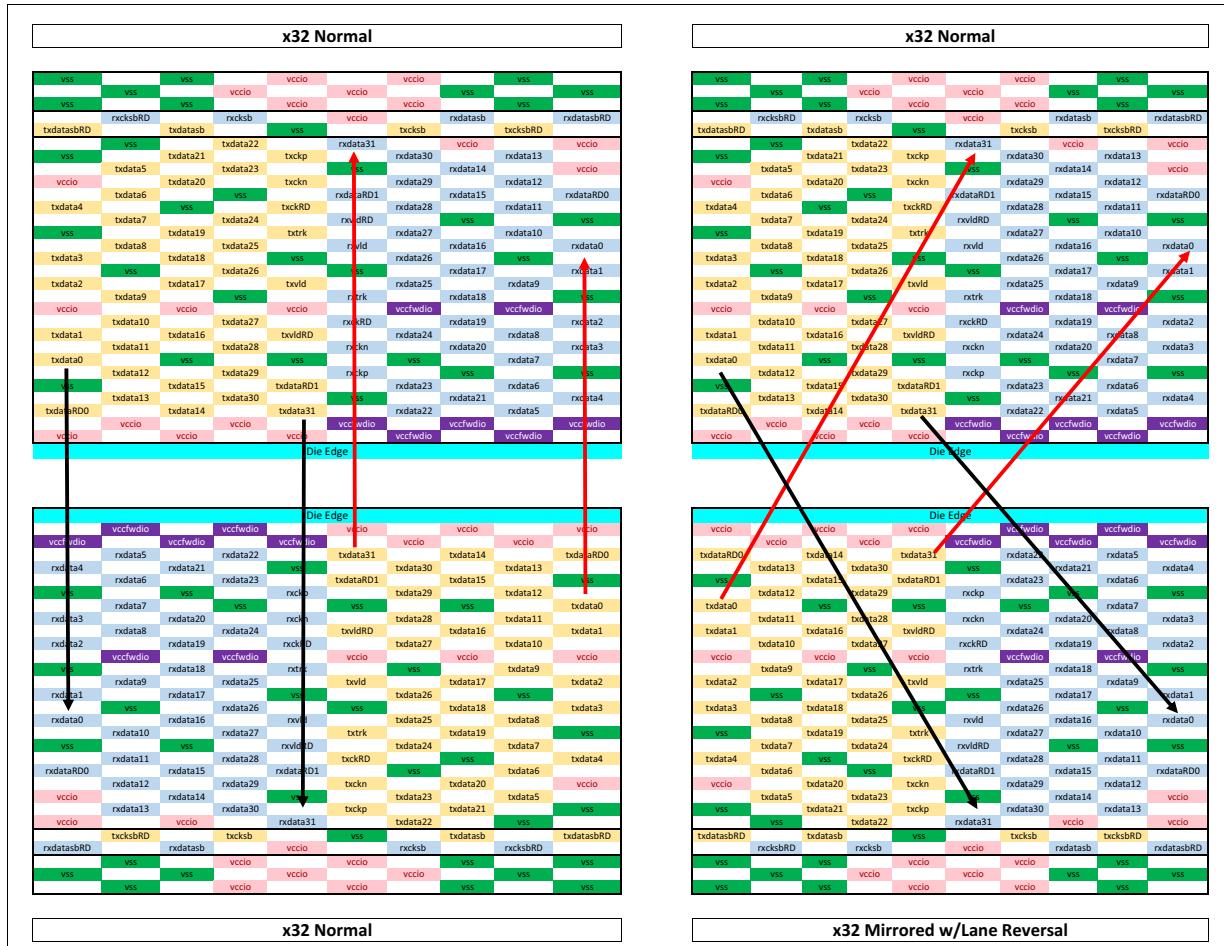


Figure 5-35. Example of Normal and Mirrored x32-to-x32 Advanced Package Module Connection



5.7.2.5 Module Naming of Advanced Package Modules

This section describes the Module naming convention of x64 and x32 Advanced Package modules in a multi-module configuration.

The Module naming is defined to help with connecting the Modules deterministically which, in turn, will help minimize the multiplexing requirements in the Multi-module PHY Logic (MMPL).

The naming of M0, M1, M2, and M3 will apply to 1, 2, or 4 Advanced Package modules that are aggregated through the MMPL.

Figure 5-36 shows the naming convention for 1, 2, or 4 Advanced Package Modules when they are connected to their “Standard Die Rotate” Module counterparts that have same number of Advanced Package Modules.

Note: The double-ended arrows in Figure 5-36 through Figure 5-39 indicate Module-to-Module connections.

Figure 5-36. Naming Convention for One-, Two-, and Four-module Advanced Package Paired with “Standard Die Rotate” Configurations

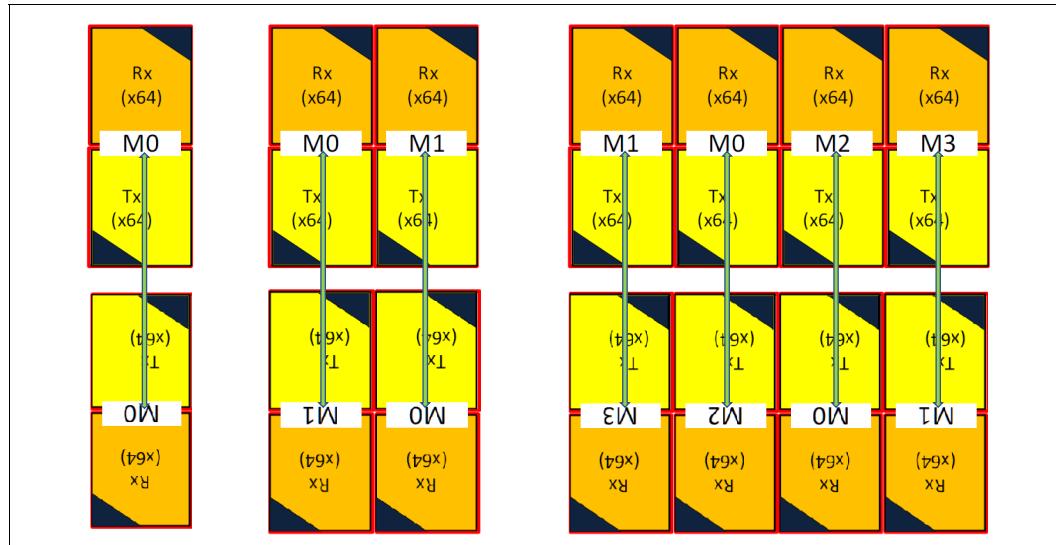


Figure 5-37 shows the naming convention for 1, 2, or 4 Advanced Package modules when they are connected to their “Mirrored Die Rotate” counterparts with the same number of Advanced Package modules.

Figure 5-37. Naming Convention for One-, Two-, and Four-module Advanced Package Paired with “Mirrored Die Rotate” Configurations

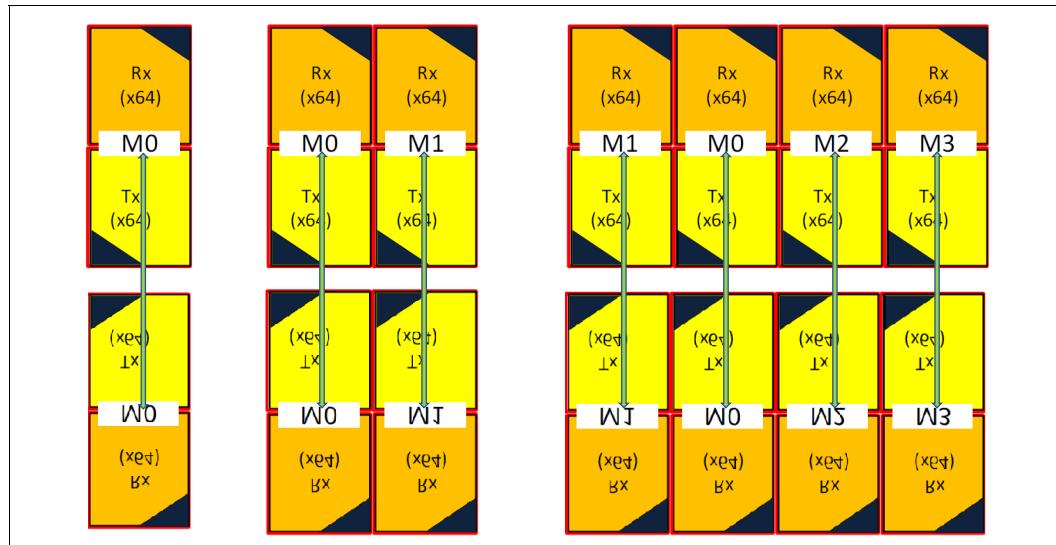


Table 5-16 summarizes the connections between the combinations shown in Figure 5-36 and Figure 5-37.

Table 5-16. Summary of Advanced Package Module Connection Combinations with Same Number of Modules on Both Sides

Advanced Package Module Connections (Same # of Modules on Both Sides)	Standard Die Rotate Counterpart	Mirrored Die Rotate Counterpart
x1 – x1	• M0 – M0	• M0 – M0
x2 – x2	• M0 – M1 • M1 – M0	• M0 – M0 • M1 – M1
x4 – x4	• M0 – M2 • M1 – M3 • M3 – M1 • M2 – M0	• M0 – M0 • M1 – M1 • M2 – M2 • M3 – M3

Figure 5-38 shows the naming convention for 1, 2, or 4 Advanced Package modules when they are connected to their “Standard Die Rotate” counterparts that have a different number of Advanced Package modules.

Figure 5-38. Examples for Advanced Package Configurations Paired with “Standard Die Rotate” Counterparts, with a Different Number of Modules

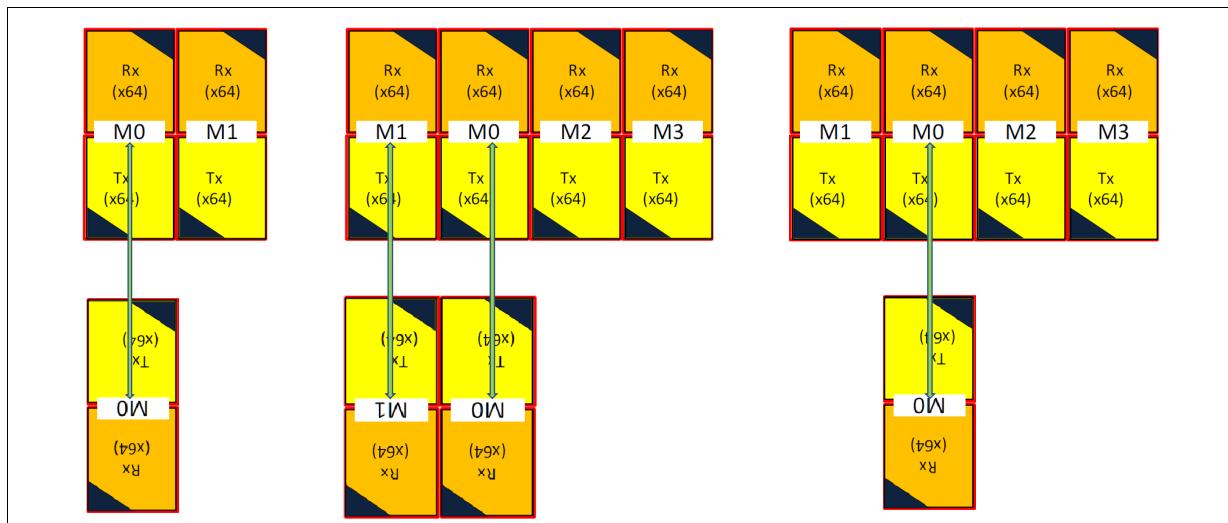


Figure 5-39 shows the naming convention for 1, 2, or 4 Advanced Package modules when they are connected to their “Mirrored Die Rotate” counterparts that have a different number of Advanced Package modules.

Figure 5-39. Examples for Advanced Package Configurations Paired with “Mirrored Die Rotate” Counterparts, with a Different Number of Modules

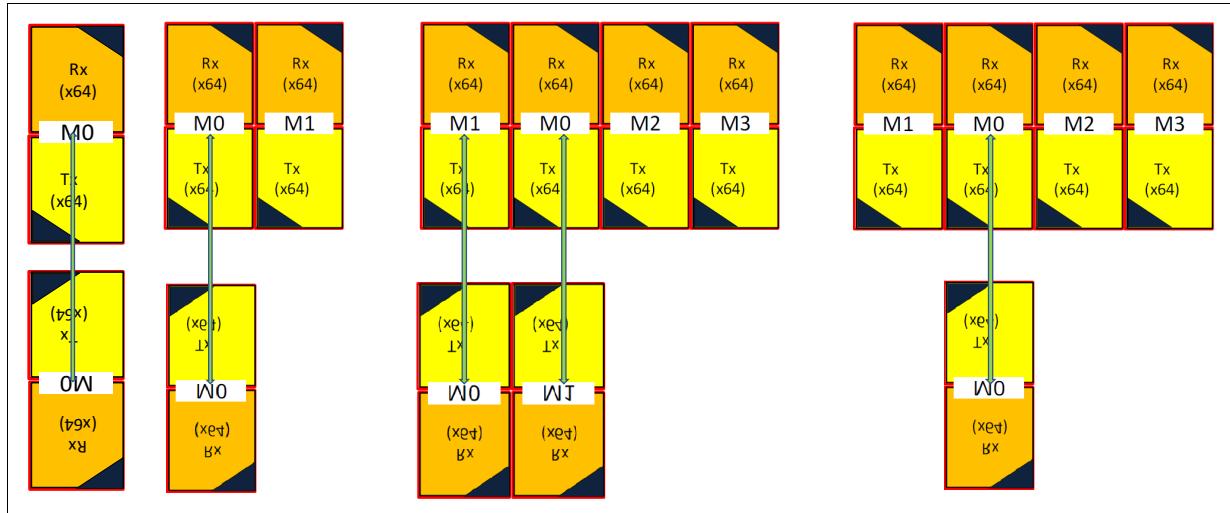


Table 5-17 summarizes the connections between the combinations shown in Figure 5-38 and Figure 5-39.

Table 5-17. Summary of Advanced Package Module Connection Combinations with Different Number of Modules on Both Sides

Advanced Package Module Connections (Different # of Modules on Both Sides)	Standard Die Rotate Counterpart ^a	Mirrored Die Rotate Counterpart ^a
x2 – x1	<ul style="list-style-type: none"> • M0 – M0 • M1 – NC 	<ul style="list-style-type: none"> • M0 – M0 • M1 – NC
x4 – x2	<ul style="list-style-type: none"> • M0 – M0 • M1 – M1 • M3 – NC • M2 – NC 	<ul style="list-style-type: none"> • M0 – M1 • M1 – M0 • M2 – NC • M3 – NC
x4 – x1	<ul style="list-style-type: none"> • M0 – M0 • M1 – NC • M3 – NC • M2 – NC 	<ul style="list-style-type: none"> • M0 – M0 • M1 – NC • M2 – NC • M3 – NC

a. NC indicates no connection.

5.7.3 Standard Package

Interconnect channel should be designed with 50 ohm characteristic impedance. Insertion loss and crosstalk for requirement at Nyquist frequency with Receiver termination is defined in [Table 5-18](#).

Table 5-18. IL and Crosstalk for Standard Package: With Receiver Termination Enabled

Data Rate	4, 8 GT/s	12, 16 GT/s	24, 32 GT/s
VTF Loss (dB) ^{a b c}	$L(0) > -4.5$ $L(f_N) > -7.5$	$L(0) > -4.5$ $L(f_N) > -6.5$	$L(0) > -4.5$ $L(f_N) > -7.5$
VTF Crosstalk (dB)	$XT(f_N) < 3 * L(f_N) - 11.5$ and $XT(f_N) < -25$	$XT(f_N) < 3 * L(f_N) - 11.5$ and $XT(f_N) < -25$	$XT(f_N) < 2.5 * L(f_N) - 10$ and $XT(f_N) < -26$

a. Voltage Transfer Function for 4 GT/s and 8 GT/s (Tx: 30 ohm / 0.3pF; Rx: 50 ohm / 0.3pF).

b. Voltage Transfer Function for 12 GT/s and 16 GT/s (Tx: 30 ohm / 0.2pF; Rx: 50 ohm / 0.2pF).

c. Voltage Transfer Function for 24 GT/s and 32 GT/s (Tx: 30 ohm / 0.125pF; Rx: 50 ohm / 0.125pF).

IL and crosstalk for requirement at Nyquist frequency without Receiver termination is defined by [Table 5-19](#). Loss and crosstalk specifications between DC and Nyquist f_N follow the same methodology defined in [Section 5.7.2.1](#).

Table 5-19. IL and Crosstalk for Standard Package: No Rx Termination

Data Rate	4-12 GT/s	16 GT/s
VTF Loss (dB) ^{a b}	$L(f_N) > -1.25$	$L(f_N) > -1.15$
VTF Crosstalk (dB)	$XT(f_N) < 7 * L(f_N) - 12.5$ and $XT(f_N) < -15$	$XT(f_N) < 4 * L(f_N) - 13.5$ and $XT(f_N) < -17$

a. Voltage Transfer Function for 4 GT/s and 8 GT/s (Tx: 30 ohm / 0.3pF; Rx: 0.2 pF).

b. Voltage Transfer Function for 12 GT/s and 16 GT/s (Tx: 30 ohm / 0.2pF; Rx: 0.2 pF).

Table 5-20. Standard Package Module Signal List (Sheet 1 of 2)

Signal Name	Count	Description
Data		
TXDATA[15:0]	16	Transmit Data
TXVLD	1	Transmit Data Valid; Enables clocking in corresponding module
TXTRK	1	Transmit Track signal
TXCKP	1	Transmit Clock Phase-1
TXCKN	1	Transmit Clock Phase-2
RXDATA[15:0]	16	Receive Data
RXVLD	1	Receive Data Valid; Enables clocking in corresponding module
RXTRK	1	Receive Track
RXCKP	1	Receive Clock Phase-1
RXCKN	1	Receive Clock Phase-2
Sideband		
TXDATASB	1	Sideband Transmit Data
RXDATASB	1	Sideband Receiver Data
TXCKSB	1	Sideband Transmit Clock

Table 5-20. Standard Package Module Signal List (Sheet 2 of 2)

Signal Name	Count	Description
RXCKSB	1	Sideband Receive Clock
Power and Voltage		
VSS		Ground Reference
VCCIO		I/O supply
VCCAON		Always on Aux supply (sideband)

5.7.3.1 x16 Standard Package Module Bump Map

Figure 5-40 and Figure 5-42 show the reference bump matrices for x16 (one module) and x32 (two module) Standard Packages, respectively.

It is strongly recommended to follow the bump matrices provided in Figure 5-40 for one module and Figure 5-42 for two module Standard Packages. The lower left corner of the bump map will be considered “origin” of a bump matrix.

Signal exit order for x16 and x32 Standard Package bump matrices are shown in Figure 5-41 and Figure 5-43, respectively.

The following rules must be followed for Standard Package bump matrices:

- The signals within a column must be preserved. For example, for a x16 (one module Standard Package) shown in Figure 5-40, Column 1 must contain the signals: `txdata0`, `txdata1`, `txdata4`, `txdata5`, and `txdatasb`.
- The signals must exit the bump field in the order shown in Figure 5-41. Layer 1 and Layer 2 are two different signal routing layers in a Standard Package.

It is strongly recommended to follow the supply and **ground** pattern shown in the bump matrices. It must be ensured that sufficient supply and **ground** bumps are provided to meet channel characteristics (FEXT and NEXT) and power-delivery requirements.

The following rules must be followed for instantiating multiple modules of Standard Package bump matrix:

- When looking at a die such that the UCIe Modules are on the south side, Tx should always precede Rx within a module along the die’s edge when going from left to right.
- When instantiating multiple modules, the modules must be stepped in the same orientation and abutted. Horizontal or vertical mirroring is not permitted.

If more Die Edge Bandwidth density is required, it is permitted to stack two modules before abutting. If two modules are stacked, the package may need to support at least four routing layers for UCIe signal routing. An example of stacked Standard Package Module instantiations is shown in Figure 5-42.

- If only one stacked module is instantiated, when looking at a die such that the UCIe Modules are on the south side, Tx should always precede Rx within a module along the die’s edge when going from left to right.
- When instantiating multiple stacked modules, the modules must be stepped in the same orientation and abutted. Horizontal or vertical mirroring is not permitted.

Note: An example of signal routing for stacked module is shown in Figure 5-44.

Figure 5-40. Standard Package Bump Map: x16 interface

Column 0	Column 1	Column 2	Column 3	Column 4	Column 5	Column 6	Column 7	Column 8	Column 9	Column 10	Column 11
vccio	txdatasb		txcksb		vccaon		vccaon		rxcksb		rxdatasb
vccio		vccio		vccio		vccio		vccio		vccio	
vss		vss		vss		vss		vss		vss	
vccio											vss
	txdata5		txckn		txdata9		txdata11		rxdata10		rxdata4
vss		vss		vss		vss		vss		vss	
	txdata4		txckp		txdata10		txdata11		rxckp		rxdata5
vss		txdata6		txdata8		vss		rxdata9		rxdata7	
	vss		vss								
vccio		txdata3		txvld		txdata13		vccio		rxdata12	
vccio						txdata15		rxdata14		rxtrk	
vss		vss		vss		vccio		vss		vss	
	txdata1										rxdata0
vccio											
	txdata0		txtrk		txdata14		txdata15		rxvld		rxdata1
vss			txdata2		txdata12		vss		rxdata13		rxdata3
Die Edge											

Figure 5-41. Standard Package x16 interface: Signal exit order

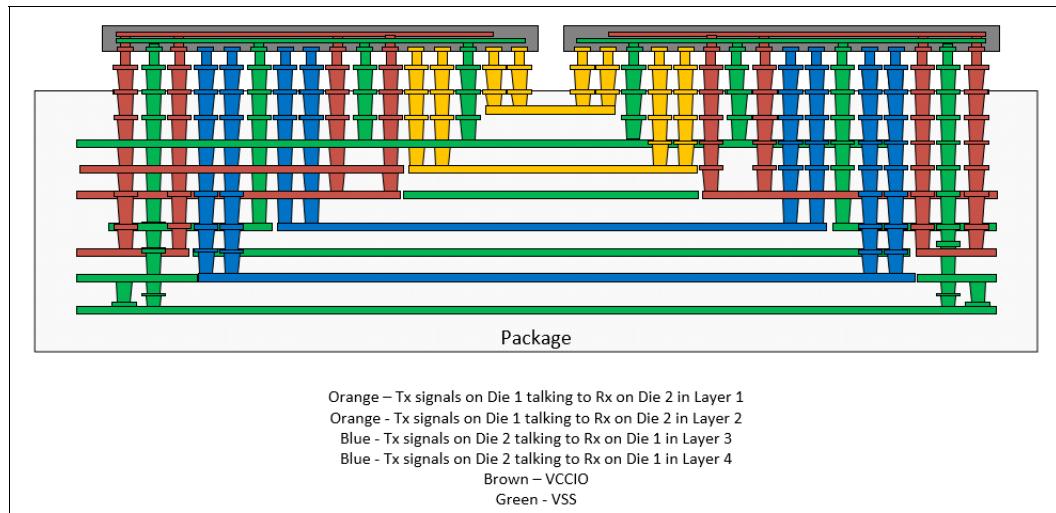
Layer 1	Tx	0	1	2	3	trk	vld	12	13	14	15	15	14	13	12	vld	trk	3	2	1	0	Rx
Layer 2	Module	4	5	6	7	ckp	ckn	8	9	10	11	11	10	9	8	ckn	ckp	7	6	5	4	Module
Sideband		txdatasb				txcksb										rxcksb					rxdatasb	

Figure 5-42. Standard Package Bump Map: x32 interface

Column 0	Column 1	Column 2	Column 3	Column 4	Column 5	Column 6	Column 7	Column 8	Column 9	Column 10	Column 11
m1txdatasb		m2rxcksb			vccaon		m2txcksb		m2txdatasb		vccaon
	m1txcksb			vccaon		vccaon		m1rxcksb		m1rxdatasb	
vccio		vccio		vccio		vccio		vccio		vccio	
vss		vss		vss		vss		vss		vss	
	m2rxdata6		m2rxdata8		vss		m2txdata9		m2txdata7		vss
m2rxdata4		m2rxckp		m2rxdata10		vss		m2txdata11		m2txckn	
vss		vss		vss		vss		vss		vss	
m2rxdata5		m2rxckn		m2rxdata11		m2txdata10		m2txckp		m2txdata4	
	m2rxdata7		m2rxdata9		vss		m2txdata8		m2txdata6		vss
vss		vss		vss		vss		vss		vss	
m2rxdata2		m2rxdata12		vss		m2txdata13		m2txdata15		m2txdata3	
m2rxdata0		m2rxtrk		m2rxdata14		vss		m2txdata15		m2txvld	
vss		vss		vss		vss		vss		m2txdata1	
m2rxdata1		m2rvld		m2rxdata15		m2txdata14		m2txtrk		m2txdata0	
	m2rxdata3		m2rxdata13		vccio		m2txdata12		m2txdata2		vccio
vccio		vccio		vccio		vccio		vccio		vccio	
vss		vss		vccio		vss		vss		vss	
m1txdata7		m1txdata9			vccio		m1rxdata10		m1rxdata8		m1rxdata6
vccio			m1txckn		m1txdata11		m1rxdata10		m1rxckp		m1rxdata4
vss			vss		vss		vss		vss		vss
m1txdata5			m1txckp		m1txdata10		m1rxdata11		m1rxckn		m1rxdata5
vss			m1txdata6		m1txdata8		vss		m1rxdata9		m1rxdata7
vccio			m1txdata3		m1txdata13		vccio		m1rxdata12		m1rxdata2
m1txdata1			m1txvld		m1txdata15		m1rxdata14		m1rxtrk		m1rxdata0
vccio			vss		vss		vccio		vss		vss
m1txdata0			m1txtrk		m1txdata14		m1rxdata15		m1rxvld		m1rxdata1
vss			m1txdata2		m1txdata12		vss		m1rxdata13		m1rxdata3
Die Edge											

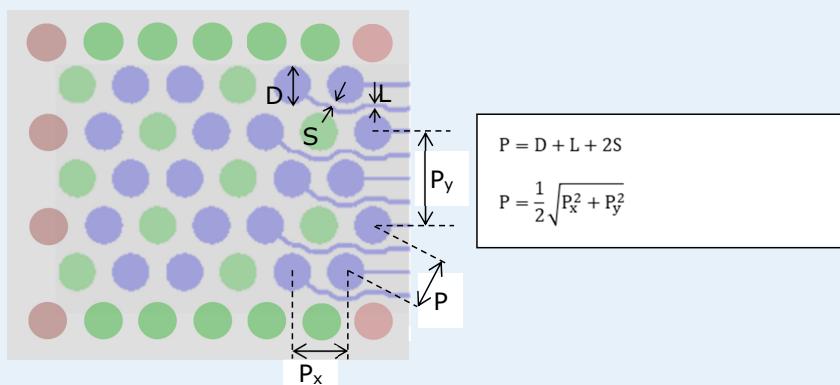
Figure 5-43. Standard Package x32 interface: Signal exit routing

Layer 1	Tx	0	1	2	3	trk	vld	12	13	14	15	15	14	13	12	vld	trk	3	2	1	0	Rx
Layer 2	Module 1	4	5	6	7	ckp	ckn	8	9	10	11	11	10	9	8	ckn	ckp	7	6	5	4	Module 1
Layer 3	Rx	0	1	2	3	trk	vld	12	13	14	15	15	14	13	12	vld	trk	3	2	1	0	Tx
Layer 4	Module 2	4	5	6	7	ckp	ckn	8	9	10	11	11	10	9	8	ckn	ckp	7	6	5	4	Module 2
Sideband		m1txdatasb	m2rxdatasb	m1txcksb	m2rxcksb	m2txcksb	m1rxcksb	m2txdatasb	m1rxdatasb													Sideband

Figure 5-44. Standard Package cross section for stacked module

IMPLEMENTATION NOTE

Figure 5-45 shows a breakout design reference with the Standard Package channel based on the bump pitch and on routing design rules.

Figure 5-45. Standard Package reference configuration

- 4-row deep breakout per routing layer
- Example 1: $P_y = 190.5 \text{ um}$, $P_x \approx 111.5 \text{ um}$, $P \approx 110 \text{ um}$
- Example 2: $P_y = 190.5 \text{ um}$, $P_x \approx 177 \text{ um}$, $P \approx 130 \text{ um}$

5.7.3.2 x8 Standard Package Module Bump Map

Designs can choose to add a UCIE-S port for sort/pre-bond test purposes in scenarios where they need the high bandwidth of UCIE, but the design is an advanced package design, or for any other reason. To reduce the chiplet's die edge when supporting such a UCIE-S usage, a x8 version of UCIE-S is provided. This is an additional option that goes beyond the available standard x16 UCIE-S port options. A UCIE-S x16 Module can optionally support connecting to a UCIE-S x8 Module and when supported, the connection is always on its lower x8 lanes (i.e., Lanes 7:0). UCIE-S x8 designs must support lane reversal and degraded mode operation to x4. UCIE-S x16 designs that support connection to a x8 Module must support lane reversal, and must support degraded mode operation to x4 on its lower 8 lanes when connected to a x8 Module.

UCIE-S x8 support is limited to a single module configuration. When a UCIE-S x8 port is connected to a multi-module x16 port, it is always connected to Module 0 UCIE-S x16.

Figure 5-46 shows the reference bump matrix for a x8 Standard Package.

Figure 5-46. Standard Package Bump Map: x8 Interface

Column 0	Column 1	Column 2	Column 3	Column 4	Column 5	Column 6	Column 7
txdatasb		txcksb		rxcksb		rxdatasb	
	vccio		vccio		vccio		vccio
vss		vss		vss		vss	
	Txdata0		Txdata7		Rxdata6		Rxdata1
vss		Txckn		vss		Rxckp	
	vss		vss		vss		vss
vccio		Txckp		vccio		Rxckn	
	Txdata1		Txdata6		Rxdata7		Rxdata0
vccio		vss		vccio		vss	
	Txdata3		Txdata4		Rxdata5		Rxdata2
vccio		Txvld		vccio		Rxtrk	
	vss		vss		vss		vss
vss		Txtrk		vss		Rxvld	
	Txdata2		Txdata5		Rxdata4		Rxdata3
Die Edge							

It is strongly recommended to follow the bump matrix provided in Figure 5-46. The lower left corner of the bump map will be considered "origin" of a bump matrix.

The same rules as mentioned for x16 and x32 Standard Package bump matrices in Section 5.7.3.1 must be followed for the x8 bump matrix.

5.7.3.3 x16 and x8 Standard Package Module Interoperability

A x8 bump matrix will either connect to another x8 bump matrix or to bits [7:0] of a x16 bump matrix.

5.7.3.4 Module Naming of Standard Package Modules

This section describes the Module naming convention of Standard Package Modules in a multi-module configuration.

The naming of M0, M1, M2, and M3 will apply to 1, 2, or 4 Standard Package modules that are aggregated through MMPL, in stacked and unstacked configuration combinations.

Figure 5-47 shows the naming convention for 1, 2, or 4 Standard Package modules when they are connected to their “Standard Die Rotate” module counterparts with the same number of Standard Package modules, with either same stack or same unstacked configuration.

Note: The double-ended arrows in [Figure 5-47](#) through [Figure 5-51](#) indicate Module-to-Module connections.

Figure 5-47. Naming Convention for One-, Two-, and Four-module Standard Package Paired with “Standard Die Rotate” Configurations

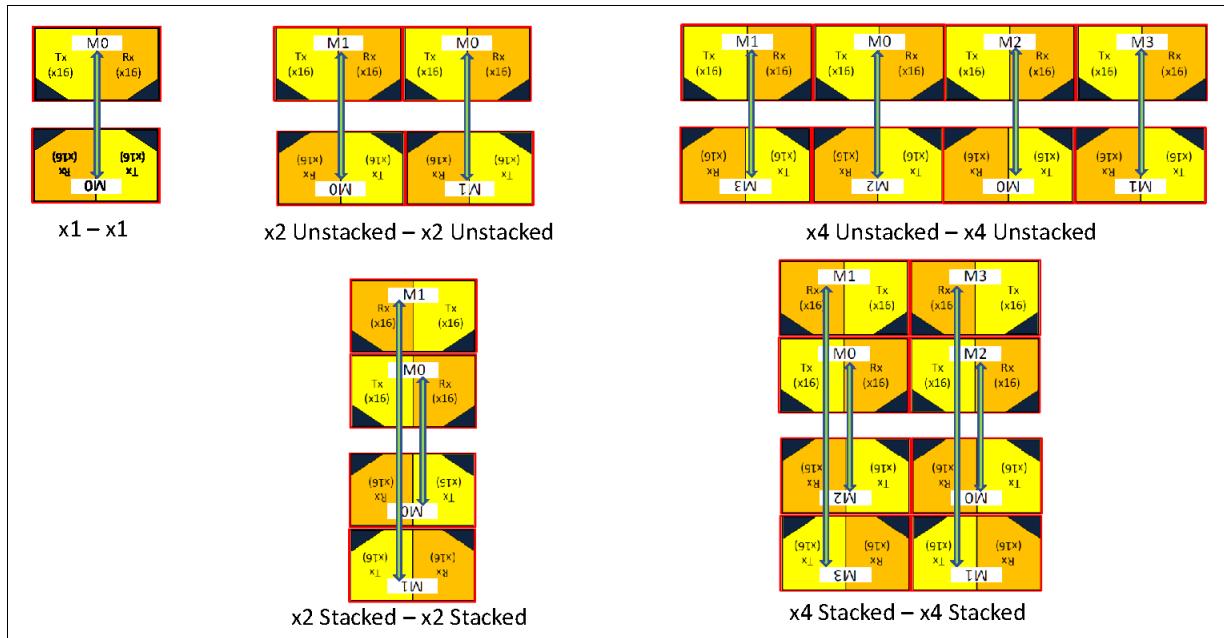


Figure 5-48 shows the naming convention for 1, 2, or 4 Standard Package modules when they are connected to their “Mirrored Die Rotate” counterparts that have same number of Standard Package modules, with either same stack or same unstacked configuration.

Figure 5-48. Naming Convention for One-, Two-, and Four-module Standard Package Paired with “Mirrored Die Rotate” Configurations

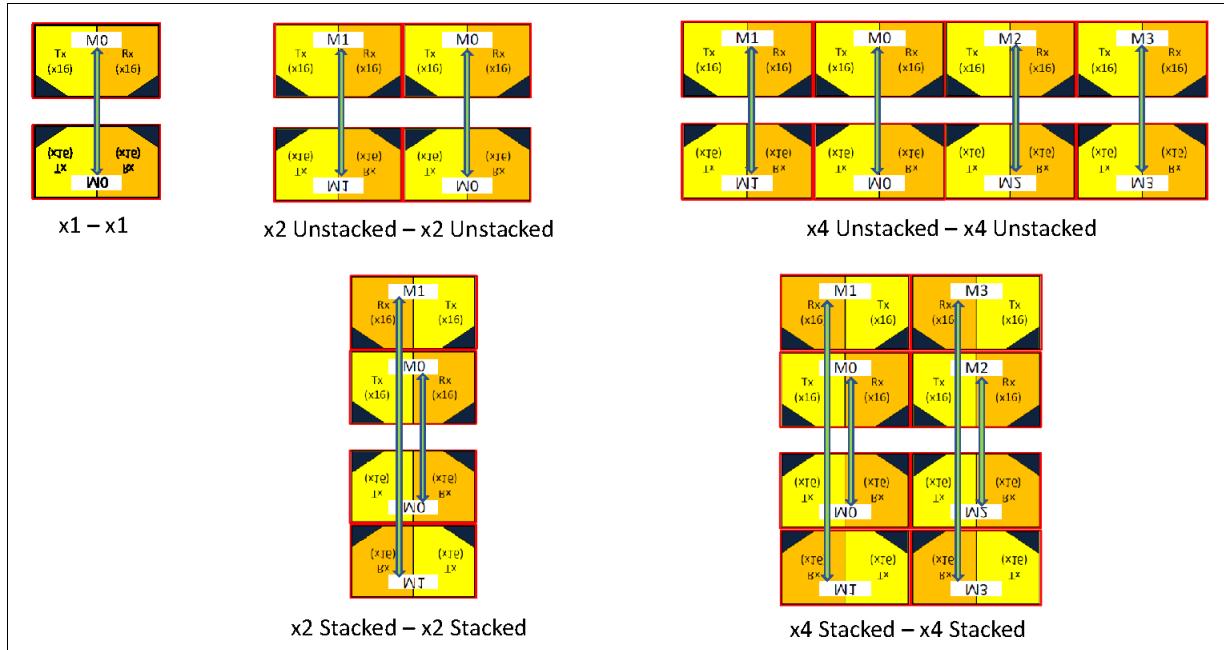


Table 5-21 summarizes the connections between the combinations shown in Figure 5-47 and Figure 5-48.

Table 5-21. Summary of Standard Package Module Connection Combinations with Same Number of Modules on Both Sides^{a b}

Standard Package Module Connections (Same # of Modules on Both Sides)	Standard Die Rotate Counterpart	Mirrored Die Rotate Counterpart	
		Option 1 (See Figure 5-48)	Option 2 ^c (See Figure 5-51)
x1 - x1	• M0 - M0	• M0 - M0	
x2 Unstacked - x2 Unstacked	• M0 - M1 • M1 - M0	• M0 - M0 • M1 - M1	
x2 Stacked - x2 Stacked	• M0 - M0 • M1 - M1	• M0 - M0 • M1 - M1	• M0 - M1 • M1 - M0
x4 Unstacked - x4 Unstacked	• M0 - M2 • M1 - M3 • M3 - M1 • M2 - M0	• M0 - M0 • M1 - M1 • M2 - M2 • M3 - M3	
x4 Stacked - x4 Stacked	• M0 - M2 • M1 - M3 • M3 - M1 • M2 - M0	• M0 - M0 • M1 - M1 • M2 - M2 • M3 - M3	• M0 - M1 • M1 - M0 • M2 - M3 • M3 - M2

a. Mirror-to-Mirror connection will be same as non-mirrored case.

b. Mirror die connectivity may have jogs and need additional layers on package.

- c. For some mirrored cases, there are possible alternative connections to allow design choices between more routing layers vs. max data rates, shown as Option 1 and Option 2 in [Table 5-21](#). For x2 – x2 Stacked and x4 – x4 Stacked cases, Option 1 typically requires 2x the routing layers and enables nominal data rates, while Option 2 enables same the layer count but at reduced max data rates due to potential crosstalk. See [Figure 5-50](#) for Option 2 connection illustrations.

[Figure 5-49](#) shows the naming convention for 1, 2, or 4 Standard Package modules when they are connected to their “Standard Die Rotate” counterparts that have a different number of Standard Package modules.

Figure 5-49. Examples for Standard Package Configurations Paired with “Standard Die Rotate” Counterparts, with a Different Number of Modules

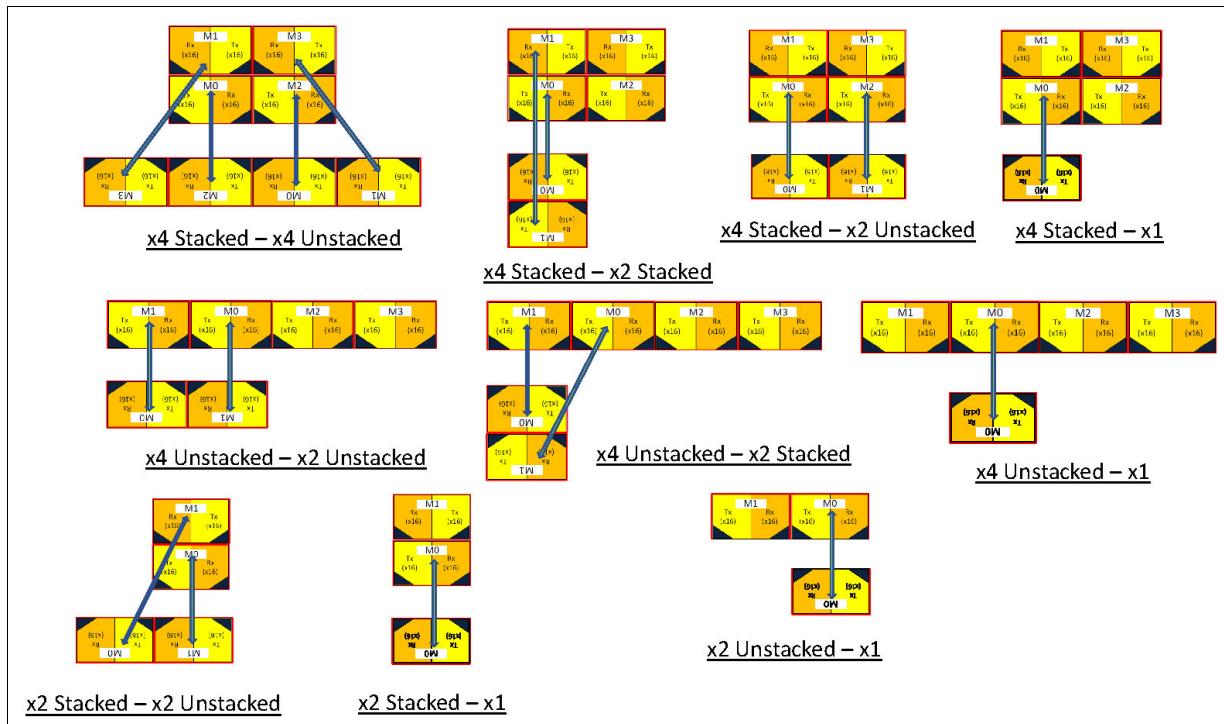


Figure 5-50 shows the naming convention for 1, 2, or 4 Standard Package Modules when they are connected to their “Mirrored Die Rotate” counterparts that have a different number of Standard Package Modules.

Figure 5-50. Examples for Standard Package Configurations Paired with “Mirrored Die Rotate” Counterparts, with a Different Number of Modules



Figure 5-51 illustrates the possible alternative connections for some mirrored cases to allow design choices between more routing layers vs. reduced max data rates due to potential crosstalk, shown as Option 2 in Table 5-21 and Table 5-22.

Figure 5-51. Additional Examples for Standard Package Configurations Paired with “Mirrored Die Rotate” Counterparts, with a Different Number of Modules

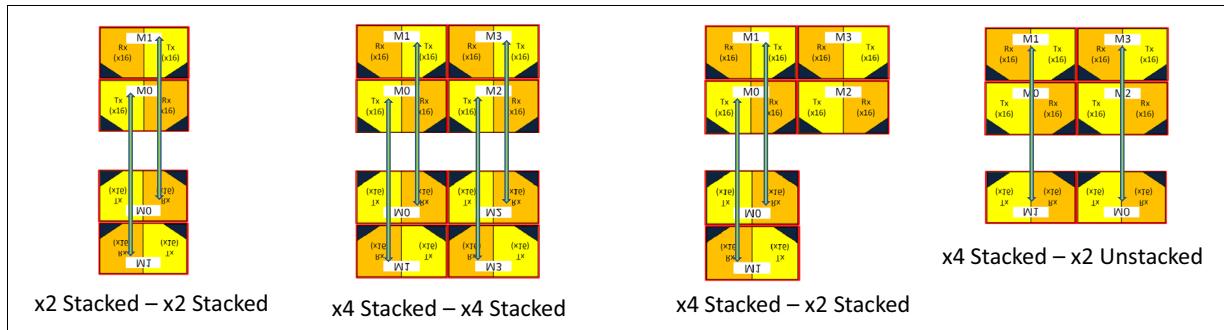


Table 5-22 summarizes the connections between the combinations shown in Figure 5-49, Figure 5-50, and Figure 5-51.

5.7.3.4.1 Module Degrade Rules

Table 5-22. Summary of Standard Package Module Connection Combinations with Different Number of Modules on Both Sides

Standard Package Module Connections (Different # of Modules on Both Sides)	Standard Die Rotate Counterpart ^a	Mirrored Die Rotate Counterpart ^a	
		Option 1 (See Figure 5-50)	Option 2 (See Figure 5-51)
x4 Stacked – x4 Unstacked	<ul style="list-style-type: none"> • M0 – M2 • M1 – M3 • M3 – M1 • M2 – M0 	<ul style="list-style-type: none"> • M0 – M0 • M1 – M1 • M2 – M2 • M3 – M3 	
x4 Stacked – x2 Stacked	<ul style="list-style-type: none"> • M0 – M0 • M1 – M1 • M3 – NC • M2 – NC 	<ul style="list-style-type: none"> • M0 – M0 • M1 – M1 • M2 – NC • M3 – NC 	<ul style="list-style-type: none"> • M0 – M1 • M1 – M0 • M2 – NC • M3 – NC
x4 Stacked – x2 Unstacked	<ul style="list-style-type: none"> • M0 – M0 • M1 – NC • M3 – NC • M2 – M1 	<ul style="list-style-type: none"> • M0 – M1 • M1 – NC • M2 – M0 • M3 – NC 	<ul style="list-style-type: none"> • M0 – NC • M1 – M1 • M2 – NC • M3 – M0
x4 Stacked – x1	<ul style="list-style-type: none"> • M0 – M0 • M1 – NC • M3 – NC • M2 – NC 	<ul style="list-style-type: none"> • M0 – M0 • M1 – NC • M3 – NC • M2 – NC 	
x4 Unstacked – x2 Unstacked	<ul style="list-style-type: none"> • M0 – M1 • M1 – M0 • M3 – NC • M2 – NC 	<ul style="list-style-type: none"> • M0 – M0 • M1 – M1 • M2 – NC • M3 – NC 	
x4 Unstacked – x2 Stacked	<ul style="list-style-type: none"> • M0 – M1 • M1 – M0 • M3 – NC • M2 – NC 	<ul style="list-style-type: none"> • M0 – M1 • M1 – M0 • M2 – NC • M3 – NC 	
x4 Unstacked – x1	<ul style="list-style-type: none"> • M0 – M0 • M1 – NC • M3 – NC • M2 – NC 	<ul style="list-style-type: none"> • M0 – M0 • M1 – NC • M3 – NC • M2 – NC 	
x2 Stacked – x2 Unstacked	<ul style="list-style-type: none"> • M0 – M1 • M1 – M0 	<ul style="list-style-type: none"> • M0 – M0 • M1 – M1 	
x2 Stacked – x1	<ul style="list-style-type: none"> • M0 – M0 • M1 – NC 	<ul style="list-style-type: none"> • M0 – M0 • M1 – NC 	
x2 Unstacked – x1	<ul style="list-style-type: none"> • M0 – M0 • M1 – NC 	<ul style="list-style-type: none"> • M0 – M0 • M1 – NC 	

a. NC indicates no connection.

On a 2-module or 4-module link, if one or more module-pairs have failed, the link will be degraded and shall comply with the following rules:

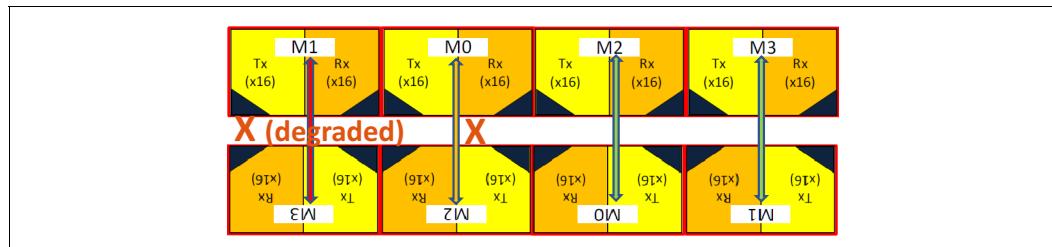
1. The degraded link shall be either one or two modules, and shall not be three modules.
 - a. For a 4-module link:
 - i. If any one module-pair failed, it shall be degraded to a 2-module link.

- ii. If any two module-pairs failed, it shall be degraded to a 2-module link.
 - iii. If any three module-pairs failed, it shall be degraded to a 1-module link.
- b. For a 2-module link:
- i. If any one module-pair failed, it shall be degraded to a 1-module link.
2. For a 4-module link, if only one module-pair failed, one additional module-pair that belongs to the "same half" (along the Die Edge) of the 4-module will be disabled/degraded.

[Figure 5-52](#) illustrates an example with a x4 Unstacked connected to a x4 Unstacked "Standard Die Rotate" counterpart with one M0 – M2 pair failed. The M1 – M3 pair on its left shall be disabled accordingly to comply with the rules defined above, which will be denoted as "x (d)" in [Table 5-23](#).

Note: The double-ended arrows in [Figure 5-52](#) indicate Module-to-Module connections.

Figure 5-52. Example of a Configuration for Standard Package, with Some Modules Disabled



[Table 5-23](#) summarizes the resulting degraded link if there are one, two, or three failed module-pairs for the x4 Unstacked to x4 Unstacked configuration.

Table 5-23. Summary of Degraded Links when Standard Package Module-pairs Fail

Module – Module Partner Pair	Number of Module-pairs Failed ^a													
	1-fail				2-fail						3-fail			
	x	x (d)	✓	✓	x	x	x	✓	✓	✓	x	x	x	✓
M0 – M2	x	x (d)	✓	✓	x	x	x	✓	✓	✓	x	x	x	✓
M1 – M3	x (d)	x	✓	✓	x	✓	✓	x	x	✓	x	x	✓	x
M3 – M1	✓	✓	x	x (d)	✓	x	✓	x	✓	x	x	✓	x	x
M2 – M0	✓	✓	x (d)	x	✓	✓	x	✓	x	x	✓	x	x	x

a. x = Failed Module – Module Partner Pair.

x (d) = Disabled Module – Module Partner Pair to comply with Degrade rules.

✓ = Functional Module – Module Partner Pair.

All other module configurations shall follow the same Module Degrade rules as defined above.

5.7.4 UCIe-S Sideband-only Port

A UCIe-S sideband-only port is also permitted for test/manageability purposes. The RDI signals to the sideband port for a sideband-only configuration are the same as for a sideband with mainband configuration (see [Chapter 10.0](#) for details of the latter).

[Figure 5-53](#) shows the bump map for a UCIe-S sideband-only port. [Figure 5-54](#) shows the supported configurations for a UCIe-S sideband-only port.

Figure 5-53. UCIe-S Sideband-only Port Bump Map

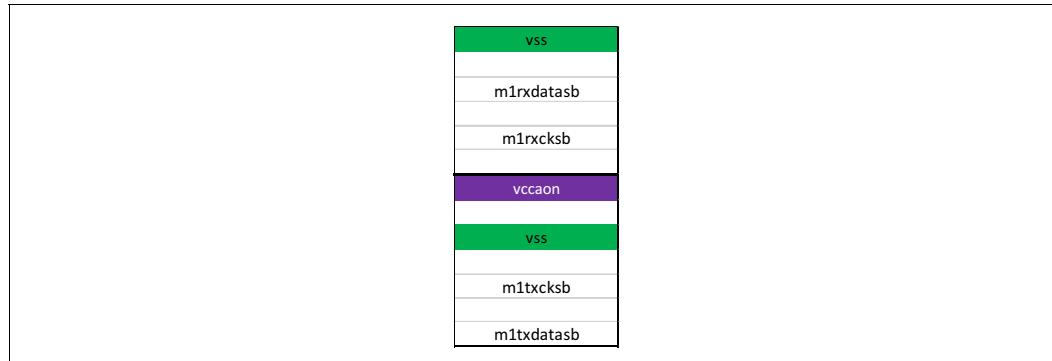
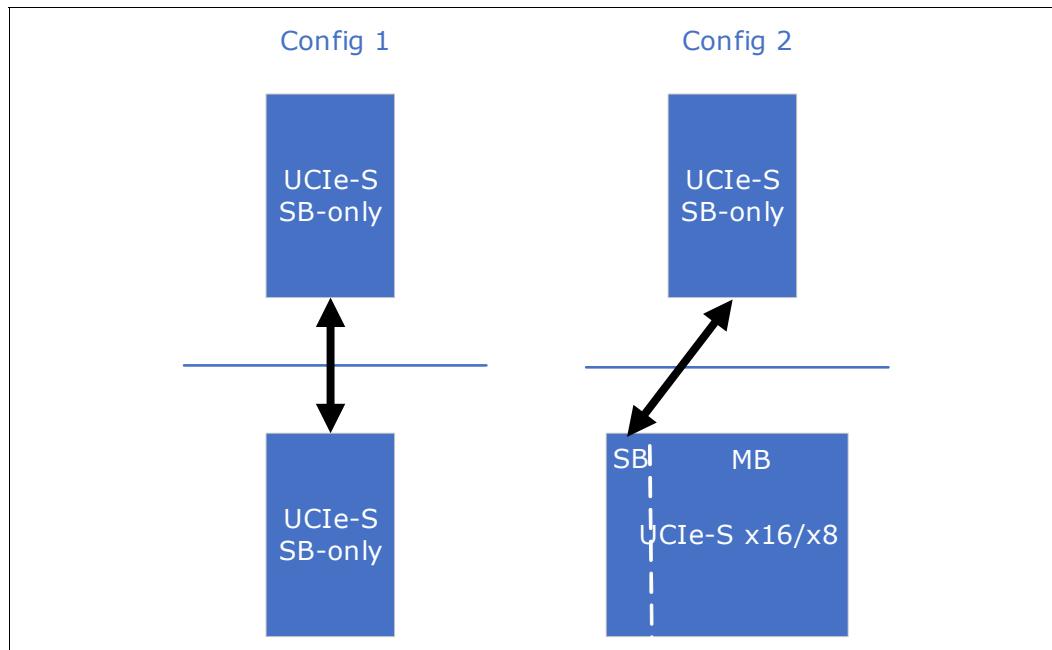


Figure 5-54. UCIe-S Sideband-only Port Supported Configurations



5.8 Tightly Coupled Mode

Tightly Coupled PHY mode is defined as when both of the following conditions are met:

- Shared Power Supply between Tx and Rx, or Forwarded Power Supply from Tx to Rx
- Channel supports larger eye mask defined in [Table 5-24](#)

In this mode, there is no Receiver termination and the Transmitter must provide full swing output. In this mode, further optimization of PHY circuit and power reduction is possible. For example, a tuned inverter can potentially be used instead of a front-end amplifier. Training complexity such as voltage reference can be simplified.

Table 5-24. Tightly Coupled Mode: Eye Mask

Data Rate	4-16 GT/s
Overall (Eye Closure due to Channel) ^a	
Eye Height ^b	250 mV
Eye Width (rectangular eye mask with specified eye height)	0.7 UI

a. With 750-mV Transmitter signal swing.

b. Centered around VCCFWDIO/2.

Loss and crosstalk requirement follow the same VTF method, adjusting to the eye mask defined in [Table 5-24](#). [Table 5-25](#) shows the specification at Nyquist frequency.

Table 5-25. Tightly Coupled Mode Channel for Advanced Package

Data Rate	4-12 GT/s	16 GT/s
VTF Loss ^a (dB)	$L(f_N) > -3$	-
VTF Crosstalk ^a (dB)	$XT(f_N) < 1.5 * L(f_N) - 21.5$ and $XT(f_N) < -23$	-

a. Based on Voltage Transfer Function (Tx: 25 ohm / 0.25 pF; Rx: 0.2 pF).

Loss and crosstalk specifications between DC and Nyquist f_N follow the same methodology defined in [Section 5.7.2.1](#).

Although the use of this mode is primarily for Advanced Package, it may also be used for Standard Package when two Dies are near one another and Receiver must be unterminated.

5.9 Interconnect redundancy Remapping

5.9.1 Advanced Package Lane Remapping

Interconnect Lane remapping is supported in Advanced Package Module to improve assembly yield and recover functionality. Each module supports:

- Four redundant bumps for Data
- One redundant bump for Clock and Track
- One redundant bump for Valid

For x64 Advanced Package modules, the four redundant bumps for data repair are divided into two groups of two. [Figure 5-55](#) shows an illustration of x64 Advanced package module redundant bump assignment for data signals. TRD_P0 and TRD_P1 are allocated to the lower 32 data Lanes and TRD_P2 and TRD_P3 are allocated to the upper 32 data Lanes. Each group is permitted to remap up

to two Lanes. For example, TD15 is a broken Lane in the lower half and TD_P32 and TD_P40 are broken Lanes in the upper 32 Lanes. [Figure 5-56](#) illustrates Lane remapping for the broken Lanes.

For x32 Advanced Package modules, only the lower 32 data lanes and TRD_P0 and TRD_P1 apply in [Figure 5-55](#) and [Figure 5-56](#).

Details and implementation of Lane remapping for Data, Clock, Track, and Valid are provided in Section 4.3.

Figure 5-55. Data Lane repair resources

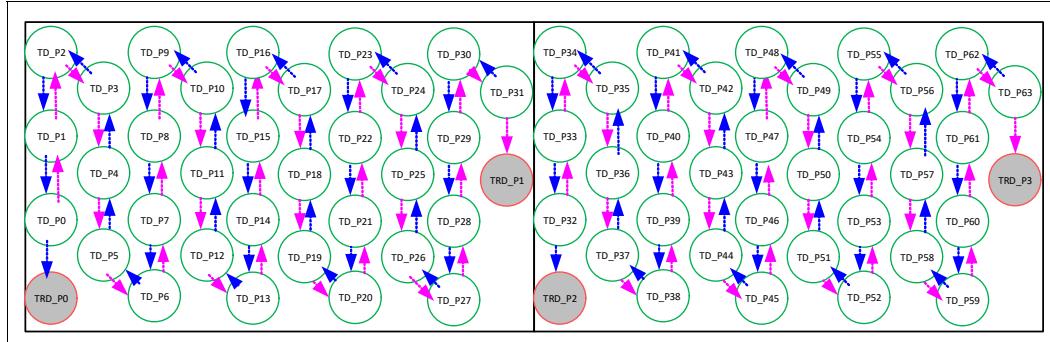
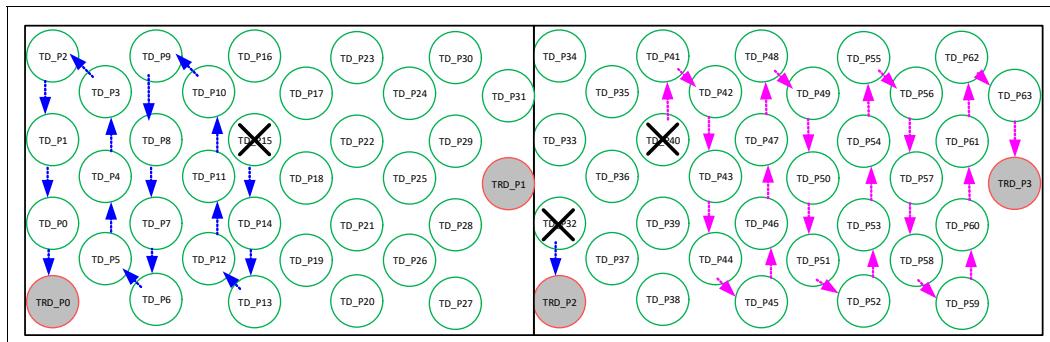


Figure 5-56. Data Lane repair



5.9.2 Standard Package Lane remapping

Lane repair is not supported in Standard Package modules.

5.10 BER requirements, CRC and retry

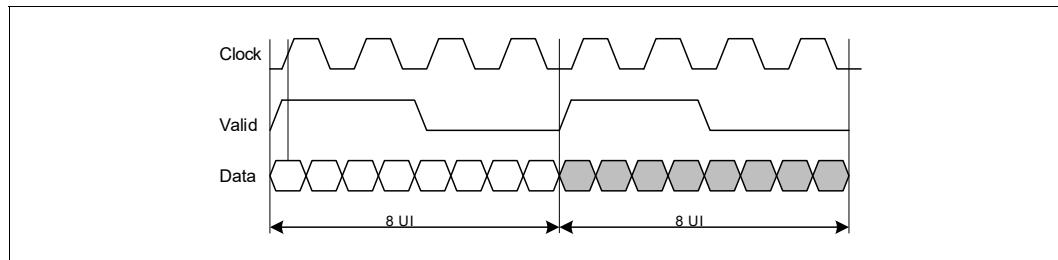
The BER requirement based on channel reach defined in [Section 5.7](#) is shown in [Table 5-26](#). Error detection and correction mechanisms such as CRC and retry are required for BER for 1E-15 to achieve the required Failure In Time (FIT) rate of significantly less than 1 (1 FIT = 1 device failure in 10^9 Hours). The UCIe spec defined CRC and retry is detailed in [Chapter 3.0](#). For the BER of 1E-27, either parity or CRC can be used and the appropriate error reporting mechanism must be invoked to ensure a FIT that is significantly less than 1.

Table 5-26. Raw BER requirements

Package Type	Data Rate (GT/s)					
	4	8	12	16	24	32
Advanced Package	1E-27	1E-27	1E-27	1E-15	1E-15	1E-15
Standard Package	1E-27	1E-27	1E-15	1E-15	1E-15	1E-15

5.11 Valid and Clock Gating

Valid is used to frame transmit data. For a single transmission of 8 UI data packet, Valid is asserted for the first 4 UI and de-asserted for the second 4 UI. [Figure 5-57](#) shows the transfer of two 8 UI data packets back to back.

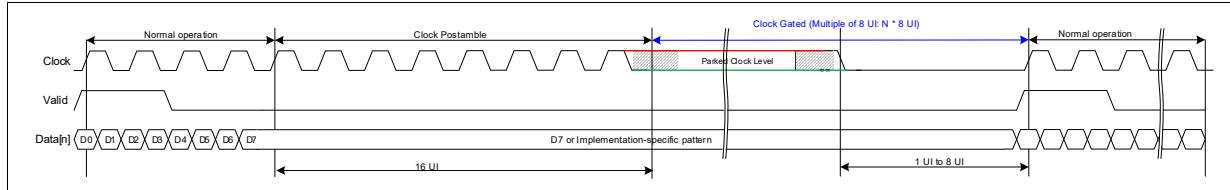
Figure 5-57. Valid Framing

As described in [Section 4.1.3](#), clock must be gated only after Valid signal remains low for 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.

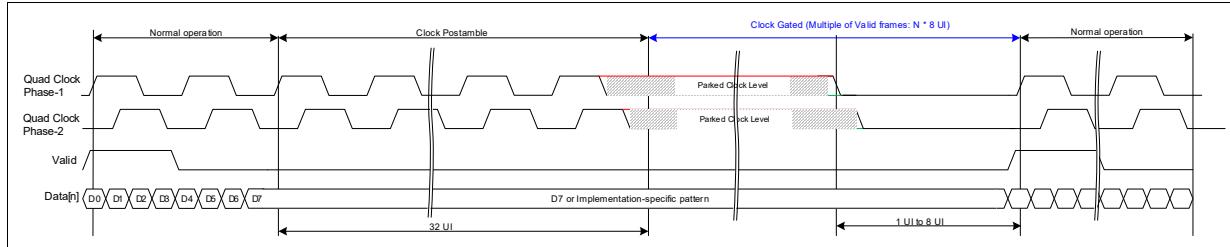
Idle state is when there is no data transmission on the mainband. During Idle state, Data, Clock, and Valid Lanes must hold values as follows:

- If the Link is unterminated (all Advanced Package and unterminated Standard Package Links), some Data Lane Transmitters are permitted to remain toggling up to the same transition density as the scrambled data without advancing the scrambler state. The remaining Data Lane Transmitters must hold the data of the last transmitted bit. Valid Lane must be held low until the next normal transmission.
 - In Strobe mode, the clock level in a clock-gated state for half-rate clocking (after meeting postamble requirement) must alternate between differential high and differential low during consecutive clock-gating events. For quarter-rate clocking, the clock level in a clock-gated state must alternate between high and low for both phases (Phase-1 and Phase-2) simultaneously. Clock must drive a differential (simultaneous) low for half- (quarter-) rate clocking for at least 1 UI or a maximum of 8 UI before normal operation. The total clock-gated period must be an integer multiple of 8 UI. Example shown in [Figure 5-58](#) and [Figure 5-59](#).
 - In Continuous mode, the clock remains free running (examples shown in [Figure 5-60](#)). Total idle period must be an integer multiple of 8 UI.

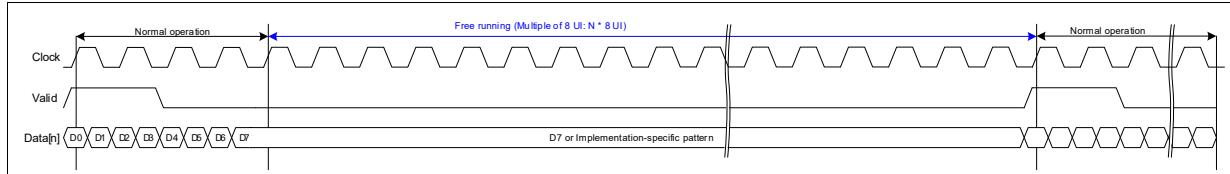
**Figure 5-58. Data, Clock, Valid Levels for Half-rate Clocking:
Clock-gated Unterminated Link**



**Figure 5-59. Data, Clock, Valid Levels for Quarter-rate Clocking:
Clock-gated Unterminated Link**



**Figure 5-60. Data, Clock, Valid Levels for Half-rate Clocking:
Continuous Clock Unterminated Link**



- If the Link is terminated (Standard Package terminated Links), some Data Lane Transmitters are permitted to remain toggling up to the same transition density as the scrambled data without advancing the scrambler state. The remaining Data Lanes Transmitters hold the data of the last-transmitted bit. Valid Lane must be held low until the next normal transmission. Note that keeping the transmitter toggling will incur extra power penalty and should be applied with discretion.
 - In Strobe mode, the clock level in a clock-gated state for half-rate clocking (after meeting postamble requirement) must alternate between differential high and differential low during consecutive clock-gating events. For quarter-rate clocking, the clock level in a clock-gated state must alternate between high and low for both phases (Phase-1 and Phase-2) simultaneously. Transmitters must precondition the Data Lanes to a 0 or 1 (V) and clock must drive a differential low for at least 1 UI or up to a maximum of 8 UIs for half- (quarter-) rate clocking before the normal transmission. The total clock-gated period must be an integer multiple of 8 UI. Example shown in [Figure 5-61](#) and [Figure 5-63](#).
 - In Continuous mode, the clock remains free running (examples shown in [Figure 5-64](#)). Transmitters must precondition the Data Lanes to a 0 or 1 (V) for at least 1 UI or up to a maximum of 8 UI. Total idle period must be an integer multiple of 8 UI.

Note: Entry into and Exit from Hi-Z state are analog transitions. Hi-Z represents Transmitter state and the actual voltage during this period will be pulled Low due to termination to **ground** at the Receiver.

Figure 5-61. Data, Clock, Valid Gated Levels for Half-rate Clocking: Terminated Link

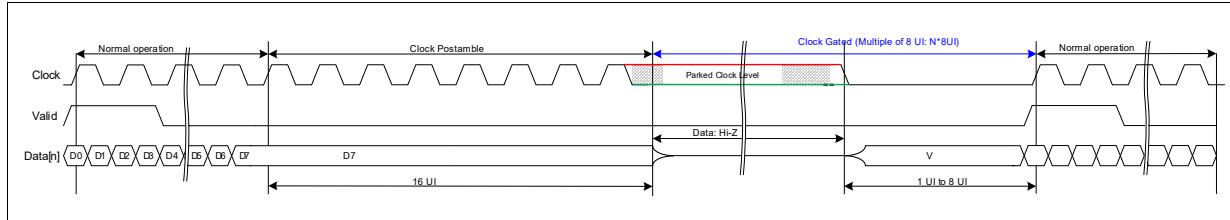


Figure 5-62. Data, Clock, Valid Gated Levels for Quarter-rate Clocking: Terminated Link

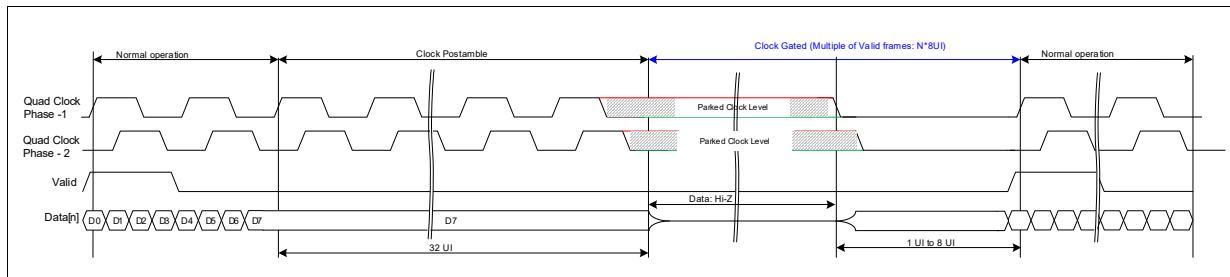
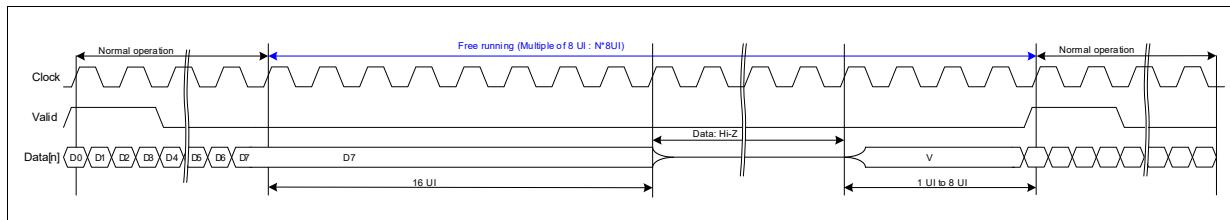


Figure 5-63. Data, Clock, Valid Gated Levels for Half-rate Clocking: Continuous Clock Terminated Link



5.12 Electrical Idle

Some training states need electrical idle when Transmitters and Receivers are waiting for generate and receive patterns.

- Electrical idle on the mainband in this Specification is described as when Transmitters and Receivers are enabled; Data, Valid and Track Lanes are held low and Clock is parked at high and low.

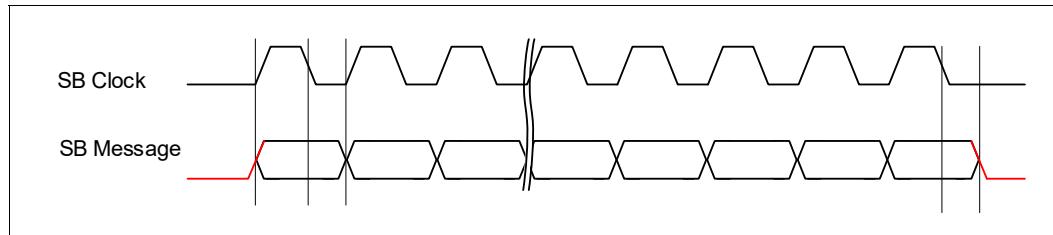
5.13 Sideband signaling

Each module supports a sideband interface. The sideband is a two-signal interface that is used for the transmit and receive directions. The sideband data is an 800 MT/s single data rate signal with an 800-MHz source. Sideband must run on power supply and clock derived from the auxiliary clock (AUXCLK) source which are always on (VCCAON). See [Section 5.13.2](#) for AUXCLK details.

Sideband data is sent edge aligned with the positive edge of the strobe. The Receiver must sample the incoming data with the strobe. The negative edge of the strobe is used to sample the data as the data uses single data rate signaling as shown in [Figure 5-64](#). Sideband transmission is described in [Section 4.1.5](#).

For Advanced Package modules, redundancy is supported for the sideband interface. Sideband initialization and repair are described in [Section 4.5.3.2](#). There is no redundancy and no Lane repair support on Standard Package modules.

Figure 5-64. Sideband signaling



5.13.1 Sideband Electrical Parameters

[Table 5-27](#) shows the sideband electrical parameters.

It is strongly recommended that the two sides of the sideband I/O Link share the same power supply rail.

Table 5-27. Sideband Parameters summary

Parameter	Min	Typ	Max	Unit
Supply voltage (VCCAON) ^a	0.65			V
TX Swing	0.8*VCCAON	-	-	V
Input high voltage (V_{IH})	0.7*VCCAON			V
Input low voltage (V_{IL})			0.3*VCCAON	V
Output high voltage (V_{OH})	0.9*VCCAON			V
Output low voltage (V_{OL})			0.1*VCCAON	V
Sideband Data Setup Time	200	-	-	ps
Sideband Data Hold Time	200	-	-	ps
Rise/Fall time for Advanced Package ^b	50	-	280	ps
Rise/Fall time for Standard Package ^c	80	-	175	ps

a. Always On power supply. The guidelines for maximum Voltage presented in [Section 1.5](#) apply to sideband signaling.

b. 20 to 80% of VCCAON level with Advanced Package reference channel load.

c. 20 to 80% of VCCAON level with Standard Package reference channel load.

5.13.2 Auxiliary Clock (AUXCLK)

Auxiliary clock (AUXCLK) may be from any clock source. Although other clock frequencies are possible, it is recommended that every chiplet should also use a 100-MHz clock source. Table 5-28 lists the permitted auxiliary clock frequency range. The minimum and maximum frequencies listed in the table indicate the limits, and do not indicate a requirement to support the entire frequency range. Reference clock (REFCLK; see Section 5.1.2) can be used if it is always on. Spread-Spectrum Clocking (SSC) is permitted. AUXCLK has reduced tolerances compared to REFCLK.

Table 5-28. AUXCLK Frequency Parameters

Symbol	Description	Limits			Unit	Notes
		Min	Rec	Max		
F _{AUXCLK}	AUXCLK Frequency	25	100	800	MHz	

§ §

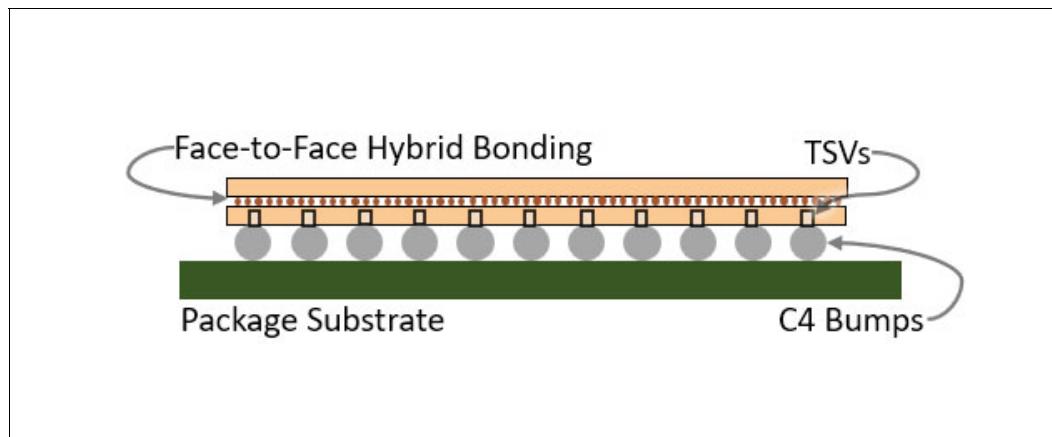
6.0 UCIE-3D

6.1 Introduction

Three-dimensional heterogeneously integrated technologies present an opportunity for the development of new electronic systems with advantages of higher bandwidth and lower power as compared to 2D and 2.5D architectures. 3D will enable applications where the scale of data movement is impractical for monolithic, 2D, or 2.5D approaches.

Universal Chiplet Interconnect express for 3D packaging (UCIE-3D) is designed as a universally applicable interface for 3D die-to-die communication. [Figure 6-1](#) illustrates an example of two dies stacked in a 3D configuration. UCIE-3D uses a two-dimensional array of interconnect bumps for data transmission between dies.

Figure 6-1. Example of 3D Die Stacking



6.2 UCIE-3D Features and Summary

While the UCIE 2D and 2.5D models strive for seamless plug-and-play interoperability, the UCIE-3D model necessitates a more-integrated approach due to the inherent characteristics of packaging technology. The objective is to offer a range of options or a “menu” from which users can select what best suits their needs. The primary objectives and general methodology for UCIE-3D are as follows:

- Circuit and logic must fit within the bump area (i.e., UCIE will continue to be bump-limited). Given the high density, this will translate to lower operating frequencies and a much-simplified circuit (e.g., at 1- μm bump-pitch, the UCIE-3D area amortized on a per-lane basis must be less than 1 μm^2).
- No D2D adapter. Low BER due to low-frequency and almost zero-channel distance — No CRC/replay is needed.
- A hardened minimal PHY such as a simple inverter/driver. The SoC Logic connects directly to the PHY.

- All debug/testability hooks are located within a common block (across all UCIE-3D Links) that is connected to the SoC Logic network inside the chiplet.
- Lane repair becomes a bundle-wide repair that is orchestrated by the SoC Logic.

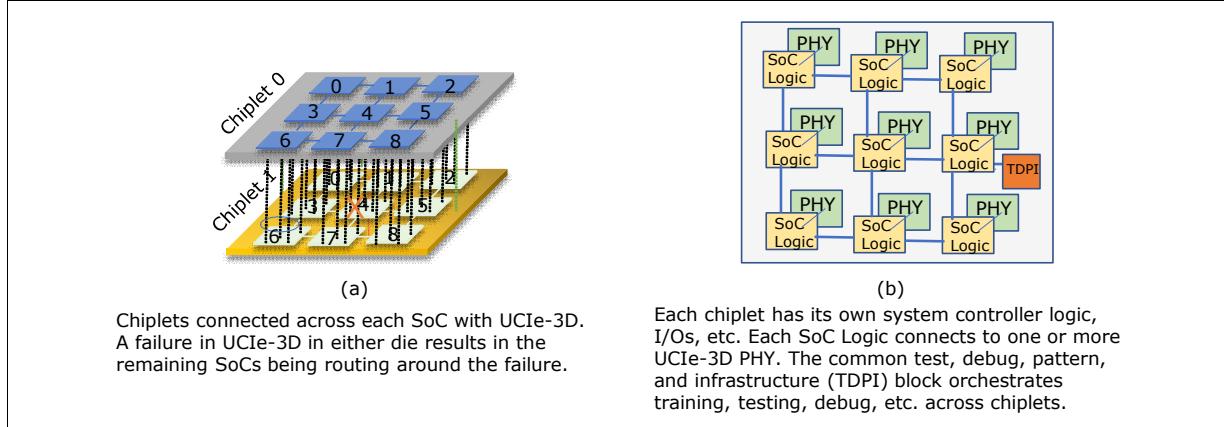
Figure 6-2. UCIE-3D Illustration

Table 6-1 summarizes the key performance indicators of the proposed UCIE-3D.

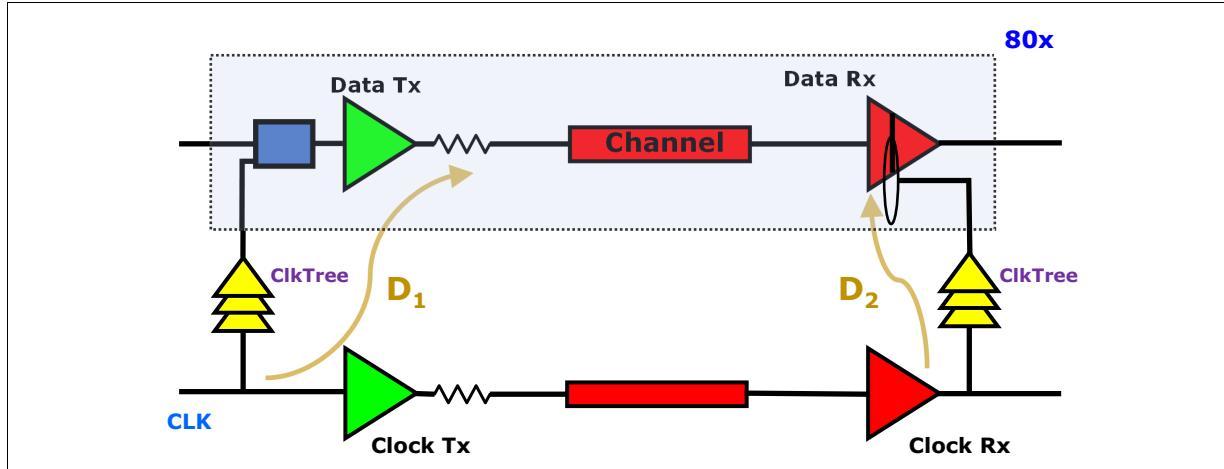
Table 6-1. UCIE-3D Key Performance Indicators

Characteristics/KPIs	UCIE-S	UCIE-A	UCIE-3D	Comments for UCIE-3D
Characteristics				
Data Rate (GT/s)	4, 8, 12, 16, 24, 32	up to 4		<ul style="list-style-type: none"> • Equal to SoC Logic Frequency — power efficiency is critical.
Width (each cluster)	16	64	80	<ul style="list-style-type: none"> • Options of reduced width to 70, 60,
Bump Pitch (um)	100 to 130	25 to 55	≤ 10 (optimized) > 10 to 25 (functional)	<ul style="list-style-type: none"> • Must scale such that UCIE-3D fits with the bump area. • Must support hybrid bonding.
Channel Reach (mm)	≤ 25	≤ 2	3D vertical	<ul style="list-style-type: none"> • F2F bonding initially; F2B, B2B, multi-stack possible.
Target for Key Metrics				
BW Die Edge (GB/s/mm)	28 to 224	165 to 1,317	N/A (vertical)	
BW Density (GB/s/mm ²)	22 to 125	188 to 1,350	4,000 at 9 um	<ul style="list-style-type: none"> • 4 TB/s/mm² at 9 um • Approximately 12 TB/s/mm² at 5 um • Approximately 35 TB/s/mm² at 3 um • Approximately 300 TB/s/mm² at 1 um
Power Efficiency Target (pJ/b)	0.50	0.25	< 0.05 at 9 um	<ul style="list-style-type: none"> • Conservatively estimated at 9-um pitch. • < 0.02 for 3-um pitch.
Low-power Entry/Exit	0.5 ns at ≤ 16 GT/s 0.5 ns to 1 ns at ≥ 24 GT/s		0 ns	<ul style="list-style-type: none"> • No preamble or postamble.
Latency (Tx + Rx)	< 2 ns (PHY + Adapter)		0.125 ns at 4 GT/s	<ul style="list-style-type: none"> • 0.5 UI, half of flop to flop.
Reliability (FIT)	$0 < \text{FIT} \ll 1$			<ul style="list-style-type: none"> • BER < 1E-27.
ESD	30-V CDM		5-V CDM $\rightarrow \leq 3V$	<ul style="list-style-type: none"> • 5-V CDM at introduction. No ESD for wafer-to-wafer hybrid bonding possible.

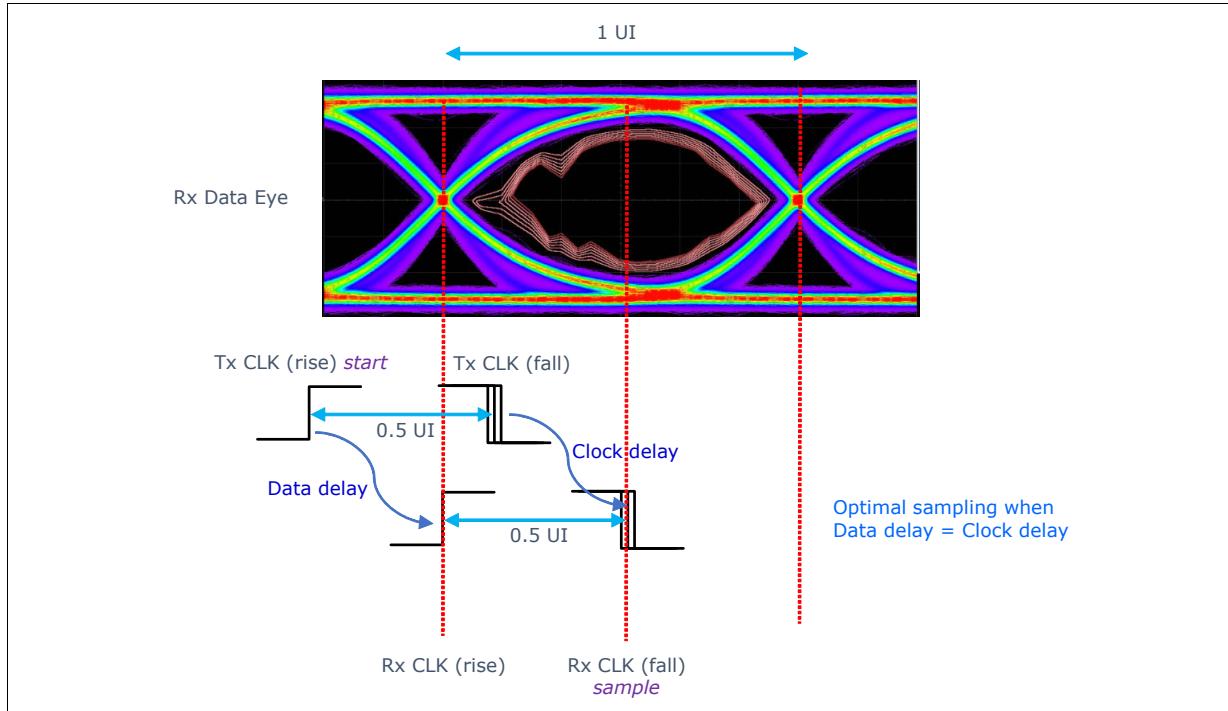
6.3 UCIE-3D Tx, Rx, and Clocking

Figure 6-3 presents the Transceiver (Trx) architecture of UCIE-3D. A matched architecture as shown in [Figure 5-4](#), [Figure 5-8](#), and [Figure 5-13](#) offers optimal supply noise rejection. However, this comes at the cost of increased power consumption. The architecture depicted in [Figure 6-3](#) circumvents this power penalty while maintaining the same level of supply noise rejection. The UCIE-3D specification will establish target values and tolerances for clock distribution delays D_1 and D_2 .

Figure 6-3. UCIE-3D PHY



It is important to highlight that UCIE-3D uses a rise-to-fall timing approach, differing from the typical on-die logic design that uses a rise-to-rise timing approach. The primary distinction between these two scenarios is that on-die logic must factor in the delay caused by combinational logic, whereas UCIE-3D features matched data and clock buffer delays, resulting in a near-zero differential. As depicted in [Figure 6-4](#), rise-to-fall timing yields the most-optimal timing margin for zero-delay differential.

Figure 6-4. Start Edge and Sample Edge

6.4 Electrical Specification

6.4.1 Timing Budget

Consideration of various factors such as jitter, noise, mismatch, and error terms are crucial for the link timing budget. [Table 6-2](#) outlines the UCIE-3D specification parameters that are pertinent to link timing. Pulse width deviation from 50% clock period includes both static error (duty-cycle error) and dynamic error (pulse-width jitter). Lane-to-lane skews account for the variation between data lanes, and Data/Clock differential delays account for the clock to center of distribution of data lanes.

Table 6-2. Timing and Mismatch Specification (Sheet 1 of 2)

Specification	Name	Min	Typ	Max	Unit	UI = 250 ps at 4 GT/s	Note
Eye Closure due to Channel	C_h		0.1		UI	25 ps	a
Pulse-width Deviation from 50% Clock Period	J_{pw}		0.08		UI pk-to-pk	20 ps	
Tx Lane-to-Lane Skew	S_{tx}		0.12		UI pk-to-pk	30 ps	
Rx Lane-to-Lane Skew	S_{rx}		0.12		UI pk-to-pk	30 ps	
Tx Data/Clock Differential Delay	D_{tx}	D_{tx_min}	D_{tx_typ}	D_{tx_max}	ps	max – min = 50 ps	b
Rx Data/Clock Differential Delay	D_{rx}	D_{rx_min}	D_{rx_typ}	D_{rx_max}	ps	max – min = 50 ps	
Alpha Factor (Tx and Rx)	α_{trx}			1.5			c
Vcc Noise	n_{vcc}			10	% pk-to-pk		d

Table 6-2. Timing and Mismatch Specification (Sheet 2 of 2)

Specification	Name	Min	Typ	Max	Unit	UI = 250 ps at 4 GT/s	Note
Tx Data/Clock Differential RJ	J _{rtx}			0.05	UI pk-to-pk at BER	12.5 ps	
Rx Data/Clock Differential RJ	J _{rrx}			0.05	UI pk-to-pk at BER	12.5 ps	
Sampling Aperture	A _p			0.03	UI	7.5 ps	

- a. Eye closure due to channel includes inter-symbol interference (ISI) and crosstalk.
b. Defined as clock to mean data, min/typ/max values are shown below.

c. Alpha factor is defined as follows for Tx and Rx, respectively:

$$\alpha_{Tx} = \frac{dD_{Tx}}{D_{Tx}} / \frac{dV_{CC}}{V_{CC}} \quad \alpha_{Rx} = \frac{dD_{Rx}}{D_{Rx}} / \frac{dV_{CC}}{V_{CC}}$$

d. This is equivalent to a variation of $\pm 5\%$ in V_{CC}. Careful mitigation is particularly needed when disturbances external to UCIE occur, such as electromagnetic coupling from through-silicon vias (TSVs).

Parameters D_{tx} and D_{rx} are V_{CC}-dependent functions. [Equation 6-1](#) defines their typical values.

Equation 6-1.

$$D_{tx_typ} = D_{rx_typ} = \frac{V_{CC}}{0.0153 V_{CC}^2 + 0.0188 V_{CC} - 0.0084}$$

where, unit of D_{tx_typ} and D_{rx_typ} is ps and unit of V_{CC} is V.

[Equation 6-2](#) and [Equation 6-3](#) define the minimum spec curve of D_{tx} and D_{rx}, respectively.

Equation 6-2.

$$D_{tx_min} = \max(D_{tx_typ} - 0.08 UI, 0)$$

Equation 6-3.

$$D_{rx_min} = \max(D_{rx_typ} - 0.08 UI, 0)$$

[Equation 6-4](#) and [Equation 6-5](#) define the maximum spec curve of D_{tx} and D_{rx}, respectively.

Equation 6-4.

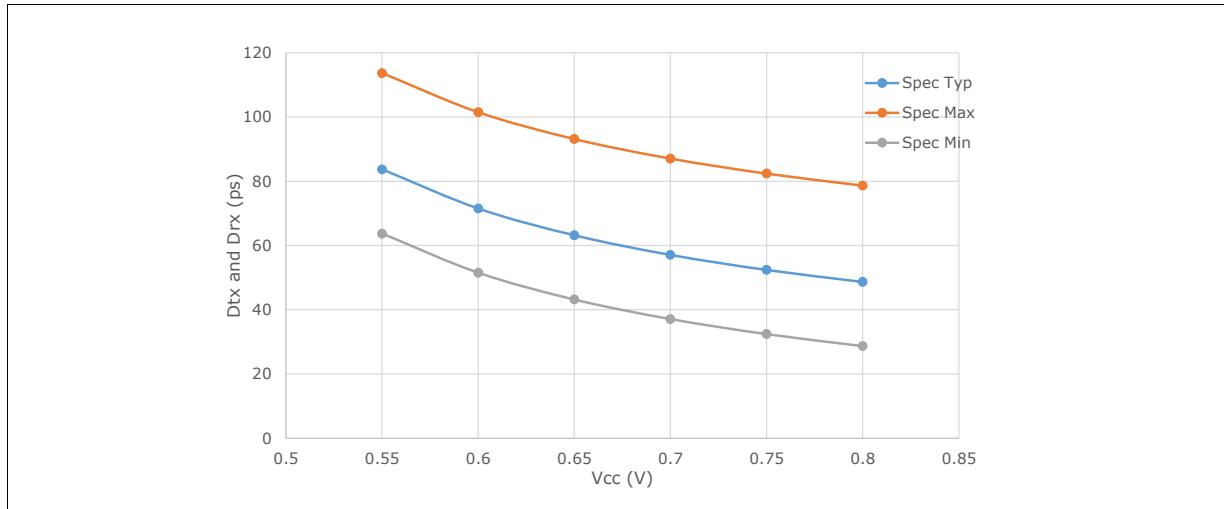
$$D_{tx_max} = D_{tx_typ} + 0.12 UI$$

Equation 6-5.

$$D_{rx_max} = D_{rx_typ} + 0.12 UI$$

Figure 6-5 illustrates a plot of the spec range for 4 GT/s.

Figure 6-5. Dtx and Drx Spec Range for 4 GT/s



The equation for delay time, derived from the general theory of buffer chain, incorporates a term proportional to Vcc and a quadratic Vcc dependence in the denominator. This equation is fitted to a specific process and design. A typical design is expected to have the same trend, and remain within the boundaries of the upper and lower curves. It is not required to align with the central curve.

Equation 6-6 is essential in closing the timing budget, subsequently leading to the defined specification limit.

Equation 6-6.

$$C_h + J_{pw} + S_{tx} + S_{rx} + \sqrt{J_{rtx}^2 + J_{rrx}^2} + A_p \\ + [max(D_{tx}) - min(D_{tx}) + max(D_{rx}) - min(D_{rx})] (1 + \alpha_{tx} n_{vcc}) < 1 UI$$

When there is a change in Vcc, as in the case of a dynamic voltage frequency scaling (DVFS) scenario, the specification range for D_{tx} and D_{rx} adjusts correspondingly. This offers a degree of design flexibility because the delay does not need to conform to a fixed band across the entire Vcc range. Given that the range from maximum to minimum remains constant, the timing margin remains unaffected.

6.4.2 ESD and Energy Efficiency

Data and clock signals shall comply with a mask on an eye diagram that specifies the following:

- Minimum voltage swing
- Minimum duration during which the output voltage will be stable
- Maximum permitted overshoot and undershoot

The Tx output swing range is between 0.40 V and 0.75 V.

Table 6-3 defines the ESD targets.

Table 6-3. ESD Specification for ≤ 10 um Bump Pitch

Parameter	Minimum
Discharge voltage (CDM)	5 V
Discharge peak current	40 mA

The feasibility of 0-V ESD should be explored for the special case of wafer-to-wafer hybrid bonding. For more details, see the *Industry Council on ESD Targets white papers*.

For > 10 um to < 25 um bump pitches, higher ESD can be permitted. The exact target will be published in a future revision of the specification.

Table 6-4 lists the Energy Efficiency targets.

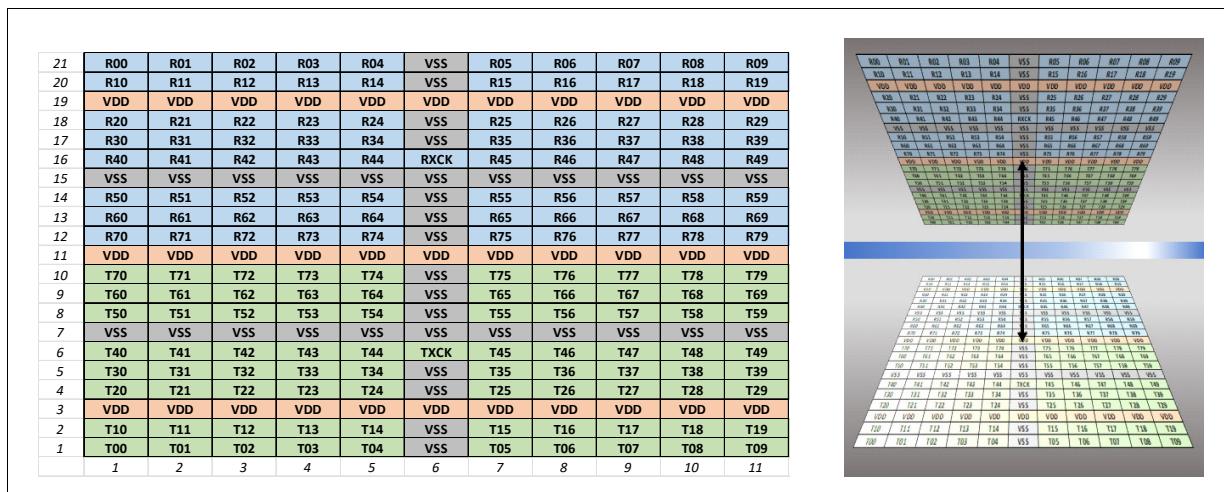
Table 6-4. Energy Efficiency Target

Bump Spacing (um)	Energy Efficiency at 4 GT/s (pJ/bit)
9	0.05
3	0.02
1	0.01
9 to 25	To be published in a future revision of the specification.

6.4.3 UCIE-3D Module and Bump Map

Figure 6-6 depicts a potential bump map for UCIE-3D. The arrangement of the signals is such that the same PHY can be utilized on both the top die and bottom die. The unit used in Figure 6-6 is the bump pitch. The estimated area for a x80 module (encompassing both Tx and Rx) in a 9-um pitch is approximately 0.02 mm². It is important to note that the area scales with the square of the bump pitch.

Figure 6-6. UCIE-3D Module Bump Map



The UCIE-3D standard does not prescribe a mandatory bump pitch; however, a 9-um pitch is recommended at introduction. As the technology advances, additional specific recommended pitch values will be established.

Although UCIE-3D does not inherently predefine an adapter, users have the flexibility to allocate some data lanes within the module for adapter functions as required, such as Valid, Data Mask, Parity, and ECC. UCIE-3D does not necessitate a sideband for initialization. If a low-bandwidth data link similar to sideband is required, it is up to the implementation to determine how to assign a group of lanes for the purpose. Bit replication or other forms of redundancy can be used to guarantee link reliability.

If modules are physically adjacent, extra VDDs can be added between them to provide physical separation, shielding, and additional power delivery.

Along with x80, the bump map of x70 Module is depicted in [Figure 6-7](#). Bump maps of additional Module widths may be incorporated in a future update to this specification if needed, using similar layout.

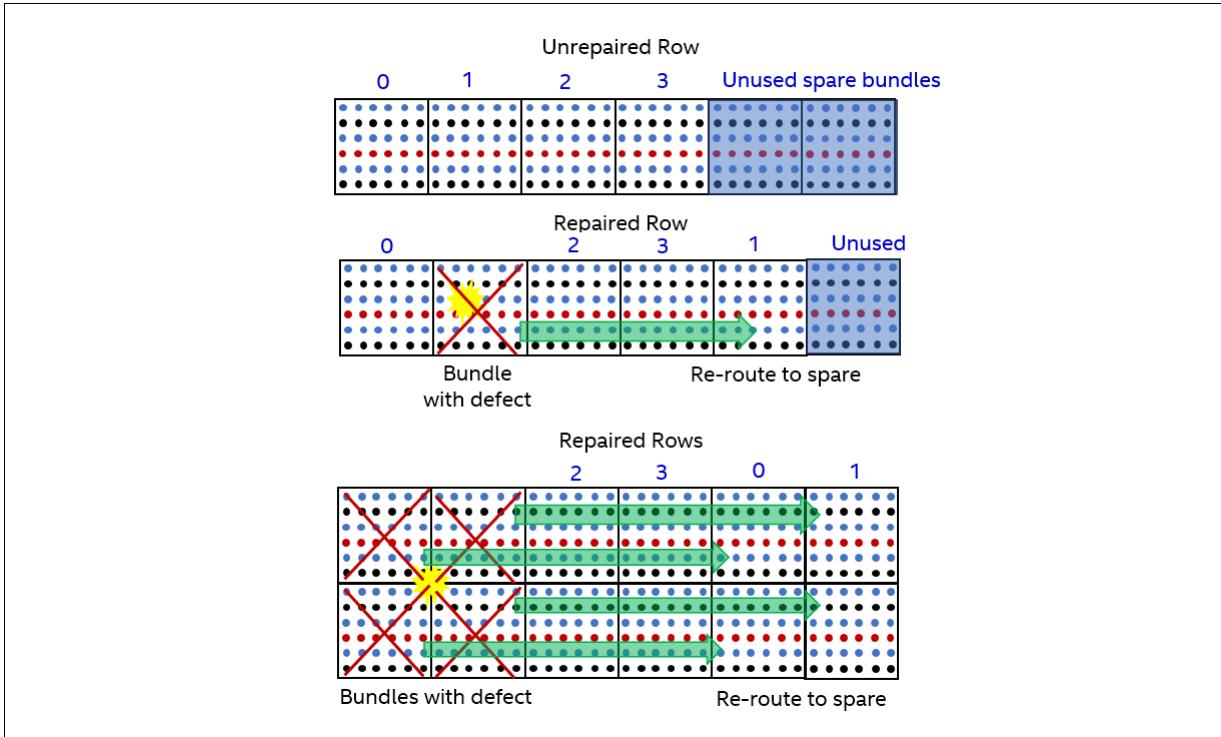
Figure 6-7. x70 Module

19	R00	R01	R02	R03	R04	VSS	R05	R06	R07	R08	R09
18	R10	R11	R12	R13	R14	VSS	R15	R16	R17	R18	R19
17	VDD	VDD	VDD	VDD	VDD	VDD	VDD	VDD	VDD	VDD	VDD
16	R20	R21	R22	R23	R24	VSS	R25	R26	R27	R28	R29
15	R30	R31	R32	R33	R34	VSS	R35	R36	R37	R38	R39
14	R40	R41	R42	R43	R44	RXCK	R45	R46	R47	R48	R49
13	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS
12	R50	R51	R52	R53	R54	VSS	R55	R56	R57	R58	R59
11	R60	R61	R62	R63	R64	VSS	R65	R66	R67	R68	R69
10	VDD	VDD	VDD	VDD	VDD	VDD	VDD	VDD	VDD	VDD	VDD
9	T60	T61	T62	T63	T64	VSS	T65	T66	T67	T68	T69
8	T50	T51	T52	T53	T54	VSS	T55	T56	T57	T58	T59
7	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS	VSS
6	T40	T41	T42	T43	T44	TXCK	T45	T46	T47	T48	T49
5	T30	T31	T32	T33	T34	VSS	T35	T36	T37	T38	T39
4	T20	T21	T22	T23	T24	VSS	T25	T26	T27	T28	T29
3	VDD	VDD	VDD	VDD	VDD	VDD	VDD	VDD	VDD	VDD	VDD
2	T10	T11	T12	T13	T14	VSS	T15	T16	T17	T18	T19
1	T00	T01	T02	T03	T04	VSS	T05	T06	T07	T08	T09
	1	2	3	4	5	6	7	8	9	10	11

6.4.4 Repair Strategy

Defect size (more exactly, Si area impacted by a single defect) is defined by a probability distribution. The size is influenced by factors such as numbers of I/Os in SoC, packaging technology used, and bump pitch. A standard needs to cover technologies from multiple companies, scalable to future bump pitches, as well as different SoC sizes. Lane repair based on fixed defect size is not practical for an effective standard.

Given these considerations, a bundle repair strategy is proposed for UCIE-3D. This involves reserving bundles within the SoC for repair purposes, which can be rerouted to serve as backup in the event of a failure, as illustrated in [Figure 6-8](#). The figure shows the cases of no repair, 1-bundle repair, and 4-bundle repairs. For a densely packed 2D UCIE Module array, it is recommended to reserve two full Modules (equivalent to four bundles) to repair a single failure. This assumes an alternating arrangement of Tx and Rx bundles in at least one direction. Each Module is equipped with one Tx bundle (comprising a x80 Tx + Clock) and one Rx bundle (comprising a x80 Rx + Clock).

Figure 6-8. Bundle Repair

To scale the general case of a large number of UCIE links, the following mathematical model can be used to compute the repair requirements:

Parameters:

- D_0 represents the defect density of the interconnect, expressed in terms of the number of failures per unit area
- A signifies the total UCIE-3D area of the chip
- δ denotes the acceptable yield loss

The model suggests reserving $2k$ full Modules, where k is determined by the subsequent equation.

Equation 6-7.

$$1 - \sum_{i=0}^k P_i(AD_0) < \delta$$

Equation 6-8.

$$P_i(x) = \frac{x^i}{i!} e^{-x}$$

The calculations in [Equation 6-7](#) and [Equation 6-8](#) assume that large interconnect defects that are comparable to bundle size are relatively rare. More spare bundles may be needed if density of large defects exceeds a limit such that [Equation 6-9](#) does not hold.

Equation 6-9.

$$1 - e^{-AD_1} < \delta$$

where, D_1 is the density of defects with diameter greater than the bundle dimension. The exact amount can be determined by simulation.

When UCIE-3D links are not densely packed, strategic placement of spacing between bundles can effectively reduce the number of repair bundles required. For example, with sufficient spacing between rows, the occurrence of a single defect eliminating four bundles can be prevented. However, the precise determination of this spacing is highly dependent on the specific technology in use, and thus, falls beyond the scope of this specification. The specification merely highlights this as a potential option.

The initiation of repair is anticipated to originate from the SoC Logic, which is external to the UCIE-3D PHY, and therefore is not elaborated on in this context. The implementation can be specific to the system.

6.4.5 Channel and Data Rate Extension

While the immediate focus of UCIE-3D is on Face-to-Face hybrid bonding, the proposed architecture is designed to be adaptable for Face-to-Back, Back-to-Back, and multi-stack configurations. Comprehensive channel and circuit simulations are necessary to determine the optimal data rate for these scenarios. Reduction of 10% or less in data rate is expected for Face-to-Back and Back-to-Back configurations.

§ §

7.0 Sideband

7.1 Protocol Specification

The usage for the sideband Link is to provide a out of band channel for Link training and an interface for sideband access of registers of the Link partner. It is also used for Link Management Packets and parameter exchanges with remote Link partner.

The same protocol is also used for local die sideband accesses over FDI and RDI. When relevant, FDI specific rules are pointed out using "FDI sideband:". When relevant, RDI specific rules are pointed out using "RDI sideband:". When relevant, UCIe Link specific rules are pointed out using "UCIe Link sideband:". If no prefix is mentioned, it is a common rule across FDI, RDI and UCIe Link.

The Physical Layer is responsible for framing and transporting sideband packets over the UCIe Link. Direct sideband access to remote die can originate from the Adapter or the Physical Layer. The Adapter forwards a remote die sideband access over RDI to the Physical Layer for framing and transport. These include register access requests, completions or messages.

The Protocol Layer has indirect access to remote die registers using the sideband mailbox mechanism. The mailbox registers reside in the Adapter, and it is the responsibility of the Adapter to initiate remote die register access requests when it receives the corresponding access trigger for the mailbox register over FDI.

FDI sideband: In the case of multi-protocol stacks, the Adapter must track which protocol stack sent the original request and route the completion back to the appropriate protocol stack.

FDI sideband: Because the Protocol Layer is only permitted indirect access to remote die registers, and direct access to local die registers, currently only Register Access requests and completions are permitted on the FDI sideband.

All sideband requests that expect a response have an 8ms timeout. A "Stall" encoding is provided for the relevant packets for Retimers, to prevent timeouts if the Retimer needs extra time to respond to the request. When stalling to prevent timeouts, it is the responsibility of the Retimer to send the corresponding Stall response once every 4ms. The Retimer must also ensure that it does not Stall indefinitely, and escalates a Link down event after a reasonable attempt to complete resolution that required stalling the requester. If a requester receives a response with a "Stall" encoding, it resets the timeout counter.

In certain cases, it is necessary for registers to be fragmented between the different layers; i.e., certain bits of a given register physically reside in the Protocol Layer, other bits reside in the Adapter, and other bits reside in the Physical Layer. UCIe takes a hierarchical decoding for these registers. For fragmented registers, if a bit does not physically reside in a given Layer, it implements that bit as Read Only and tied to 0. Hence reads would return 0 for those bits from that Layer, and writes would have no effect on those bits. As an example, for reads, Protocol Layer would forward these requests to the Adapter on FDI and the Protocol Layer will OR the data responded by the Adapter with its local register before responding to software. The Adapter must do the same if any bits of that register reside in the Physical Layer before responding to the Protocol Layer.

7.1.1 Packet Types

Three different categories of packets are permitted:

- Register Accesses: These can be Configuration (CFG) or Memory Mapped accesses for both Reads or Writes are supported. These can be associated with 32b of data or 64b of data. All register accesses (Reads or Writes) have an associated completion.
- Messages without data: These can be Link Management (LM), or Vendor Defined Packets. These do not carry additional data payloads.
- Messages with data: These can be Parameter Exchange (PE), Link Training related or Vendor Defined, and carry 64b of data.
- Management Transport Messages: If Management Transport protocol is supported, Management Transport Messages with data or without data are supported (see [Section 7.1.2.4](#) and [Section 7.1.2.5](#), respectively).

Every packet carries a 5-bit opcode, a 3-bit source identifier (srcid), and a 3-bit destination identifier (dstid). The 5-bit opcode indicates the packet type, as well as whether the packet carries no data, 32b of data or 64b of data.

[Table 7-1](#) gives the mapping of opcode encodings to Packet Types.

Table 7-1. Opcode Encodings Mapped to Packet Types

Opcode Encoding	Packet Type
00000b	32b Memory Read
00001b	32b Memory Write
00010b	32b DMS Register Read
00011b	32b DMS Register Write
00100b	32b Configuration Read
00101b	32b Configuration Write
01000b	64b Memory Read
01001b	64b Memory Write
01010b	64b DMS Register Read
01011b	64b DMS Register Write
01100b	64b Configuration Read
01101b	64b Configuration Write
10000b	Completion without Data
10001b	Completion with 32b Data
10010b	Message without Data
10111b	Management Port Messages without Data
11000b	Management Port Message with Data
11001b	Completion with 64b Data
11011b	Message with 64b Data
Other encodings	Reserved

[Table 7-2](#), [Table 7-3](#), and [Table 7-4](#) give the encodings of source and destination identifiers. It is not permitted for Protocol Layer from one side of the Link to directly access Protocol Layer of the remote Link partner over sideband (this should be done via mainband).

Table 7-2. FDI sideband: srcid and dstid encodings on FDI

Field ^a	Description
srcid[2:0]	000b: Stack 0 Protocol Layer 100b: Stack 1 Protocol Layer other encodings are reserved.
dstid[2:0]	001b: D2D Adapter 010b: Physical Layer other encodings are reserved.

- a. srcid and dstid are Reserved for completion messages transferred over FDI. The Protocol Layer must correlate the completions to original requests using the Tag field. Currently, no requests are permitted from Adapter to Protocol Layer over FDI sideband.

Table 7-3. RDI sideband: srcid and dstid encodings on RDI

Field ^a	Description
srcid[2:0]	000b: Stack 0 Protocol Layer 001b: D2D Adapter 011b: Management Port Gateway (see Section 8.2) 100b: Stack 1 Protocol Layer other encodings are reserved.
dstid[2]	0b: Local die terminated request 1b: Remote die terminated request
dstid[1:0]	For Local die terminated requests: 10b: Physical Layer other encodings are reserved. For Remote die terminated Register Access Requests: dstid[1:0] is Reserved For Remote die terminated Register Access Completions: 01b: D2D Adapter other encodings are reserved. For Remote die terminated messages: 01b: D2D Adapter message 10b: Physical Layer message 11b: Management Port Gateway message (see Section 8.2)

- a. srcid and dstid are Reserved for completion messages transferred over RDI for local Register Access completions. For Register Access completions, the Adapter must correlate the completions to original requests using the Tag field regardless of dstid field. Both local and remote Register Access requests are mastered by the Adapter with unique Tag encodings.

Table 7-4. UCIe Link sideband: srcid and dstid encodings for UCIe Link

Field	Description
srcid[2:0]	001b: D2D Adapter 010b: Physical Layer 011b: Management Port Gateway (see Section 8.2) other encodings are reserved
dstid[2]	1b: Remote die terminated request other encodings are reserved
dstid[1:0]	For Register Access requests: dstid[1:0] is Reserved. For Remote die terminated Register Access Completions: 01b: D2D Adapter other encodings are reserved. For Remote die terminated messages: 01b: D2D Adapter message 10b: Physical Layer message 11b: Management Port Gateway message (see Section 8.2)

7.1.2 Packet Formats

All the figures in this section show examples assuming a 32-bit interface of RDI/FDI transfer for sideband packets, hence the headers and data are shown in Phases of 32 bits.

Note that the sideband packet format figures provided in this chapter show the packet format over multiple 32-bit Phases. This is for representation purposes only. For transport over the UCIe sideband bumps (serial interface), the transfer occurs as a 64-bit serial packet at a time. For headers, the transmission order is bit 0 of Phase 0 as bit 0 of the serial packet (D0 in [Figure 4-8](#)), bit 1 of Phase 0 as bit 1 of the serial packet, etc., followed by bit 0 of Phase 1 as bit 32 of the serial packet, bit 1 of Phase 1 as bit 33 of the serial packet, etc., until bit 31 of Phase 1 as bit 63 of the serial packet.

Data (if present) is sent as a subsequent serial packet, with bit 0 of Phase 2 as bit 0 of the serial packet (D0 in [Figure 4-8](#)), bit 1 of Phase 2 as bit 1 of the serial packet, etc., followed by bit 0 of Phase 3 as bit 32 of the serial packet, bit 1 of Phase 3 as bit 33 of the serial packet, etc., until bit 31 of Phase 3 as bit 63 of the serial packet.

7.1.2.1 Register Access Packets

[Figure 7-1](#) shows the packet format for Register Access requests. [Table 7-5](#) gives the description of the fields other than the opcode, srcid, and dstid.

Table 7-5. Field descriptions for Register Access Requests

Field	Description
CP	Control Parity (CP) is the even parity of all the header bits excluding DP.
DP	Data Parity is the even parity of all bits in the data payload. If there is no data payload, this bit is set to 0b.
Cr	If 1b, indicates one credit return for credited sideband messages. This field is only used by the Adapter for remote Link partner's credit returns for E2E credits. It is not used for local FDI or RDI credit loops.
Addr[23 : 0]	Address of the request. Different opcodes use this field differently. See Table 7-6 for details. The following rules apply for the address field: For 64-bit request, Addr[2:0] is reserved. For 32-bit request, Addr[1:0] is reserved.
BE[7 : 0]	Byte Enables for the Request. It is NOT required to be contiguous. BE[7:4] are reserved if the opcode is for a 32-bit request.
EP	Data Poison. If poison forwarding is enabled, the completer can poison the data on internal errors. Setting the EP bit is optional, the conditions for setting it to 1 are implementation-specific. Typical usages involve giving additional FIT protection against data integrity errors on internal data buffers. A Receiver must not modify the contents of the target location for requests with data payload that have the EP bit set. It must return UR for the completion status of requests with an EP bit set.
Tag[4 : 0]	Tag is a 5-bit field generated by the requester, and it must be unique for all outstanding requests that require a completion. The original requester uses the Tag to associate returning completions with the original request.
Data	Payload. Can be 32 bits or 64 bits wide depending on the Opcode.

Table 7-6. Mapping of Addr[23:0] for Different Requests

Opcode	Description
Memory Reads/Writes	<p>{RL[3:0], Offset[19:0]}</p> <p>Offset is the Byte Offset.</p> <p>RL[3:0] encodings are as follows:</p> <ul style="list-style-type: none"> 0h: Register Locator 0 1h: Register Locator 1 2h: Register Locator 2 3h: Register Locator 3 <p>Fh: Accesses for Protocol specific MMIO registers that are shadowed in the Adapter (e.g., ARB/MUX registers defined in the <i>CXL Specification</i>). The offsets for these registers are implementation specific, and the protocol layer must translate accesses to match the offsets implemented in the Adapter.</p> <p>Other encodings are reserved.</p> <p>For accesses to Reserved RL encodings, the completer must respond with a UR.</p>
Configuration Reads/Writes	<p>{RL[3:0], Rsvd[7:0], Byte Offset[11:0]}, where</p> <p>RL[3:0] encodings are as follows:</p> <ul style="list-style-type: none"> 0h: UCIE Link DVSEC <p>Fh: Accesses for Protocol specific configuration registers that are shadowed in the Adapter (e.g., ARB/MUX registers defined in the <i>CXL Specification</i>). The offsets for these registers are implementation specific, and the protocol layer must translate accesses to match the offsets implemented in the Adapter.</p> <p>Other encodings are reserved.</p> <p>For accesses to Reserved RL encodings, the completer must respond with a UR.</p>
DMS Register Reads/Writes	<p>These allow for accessing the DMS registers implemented in UCIE Spoke Type 0, 1, or 2.</p> <p>Addr[21:0] provides the register offset in DMS register space, relative to the start of the Spoke's register space, that corresponds to the DevID. A maximum of 4 MB of address space is possible for UCIE D2D/PHY Spokes. These opcodes are always targeted at the local D2D or PHY registers (i.e., these opcodes never target the remote link partner).</p> <p>Addr[23:22] encodings are as follows:</p> <ul style="list-style-type: none"> 00b: Spoke registers. 01b: Reserved. 10b: Reserved. 11b: Used for other chiplet UMAP registers that are shadowed in the D2D or PHY, if any. The definitions of these registers and offsets are implementation-specific.

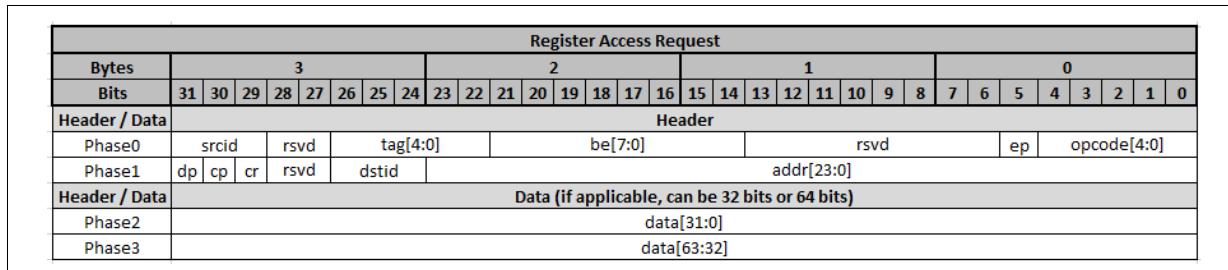
Figure 7-1. Format for Register Access Request

Figure 7-2 gives the format for Register Access completions.

Figure 7-2. Format for Register Access Completions

Register Access Completions																																																											
Bytes	3							2							1							0																																					
Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																											
Header / Data	Header																																																										
Phase0	srcid			rsvd			tag[4:0]				be[7:0]					rsvd			ep	opcode[4:0]						Status																																	
Phase1	dp	cp	cr	rsvd			dstid			rsvd														Status																																			
Header / Data	Data (if completion with data, can be 32 bits or 64 bits)																																																										
Phase2	data[31:0]																																																										
Phase3	data[63:32]																																																										

Table 7-7 gives the field descriptions for a completion.

Table 7-7. Field Descriptions for a Completion

Field	Description
Tag [4:0]	Completion Tag associated with the corresponding Request. The requester uses this to associate the completion with the original request.
CP	Control Parity. All fields other than "DP" and "CP" in the Header are protected by Control Parity, and the parity scheme is even (including reserved bits).
DP	Data Parity. All fields in data are protected by data parity, and the parity scheme is even.
Cr	If 1b, indicates one credit return for credited sideband messages. This field is only used by the Adapter for remote Link partner's credit returns for E2E credits. It is not used for local FDI or RDI credit loops.
EP	Data Poison. If poison forwarding is enabled, the completer can poison the data on internal errors. Setting the EP bit is optional, the conditions for setting it to 1 are implementation-specific. Typical usages involve giving additional FIT protection against data integrity errors on internal data buffers. A Receiver must not modify the contents of the target location for requests with data payload that have the EP bit set. It must return UR for the completion status of requests with an EP bit set.
BE [7:0]	Byte Enables for the Request. Completer returns the same value that the original request had (this avoids the requester from having to save off the BE value). BE[7:4] are reserved if the opcode is for a 32-bit request.
Status [2:0]	Completion Status 000b - Successful Completion (SC). This can be a completion with or without data, depending on the original request (it must set the appropriate Opcode). If the original request was a write, it is a completion without data. If the original request was a read, it is a completion with data. 001b - Unsupported Request (UR). On UCIe, this is a completion with 64b Data when a request is aborted by the completer, and the Data carries the original request header that resulted in UR. This enables easier header logging at the requester. Register Access requests that timeout must also return UR status, but for those the completion is without Data. 100b - Completer Abort (CA). On UCIe, this is a completion with 64b Data, and the Data carries original request header that resulted in UR. This enables easier header logging at the requester. 111b - Stall. Receiving a completion with Stall encoding must reset the timeout at the requester. Completer must send a Stall once every 4ms if it is not ready to respond to the original request. Other encodings are reserved. An error is logged in the Sideband Mailbox Status if a CA was received or if the number of timeouts exceed the programmed threshold. For timeouts below the programmed threshold, a UR is returned to the requester.
Data	Payload. 32 bits or 64 bits depending on the Opcode.

7.1.2.2 Messages without Data

Figure 7-3 shows the Format for Messages without data payloads. These can be Link Management Packets, NOPs or Vendor Defined message packets.

Figure 7-3. Format for Messages without Data

Messages without Data																																
Bytes	3							2							1							0										
Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Header / Data	Header																															
Phase0	srcid	rsvd	rsvd			msgcode[7:0]				rsvd				opcode[4:0]							MsgSubcode[7:0]											
Phase1	dp	cp	rsvd	dstid		MsgInfo[15:0]																										

The definitions of opcode, srcid, dstid, dp, and cp fields are the same as Register Access packets.

[Table 7-8](#) and [Table 7-9](#) give the encodings of the different messages without data that are send on UCIe. Some Notes on the different message categories are listed below:

- {NOP.Crd} — These are used for E2E Credit returns. The destination must be D2D Adapter.
- {LinkMgmt.RDI.*} — These are used to coordinate RDI state transitions, the source and destination is Physical Layer.
- {LinkMgmt.Adapter0.*} — These are used to coordinate Adapter LSM state transitions for the Adapter LSM corresponding to Stack 0 Protocol Layer. The source and destination is D2D Adapter.
- {LinkMgmt.Adapter1.*} — These are used to coordinate Adapter LSM state transitions for the Adapter LSM corresponding to Stack 1 Protocol Layer. The source and destination is D2D Adapter.
- {ParityFeature.*} — This is used to coordinate enabling of the Parity insertion feature. The source and destination for this must be the D2D Adapter.
- {ErrMsg} — This is used for error reporting and escalation from the remote Link Partner. This is sent from the Retimer or Device die to the Host, and the destination must be the D2D Adapter.

Table 7-8. Message Encodings for Messages without Data (Sheet 1 of 3)

Name	Msgcode	Msgsubcode	MsgInfo	Description
{Nop.Crd}	00h	00h	0000h:Reserved 0001h:1 Credit return 0002h: 2 Credit returns 0003h: 3 Credit returns 0004h: 4 Credit returns	Explicit Credit return from Remote Link partner for credited messages.
{LinkMgmt.RDI.Req.Active}	01h	01h	Reserved	Active Request for RDI SM.
{LinkMgmt.RDI.Req.L1}		04h		L1 Request for RDI SM.
{LinkMgmt.RDI.Req.L2}		08h		L2 Request for RDI SM.
{LinkMgmt.RDI.Req.LinkReset}		09h		LinkReset Request for RDI SM.
{LinkMgmt.RDI.Req.LinkError}		0Ah		LinkError Request for RDI SM.
{LinkMgmt.RDI.Req.Retrain}		0Bh		Retrain Request for RDI SM.
{LinkMgmt.RDI.Req.Disable}		0Ch		Disable Request for RDI SM.

Table 7-8. Message Encodings for Messages without Data (Sheet 2 of 3)

Name	Msgcode	Msgsubcode	MsgInfo	Description
{LinkMgmt.RDI.Rsp.Active}	02h	01h	0000h: Regular Response FFFFh: Stall Response	Active Response for RDI SM.
{LinkMgmt.RDI.Rsp.PMNAK}		02h		PMNAK Response for RDI SM
{LinkMgmt.RDI.Rsp.L1}		04h		L1 Response for RDI SM.
{LinkMgmt.RDI.Rsp.L2}		08h		L2 Response for RDI SM.
{LinkMgmt.RDI.Rsp.LinkReset}		09h		LinkReset Response for RDI SM.
{LinkMgmt.RDI.Rsp.LinkError}		0Ah		LinkError Response for RDI SM.
{LinkMgmt.RDI.Rsp.Retrain}		0Bh		Retrain Response for RDI SM.
{LinkMgmt.RDI.Rsp.Disable}		0Ch		Disable Response for RDI SM.
{LinkMgmt.Adapter.0.Req.Active}	03h	01h	0000h: Regular Request FFFFh: Stall	Active Request for Stack 0 Adapter LSM. The Stall encoding is provided for Retimers to avoid the Adapter LSM transition to Active timeout as described in Section 9.5.3.8 .
{LinkMgmt.Adapter.0.Req.L1}		04h	Reserved	L1 Request for Stack 0 Adapter LSM.
{LinkMgmt.Adapter.0.Req.L2}		08h		L2 Request for Stack 0 Adapter LSM.
{LinkMgmt.Adapter.0.Req.LinkReset}		09h		LinkReset Request for Stack 0 Adapter LSM.
{LinkMgmt.Adapter.0.Req.Disable}		0Ch		Disable Request for Stack 0 Adapter LSM.
{LinkMgmt.Adapter.0.Rsp.Active}	04h	01h	0000h: Regular Response FFFFh: Stall Response	Active Response for Stack 0 Adapter LSM.
{LinkMgmt.Adapter.0.Rsp.PMNAK}		02h		PMNAK Response for Stack 0 Adapter LSM.
{LinkMgmt.Adapter.0.Rsp.L1}		04h		L1 Response for Stack 0 Adapter LSM.
{LinkMgmt.Adapter.0.Rsp.L2}		08h		L2 Response for Stack 0 Adapter LSM.
{LinkMgmt.Adapter.0.Rsp.LinkReset}		09h		LinkReset Response for Stack 0 Adapter LSM.
{LinkMgmt.Adapter.0.Rsp.Disable}		0Ch		Disable Response for Stack 0 Adapter LSM.

Table 7-8. Message Encodings for Messages without Data (Sheet 3 of 3)

Name	Msgcode	Msgsubcode	MsgInfo	Description
{LinkMgmt.Adapter 1.Req.Active}	05h	01h	0000h: Regular Request FFFFh: Stall	Active Request for Stack 1 Adapter LSM. The Stall encoding is provided for Retimers to avoid the Adapter LSM transition to Active timeout as described in Section 9.5.3.8 .
{LinkMgmt.Adapter 1.Req.L1}		04h	Reserved	L1 Request for Stack 1 Adapter LSM.
{LinkMgmt.Adapter 1.Req.L2}		08h		L2 Request for Stack 1 Adapter LSM.
{LinkMgmt.Adapter 1.Req.LinkReset}		09h		LinkReset Request for Stack 1 Adapter LSM.
{LinkMgmt.Adapter 1.Req.Disable}		0Ch		Disable Request for Stack 1 Adapter LSM.
{LinkMgmt.Adapter 1.Rsp.Active}	06h	01h	0000h: Regular Response FFFFh: Stall Response	Active Response for Stack 1 Adapter LSM.
{LinkMgmt.Adapter 1.Rsp.PMNAK}		02h		PMNAK Response for Stack 1 Adapter LSM
{LinkMgmt.Adapter 1.Rsp.L1}		04h		L1 Response for Stack 1 Adapter LSM.
{LinkMgmt.Adapter 1.Rsp.L2}		08h		L2 Response for Stack 1 Adapter LSM.
{LinkMgmt.Adapter 1.Rsp.LinkReset}		09h		LinkReset Response for Stack 1 Adapter LSM.
{LinkMgmt.Adapter 1.Rsp.Disable}		0Ch		Disable Response for Stack 1 Adapter LSM.
{ParityFeature.Req}	07h	00h	Reserved	Parity Feature enable request.
{ParityFeature.Ack}	08h	00h	0000h: Regular Response	Parity Feature enable Ack.
{ParityFeature.Nak}		01h	FFFFh: Stall Response	Parity Feature enable Nak.
{ErrMsg}	09h	00h	Reserved	Correctable Error Message.
		01h		Non-Fatal Error Message.
		02h		Fatal Error Message.
{Vendor Defined Message}	FFh	--	Vendor ID	<p>Vendor Defined Messages. These can be exchanged at any time after sideband is functional post SBINIT. Interoperability is vendor defined. Unsupported vendor defined messages must be discarded by the receiver.</p> <p>Note that this is NOT the UCIE Vendor ID, but rather the unique identifier of the chiplet vendor that is defining and using these messages.</p>
All other encodings not mentioned in this table are reserved.				

Table 7-9. Link Training State Machine related Message encodings for messages without data (Sheet 1 of 4)

Message	MsgInfo[15:0]	MsgCode[7:0]	MsgSubcode[7:0]
{Start Tx Init D to C point test resp}	0000h	8Ah	01h
{LFSR_clear_error req}	0000h	85h	02h
{LFSR_clear_error resp}	0000h	8Ah	02h
{Tx Init D to C results req}	0000h	85h	03h
{End Tx Init D to C point test req}	0000h	85h	04h
{End Tx Init D to C point test resp}	0000h	8Ah	04h
{Start Tx Init D to C eye sweep resp}	0000h	8Ah	05h
{End Tx Init D to C eye sweep req}	0000h	85h	06h
{End Tx Init D to C eye sweep resp}	0000h	8Ah	06h
{Start Rx Init D to C point test resp}	0000h	8Ah	07h
{Rx Init D to C Tx Count Done req}	0000h	85h	08h
{Rx Init D to C Tx Count Done resp}	0000h	8Ah	08h
{End Rx Init D to C point test req}	0000h	85h	09h
{End Rx Init D to C point test resp}	0000h	8Ah	09h
{Start Rx Init D to C eye sweep resp}	0000h	8Ah	0Ah
{Rx Init D to C results req}	0000h	85h	0Bh
{End Rx Init D to C eye sweep req}	0000h	85h	0Dh
{End Rx Init D to C eye sweep resp}	0000h	8Ah	0Dh
{SBINIT out of Reset}	[15:4] : Reserved [3:0] : Result ^a	91h	00h
{SBINIT done req}	0000h	95h	01h
{SBINIT done resp}	0000h	9Ah	01h
{MBINIT.CAL Done req}	0000h	A5h	02h
{MBINIT.CAL Done resp}	0000h	AAh	02h
{MBINIT.REPAIRCLK init req}	0000h	A5h	03h
{MBINIT.REPAIRCLK init resp}	0000h	AAh	03h
{MBINIT.REPAIRCLK result req}	0000h	A5h	04h
{MBINIT.REPAIRCLK result resp}	[15:4]: Reserved [3]: Compare Results from RRDCK_L [2]: Compare Results from RTRK_L [1]: Compare Results from RCKN_L [0]: Compare Results from RCKP_L	AAh	04h
{MBINIT.REPAIRCLK apply repair req}	[15:4]: Reserved [3:0]: Repair Encoding Fh: Reserved 0h: Repair RCLKP_L 1h: Repair RCLKN_L 2h: Repair RTRK_L 7h: Reserved	A5h	05h
{MBINIT.REPAIRCLK apply repair resp}	0000h	AAh	05h

Table 7-9. Link Training State Machine related Message encodings for messages without data (Sheet 2 of 4)

Message	MsgInfo[15:0]	MsgCode[7:0]	MsgSubcode[7:0]
{MBINIT.REPAIRCLK check repair init req}	0000h	A5h	06h
{MBINIT.REPAIRCLK check repair init resp}	0000h	AAh	06h
{MBINIT.REPAIRCLK check results req}	0000h	A5h	07h
{MBINIT.REPAIRCLK check results resp}	[15:4]: Reserved [3]: Compare Results from RRDCK_L [2]: Compare Results from RTRK_L [1]: Compare Results from RCKN_L [0]: Compare Results from RCKP_L	AAh	07h
{MBINIT.REPAIRCLK done req}	0000h	A5h	08h
{MBINIT.REPAIRCLK done resp}	0000h	AAh	08h
{MBINIT.REPAIRVAL init req}	0000h	A5h	09h
{MBINIT.REPAIRVAL init resp}	0000h	AAh	09h
{MBINIT.REPAIRVAL result req}	0000h	A5h	0Ah
{MBINIT.REPAIRVAL result resp}	[15:2]: Reserved [1]: Compare Results from RRDVLD_L [0]: Compare Results from RVLD_L	AAh	0Ah
{MBINIT.REPAIRVAL apply repair req}	[15:2]: Reserved [1:0]: Repair Encoding 3h: Reserved 0h: Repair RVLD_L 1h: Reserved	A5h	0Bh
{MBINIT.REPAIRVAL apply repair resp}	0000h	AAh	0Bh
{MBINIT.REPAIRVAL done req}	0000h	A5h	0Ch
{MBINIT.REPAIRVAL done resp}	0000h	AAh	0Ch
{MBINIT.REVERSALMB init req}	0000h	A5h	0Dh
{MBINIT.REVERSALMB init resp}	0000h	AAh	0Dh
{MBINIT.REVERSALMB clear error req}	0000h	A5h	0Eh
{MBINIT.REVERSALMB clear error resp}	0000h	AAh	0Eh
{MBINIT.REVERSALMB result req}	0000h	A5h	0Fh
{MBINIT.REVERSALMB done req}	0000h	A5h	10h
{MBINIT.REVERSALMB done resp}	0000h	AAh	10h
{MBINIT.REPAIRMB start req}	0000h	A5h	11h
{MBINIT.REPAIRMB start resp}	0000h	AAh	11h
{MBINIT.REPAIRMB Apply repair resp}	0000h	AAh	12h
{MBINIT.REPAIRMB end req}	0000h	A5h	13h
{MBINIT.REPAIRMB end resp}	0000h	AAh	13h
{MBINIT.REPAIRMB apply degrade req}	[15:3]: Reserved [2:0]: Standard package logical Lane map	A5h	14h
{MBINIT.REPAIRMB apply degrade resp}	0000h	AAh	14h

Table 7-9. Link Training State Machine related Message encodings for messages without data (Sheet 3 of 4)

Message	MsgInfo[15:0]	MsgCode[7:0]	MsgSubcode[7:0]
{MBTRAIN.VALVREF start req}	0000h	B5h	00h
{MBTRAIN.VALVREF start resp}	0000h	BAh	00h
{MBTRAIN.VALVREF end req}	0000h	B5h	01h
{MBTRAIN.VALVREF end resp}	0000h	BAh	01h
{MBTRAIN.DATAVREF start req}	0000h	B5h	02h
{MBTRAIN.DATAVREF start resp}	0000h	BAh	02h
{MBTRAIN.DATAVREF end req}	0000h	B5h	03h
{MBTRAIN.DATAVREF end resp}	0000h	BAh	03h
{MBTRAIN.SPEEDIDLE done req}	0000h	B5h	04h
{MBTRAIN.SPEEDIDLE done resp}	0000h	BAh	04h
{MBTRAIN.TXSELFICAL Done req}	0000h	B5h	05h
{MBTRAIN.TXSELFICAL Done resp}	0000h	BAh	05h
{MBTRAIN.RXCLKCAL start req}	0000h	B5h	06h
{MBTRAIN.RXCLKCAL start resp}	0000h	BAh	06h
{MBTRAIN.RXCLKCAL done req}	0000h	B5h	07h
{MBTRAIN.RXCLKCAL done resp}	0000h	BAh	07h
{MBTRAIN.VALTRAINCENTER start req}	0000h	B5h	08h
{MBTRAIN.VALTRAINCENTER start resp}	0000h	BAh	08h
{MBTRAIN.VALTRAINCENTER done req}	0000h	B5h	09h
{MBTRAIN.VALTRAINCENTER done resp}	0000h	BAh	09h
{MBTRAIN.VALTRAINVREF start req}	0000h	B5h	0Ah
{MBTRAIN.VALTRAINVREF start resp}	0000h	BAh	0Ah
{MBTRAIN.VALTRAINVREF done req}	0000h	B5h	0Bh
{MBTRAIN.VALTRAINVREF done resp}	0000h	BAh	0Bh
{MBTRAIN.DATATRAINCENTER1 start req}	0000h	B5h	0Ch
{MBTRAIN.DATATRAINCENTER1 start resp}	0000h	BAh	0Ch
{MBTRAIN.DATATRAINCENTER1 end req}	0000h	B5h	0Dh
{MBTRAIN.DATATRAINCENTER1 end resp}	0000h	BAh	0Dh
{MBTRAIN.DATATRAINVREF start req}	0000h	B5h	0Eh
{MBTRAIN.DATATRAINVREF start resp}	0000h	BAh	0Eh
{MBTRAIN.DATATRAINVREF end req}	0000h	B5h	10h
{MBTRAIN.DATATRAINVREF end resp}	0000h	BAh	10h
{MBTRAIN.RXDESKEW start req}	0000h	B5h	11h
{MBTRAIN.RXDESKEW start resp}	0000h	BAh	11h
{MBTRAIN.RXDESKEW end req}	0000h	B5h	12h
{MBTRAIN.RXDESKEW end resp}	0000h	BAh	12h
{MBTRAIN.DATATRAINCENTER2 start req}	0000h	B5h	13h
{MBTRAIN.DATATRAINCENTER2 start resp}	0000h	BAh	13h
{MBTRAIN.DATATRAINCENTER2 end req}	0000h	B5h	14h
{MBTRAIN.DATATRAINCENTER2 end resp}	0000h	BAh	14h

Table 7-9. Link Training State Machine related Message encodings for messages without data (Sheet 4 of 4)

Message	MsgInfo[15:0]	MsgCode[7:0]	MsgSubcode[7:0]
{MBTRAIN.LINKSPEED start req}	0000h	B5h	15h
{MBTRAIN.LINKSPEED start resp}	0000h	BAh	15h
{MBTRAIN.LINKSPEED error req}	0000h	B5h	16h
{MBTRAIN.LINKSPEED error resp}	0000h	BAh	16h
{MBTRAIN.LINKSPEED exit to repair req}	0000h	B5h	17h
{MBTRAIN.LINKSPEED exit to repair resp}	0000h	BAh	17h
{MBTRAIN.LINKSPEED exit to speed degrade req}	0000h	B5h	18h
{MBTRAIN.LINKSPEED exit to speed degrade resp}	0000h	BAh	18h
{MBTRAIN.LINKSPEED done req}	0000h: for regular response	B5h	19h
{MBTRAIN.LINKSPEED done resp}	0000h: for regular response FFFFh: for stall	BAh	19h
{MBTRAIN.LINKSPEED multi-module disable module resp}	0000h	BAh	1Ah
{MBTRAIN.LINKSPEED exit to phy retrain req}	0000h	B5h	1Fh
{MBTRAIN.LINKSPEED exit to phy retrain resp}	0000h	BAh	1Fh
{MBTRAIN.REPAIR init req}	0000h	B5h	1Bh
{MBTRAIN.REPAIR init resp}	0000h	BAh	1Bh
{MBTRAIN.REPAIR Apply repair resp}	0000h	BAh	1Ch
{MBTRAIN.REPAIR end req}	0000h	B5h	1Dh
{MBTRAIN.REPAIR end resp}	0000h	BAh	1Dh
{MBTRAIN.REPAIR Apply degrade req}	[15:3]: Reserved [2:0]: Standard Package logical Lane map ^b	B5h	1Eh
{MBTRAIN.REPAIR Apply degrade resp}	0000h	BAh	1Eh
{PHYRETRAIN.retrain start req}	[15:3]: Reserved [2:0]: Retrain Encoding	C5h	01h
{PHYRETRAIN.retrain start resp}	[15:3]: Reserved [2:0]: Retrain Encoding	CAh	01h
{TRAINERROR Entry req}	0000h	E5h	00h
{TRAINERROR Entry resp}	0000h	EAh	00h
{RECAL.track pattern init req}	0000h	D5h	00h
{RECAL.track pattern init resp}	0000h	DAh	00h
{RECAL.track pattern done req}	0000h	D5h	01h
{RECAL.track pattern done resp}	0000h	DAh	01h

a. See [Section 4.5.3.2](#)b. See [Table 4-9](#)

7.1.2.3 Messages with data payloads

[Figure 7-4](#) shows the formats for Messages with data payloads. The definitions of opcode, srcid, dstid, dp, and cp fields are the same as Register Access packets.

Figure 7-4. Format for Messages with data payloads

Messages with data																																		
Bytes	3							2							1							0												
Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Header / Data	Header																																	
Phase0	srcid	rsvd	rsvd	rsvd	rsvd	msgcode[7:0]	rsvd																											
Phase1	dp	cp	rsvd	dstid	MsgInfo[15:0]	MsgSubcode[7:0]																												
Header / Data	Data																																	
Phase2																																		
Phase3																																		

Table 7-10 and Table 7-11 give the message encodings.

Table 7-10. Message encodings for Messages with Data (Sheet 1 of 3)

Name	Msg code	Msgsubcode	MsgInfo	Data Bit Encodings	Description
{AdvCap.Adapter}	01h	00h	0000h: Regular Message FFFFh: Stall Message	[0]: "Raw Format" [1]: "68B Flit Mode" [2]: "CXL 256B Flit Mode" [3]: "PCIe Flit Mode" [4]: "Streaming" [5]: "Retry" [6]: "Multi_Protocol_Enable" [7]: "Stack0_Enable" [8]: "Stack1_Enable" [9]: "CXL_LatOpt_Fmt5" [10]: "CXL_LatOpt_Fmt6" [11]: "Retimer" [20:12]: "Retimer Credits" [21]: "DP" [22]: "UP" [23]: "68B Flit Format" [24]: "Standard 256B End Header Flit Format" [25]: "Standard 256B Start Header Flit Format" [26]: "Latency-Optimized 256B without Optional Bytes Flit Format" [27]: "Latency-Optimized 256B with Optional Bytes Flit Format" [28]: "Enhanced_Multi_Protocol_Enable" [29]: "Stack 0 Maximum Bandwidth_Limit" [30]: "Stack 1 Maximum Bandwidth_Limit" [31]: "Management Transport Protocol" [63:32]: Reserved	Advertised Capabilities of the D2D Adapter
{FinCap.Adapter}	02h	00h	0000h: Regular Message FFFFh: Stall Message	[0]: "Raw Format" [1]: "68B Flit Mode" [2]: "CXL 256B Flit Mode" [3]: "PCIe Flit Mode" [4]: "Streaming" [5]: "Retry" [6]: "Multi_Protocol_Enable" [7]: "Stack0_Enable" [8]: "Stack1_Enable" [9]: "CXL_LatOpt_Fmt5" [10]: "CXL_LatOpt_Fmt6" [11]: "Retimer" [20:12]: "Retimer Credits" [21]: "DP" [22]: "UP" [23]: "68B Flit Format" [24]: "Standard 256B End Header Flit Format" [25]: "Standard 256B Start Header Flit Format" [26]: "Latency-Optimized 256B without Optional Bytes Flit Format" [27]: "Latency-Optimized 256B with Optional Bytes Flit Format" [28]: "Enhanced_Multi_Protocol_Enable" [29]: "Stack 0 Maximum Bandwidth_Limit" [30]: "Stack 1 Maximum Bandwidth_Limit" [31]: "Management Transport Protocol" [63:32]: Reserved	Finalized Capability of the D2D Adapter

Table 7-10. Message encodings for Messages with Data (Sheet 2 of 3)

Name	Msg code	Msgsubcode	MsgInfo	Data Bit Encodings	Description
{AdvCap.CXL}	01h	01h	0000h: Post negotiation, if Enhanced_Multi_Protocol_Enable is 0b, or it is 1b and the message is for Stack 0. 0001h: Post negotiation, if Enhanced_Multi_Protocol_Enable is 1b and the message is for Stack 1. FFFFh: Stall Message	[23:0] : Flexbus Mode negotiation usage bits as defined for Symbols 12-14 of Modified TS1/TS2 Ordered Set in CXL Specification, with the following additional rules: <ul style="list-style-type: none">• [0]: PCIe capable/enable - this must be 1b for PCIe Non-Flit Mode.• [1]: CXL.io capable/enable - this must be 0b for PCIe Non-Flit Mode.• [2]: CXL.mem capable/enable - this must be 0b for PCIe Non-Flit Mode.• [3]: CXL.cache capable/enable - this must be 0b for PCIe Non-Flit Mode.• [4]: CXL 68B Flit and VH capable; must be set for ports that support CXL protocols, as specified in the Protocol Layer interoperability requirements.• [8]: Multi-Logical Device - must be set to 0b for PCIe Non-Flit Mode.• [9]: Reserved.• [12:10]: these bits do not apply for UCIE, must be 0b.• [14]: Retimer 2 - does not apply for UCIE, must be 0b.• [15]: CXL.io Throttle - must be 0b for PCIe Non-Flit Mode.• [17:16]: NOP Hint Info - does not apply for UCIE, and must be 0.	Advertised Capabilities for CXL protocol.
{FinCap.CXL}	02h	01h	0000h: Post negotiation, if Enhanced_Multi_Protocol_Enable is 0b, or it is 1b and the message is for Stack 0. 0001h: Post negotiation, if Enhanced_Multi_Protocol_Enable is 1b and the message is for Stack 1. FFFFh: Stall Message	[0]: "68B Flit Mode" [1]: "CXL 256B Flit Mode" [2]: "PCIe Flit Mode" [3]: "Streaming Protocol" [4]: "Management Transport Protocol" [63:5]: Reserved	Finalized Capabilities for CXL protocol.
{MultiProtAdvCap.Adapter}	01h	02h	0000h: Reserved FFFFh: Stall Message	[0]: "68B Flit Mode" [1]: "CXL 256B Flit Mode" [2]: "PCIe Flit Mode" [3]: Reserved [4]: "Management Transport Protocol" [63:5]: Reserved	Protocol Advertisement for Stack 1 when Enhanced Multi_Protocol_Enable is negotiated
{MultiProtFinCap.Adapter}	02h	02h	0000h: Reserved FFFFh: Stall Message	[0]: "68B Flit Mode" [1]: "CXL 256B Flit Mode" [2]: "PCIe Flit Mode" [3]: Reserved [4]: "Management Transport Protocol" [63:5]: Reserved	Finalized Capability for Protocol negotiation when Enhanced Multi_Protocol_Enable is negotiated and Stack 1 is PCIe or CXL

Table 7-10. Message encodings for Messages with Data (Sheet 3 of 3)

Name	Msg code	MsgSubcode	MsgInfo	Data Bit Encodings	Description
{Vendor Defined Message}	FFh	--	Vendor ID		<p>Vendor Defined Messages. These can be exchanged at any time after sideband is functional post SBINIT. Interoperability is vendor defined. Unsupported vendor defined messages must be discarded by the receiver.</p> <p>Note that this is NOT the UCIE Vendor ID, but rather the unique identifier of the chiplet vendor that is defining and using these messages.</p>
All other encodings not mentioned in this table are reserved.					

Table 7-11. Link Training State Machine related encodings (Sheet 1 of 6)

Message	MsgInfo[15:0]	MsgCode [7:0]	MsgSubcode [7:0]	Data Field[63:0]
{Start Tx Init D to C point test req}	[15:0]: Maximum comparison error threshold	85h	01h	<p>[63:60]: Reserved</p> <p>[59]: Comparison Mode (0: Per Lane; 1: Aggregate)</p> <p>[58:43]: Iteration Count Settings</p> <p>[42:27]: Idle Count settings</p> <p>[26:11]: Burst Count settings</p> <p>[10]: Pattern Mode (0: continuous mode, 1: Burst Mode)</p> <p>[9:6] : Clock Phase control at Tx Device (0h: Clock PI Center, 1h: Left Edge, 2h: Right Edge)</p> <p>[5:3] : Valid Pattern (0h: Functional pattern)</p> <p>[2:0]: Data pattern (0h: LFSR, 1h: Per Lane ID)</p>

Table 7-11. Link Training State Machine related encodings (Sheet 2 of 6)

Message	MsgInfo[15:0]	MsgCode [7:0]	MsgSubcode [7:0]	Data Field[63:0]
{Tx Init D to C results resp}	[15:6]: Reserved [5]: Valid Lane comparison results [4]: Cumulative Results of all Lanes (0: Fail (Errors > Max Error Threshold), 1: Pass (Errors <= Max Error Threshold)). [3:0]: UCIE-A: Compare results from Redundant Lanes (0h: Fail (Errors > Max Error Threshold), 1h: Pass (Errors <= Max Error Threshold)) (RRD_L[3], RRD_L[2], RRD_L[1], RRD_L[0]) UCIE-S: Reserved RRD_L[3] and RRD_L[2] are reserved for UCIE-A x32 as a transmitter of this message.	8Ah	03h	[63:0]: Compare Results of individual Data Lanes (0h: Fail (Errors > Max Error Threshold), 1h: Pass (Errors <= Max Error Threshold)) UCIE-A {RD_L[63], RD_L[62], ..., RD_L[1], RD_L[0]} UCIE-S {48'h0, RD_L[15], RD_L[14], ..., RD_L[1], RD_L[0]} UCIE-A x32 {32'h0, RD_L[31], RD_L[30], ..., RD_L[0]} UCIE-S x8 {56'h0, RD_L[7], RD_L[6], ..., RD_L[1], RD_L[0]}
{Start Tx Init D to C eye sweep req}	[15:0]: Maximum comparison error threshold	85h	05h	[63:60]: Reserved [59]: Comparison Mode (0: Per Lane; 1: Aggregate) [58:43]: Iteration Count Settings [42:27]: Idle Count settings [26:11]: Burst Count settings [10]: Pattern Mode (0: continuous mode, 1: Burst Mode) [9:6]: Clock Phase control at Tx Device (0h: Clock PI Center, 1h: Left Edge, 2h: Right Edge) [5:3]: Valid Pattern (0h: Functional pattern) [2:0]: Data pattern (0h: LFSR, 1h: Per Lane ID)
{Start Rx Init D to C point test req}	[15:0]: Maximum comparison error threshold	85h	07h	[63:60]: Reserved [59]: Comparison Mode (0: Per Lane; 1: Aggregate) [58:43]: Iteration Count Settings [42:27]: Idle Count settings [26:11]: Burst Count settings [10]: Pattern Mode (0: continuous mode, 1: Burst Mode) [9:6]: Clock Phase control at Transmitter (0h: Clock PI Center, 1h: Left Edge, 2h: Right Edge) [5:3]: Valid Pattern (0h: Functional pattern) [2:0]: Data pattern (0h: LFSR, 1h: Per Lane ID)

Table 7-11. Link Training State Machine related encodings (Sheet 3 of 6)

Message	MsgInfo[15:0]	MsgCode [7:0]	MsgSubcode [7:0]	Data Field[63:0]
{Start Rx Init D to C eye sweep req}	[15:0]: Maximum comparison error threshold	85h	0Ah	[63:60]: Reserved [59]: Comparison Mode (0: Per Lane; 1: Aggregate) [58:43]: Iteration Count Settings [42:27]: Idle Count settings [26:11]: Burst Count settings [10]: Pattern Mode (0: continuous mode, 1: Burst Mode) [9:6]: Clock Phase control at Transmitter (0h: Clock PI Center, 1h: Left Edge, 2h: Right Edge) [5:3]: Valid Pattern (0h: Functional pattern) [2:0]: Data pattern (0h: LFSR, 1h: Per Lane ID)
{Rx Init D to C results resp}	[15:6]: Reserved [5]: Valid Lane comparison result [4]: Cumulative Results of all Lanes (0: Fail (Errors > Max Error Threshold), 1: Pass (Errors <= Max Error Threshold)). [3:0]: PCIe-A: Compare results from Redundant Lanes (0h: Fail (Errors > Max Error Threshold), 1h: Pass (Errors <= Max Error Threshold)) (RRD_L[3], RRD_L[2], RRD_L[1], RRD_L[0]) PCIe-S: Reserved RRD_L[3] and RRD_L[2] are reserved for PCIe-A x32 as a transmitter of this message.	8Ah	0Bh	[63:0]: Compare Results of individual Data Lanes (0h: Fail (Errors > Max Error Threshold), 1h: Pass (Errors <= Max Error Threshold)) PCIe-A {RD_L[63], RD_L[62], ..., RD_L[1], RD_L[0]} PCIe-S {48'h0, RD_L[15], RD_L[14], ..., RD_L[1], RD_L[0]} PCIe-A x32 {32'h0, RD_L[31], RD_L[30], ..., RD_L[0]} PCIe-S x8 {56'h0, RD_L[7], RD_L[6], ..., RD_L[1], RD_L[0]}
{Rx Init D to C sweep done with results}	0000h	81h	0Ch	[63:16]: Reserved [15:8]: Right Edge [7:0]: Left Edge
{MBINIT.PARAM configuration req}	0000h	A5h	00h	[63:15]: Reserved [14]: Sideband feature extensions is supported (1) or not supported (0) [13]: PCIe-A x32 [12:11]: Module ID: 0h: 0, 1h: 1, 2h: 2, 3h:3 [10]: Clock Phase: 0b: Differential clock, 1b: Quadrature phase [9]: Clock Mode - 0b: Strobe mode; 1b: Continuous mode [8:4]: Voltage Swing - The encodings are the same as the "Supported Tx Vswing encodings" field of the PHY Capability register [3:0]: Max IO Link Speed - The encodings are the same as "Max Link Speeds" field of the PCIe Link Capability register

Table 7-11. Link Training State Machine related encodings (Sheet 4 of 6)

Message	MsgInfo[15:0]	MsgCode [7:0]	MsgSubcode [7:0]	Data Field[63:0]
{MBINIT.PARAM configuration resp}	0000h	AAh	00h	[63:11]: Reserved [10]: Clock Phase: 0b: Differential clock, 1b: Quadrature phase [9]: Clock Mode - 0b: Strobe mode; 1b: Continuous mode [8:4]: Reserved [3:0]: Max IO Link Speed - The encodings are the same as "Max Link Speeds" field of the UCIe Link Capability register
{MBINIT.PARAM SBFE req}	0000h: Regular Message	A5h	01h	[63:3]: Reserved [2]: Sideband-only (SO) port (1), full UCIe port (0) [1]: Sideband Performant Mode Operation (PMO) is supported (1) or not supported (0) [0]: Management Transport protocol is supported (1) or not supported (0)
{MBINIT.PARAM SBFE resp}	0000h: Regular Message FFFFh: Stall Message	AAh	01h	[63:3]: Reserved [2]: Sideband-only (SO) port (1), full UCIe port (0) [1]: Sideband Performant Mode Operation (PMO) is negotiated (1) or not supported (0) [0]: Management Transport protocol is supported (1) or not supported (0)
{MBINIT.REVERSAL MB result resp}	The error condition for this flow is NOT observing 16 consecutive iterations of the expected pattern. The error threshold is always 0 for this test. [15:4]: Reserved [3:0]: UCIe-A: Compare results from Redundant Lanes (0h: Fail (Errors > Max Error Threshold), 1h: Pass (Errors <= Max Error Threshold)) (RRD_L[3], RRD_L[2], RRD_L[1], RRD_L[0]) UCIe-S: Reserved RRD_L[3] and RRD_L[2] are reserved for UCIe-A x32 as a transmitter of this message.	AAh	0Fh	The error condition for this flow is NOT observing 16 consecutive iterations of the expected pattern. The error threshold is always 0 for this test. [63:0]: Compare Results of individual Data Lanes (0h: Fail (Errors > Max Error Threshold), 1h: Pass (Errors <= Max Error Threshold)) UCIe-A {RD_L[63], RD_L[62], ..., RD_L[1], RD_L[0]} UCIe-S {48'h0, RD_L[15], RD_L[14], ..., RD_L[1], RD_L[0]} UCIe-A x32 {32'h0, RD_L[31], RD_L[30], ..., RD_L[0]} UCIe-S x8 {56'h0, RD_L[7], RD_L[6], ..., RD_L[1], RD_L[0]}

Table 7-11. Link Training State Machine related encodings (Sheet 5 of 6)

Message	MsgInfo[15:0]	MsgCode [7:0]	MsgSubcode [7:0]	Data Field[63:0]
{MBINIT.REPAIRMB Apply repair req}	0000h	A5h	12h	<p>[31:24] : Repair Address for TRD_P[3]: Indicates the physical Lane repaired when TRD_P[3] is used in remapping scheme. This is reserved for UCIE-A x32 as a transmitter of this message.</p> <p>20h: Invalid 21h: TD_P[33] Repaired 22h: TD_P[34] Repaired 3Eh: TD_P[62] Repaired 3Fh: TD_P[63] Repaired F0h: Reserved FFh: No Repair</p> <p>[23:16]: Repair Address for TRD_P[2]: Indicates the physical Lane repaired when TRD_P[2] is used in remapping scheme. This is reserved for UCIE-A x32 as a transmitter of this message.</p> <p>20h: TD_P[32] Repaired 21h: TD_P[33] Repaired 22h: TD_P[34] Repaired 3Eh: TD_P[62] Repaired 3Fh: TD_P[63] Repaired F0h: Reserved FFh: No Repair</p> <p>[15:8]: Repair Address for TRD_P[1]: Indicates the physical Lane repaired when TRD_P[1] is used in remapping scheme.</p> <p>00h: Invalid 01h: TD_P[1] Repaired 02h: TD_P[2] Repaired 1Eh: TD_P[30] Repaired 1Fh: TD_P[31] Repaired F0h: Reserved FFh: No Repair</p> <p>[7:0]: Repair Address for TRD_P[0]: Indicates the physical Lane repaired when TRD_P[0] is used in remapping scheme.</p> <p>00h: TD_P[0] Repaired 01h: TD_P[1] Repaired 02h: TD_P[2] Repaired 1Eh: TD_P[30] Repaired 1Fh: TD_P[31] Repaired F0h: Reserved FFh: No Repair</p>

Table 7-11. Link Training State Machine related encodings (Sheet 6 of 6)

Message	MsgInfo[15:0]	MsgCode [7:0]	MsgSubcode [7:0]	Data Field[63:0]
{MBTRAIN.REPAIR Apply repair req}	0000h	B5h	1Ch	<p>[31:24] : Repair Address for TRD_P[3]: Indicates the physical Lane repaired when TRD_P[3] is used in remapping scheme. This is reserved for UCIE-A x32 as a transmitter of this message.</p> <p>20h: Invalid 21h: TD_P[33] Repaired 22h: TD_P[34] Repaired 3Eh: TD_P[62] Repaired 3Fh: TD_P[63] Repaired F0h: Reserved FFh: No Repair</p> <p>[23:16]: Repair Address for TRD_P[2]: Indicates the physical Lane repaired when TRD_P[2] is used in remapping scheme. This is reserved for UCIE-A x32 as a transmitter of this message.</p> <p>20h: TD_P[32] Repaired 21h: TD_P[33] Repaired 22h: TD_P[34] Repaired 3Eh: TD_P[62] Repaired 3Fh: TD_P[63] Repaired F0h: Reserved FFh: No Repair</p> <p>[15:8]: Repair Address for TRD_P[1]: Indicates the physical Lane repaired when TRD_P[1] is used in remapping scheme.</p> <p>00h: Invalid 01h: TD_P[1] Repaired 02h: TD_P[2] Repaired 1Eh: TD_P[30] Repaired 1Fh: TD_P[31] Repaired F0h: Reserved FFh: No Repair</p> <p>[7:0]: Repair Address for TRD_P[0]: Indicates the physical Lane repaired when TRD_P[0] is used in remapping scheme.</p> <p>00h: TD_P[0] Repaired 01h: TD_P[1] Repaired 02h: TD_P[2] Repaired 1Eh: TD_P[30] Repaired 1Fh: TD_P[31] Repaired F0h: Reserved FFh: No Repair</p>

7.1.2.4 Management Port Message (MPM) with Data

As with all sideband messages, Management Port Messages with Data also carry a 1-QWORD header. This is referred to as “MPM header” (see [Figure 7-5](#)) for the remainder of this section. The payload in these messages is referred to as “MPM payload” for the remainder of this section.

Bits [21:14] in the first DW of the MPM Hdr of a MPM with Data message, forms an 8b msgcode that denotes a specific MPM with Data message. [Table 7-12](#) summarizes the supported MPM with Data messages over sideband.

Support for these messages is optional and negotiated as described in [Section 8.2.3.1](#).

Table 7-12. Supported MPM with Data Messages on Sideband

msgcode	Message
01h	Encapsulated MTP Message
FFh	Vendor-defined Management Port Gateway Message
Others	Reserved

7.1.2.4.1 Common Fields in MPM Header of MPM with Data Messages on Sideband

[Figure 7-5](#) shows and [Table 7-13](#) describes the common fields in the MPM header of MPM with data messages on the sideband.

Figure 7-5. Common Fields in MPM Header of all MPM with Data Messages on Sideband

3			2			1			0																						
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
srcid=011b	rsvd		re sp	vc		msgcode				length						rs vd	opcode = 11000b														
rs vd	cp	rsvd	dstid=111b	msgcode-specific									rsvd	msgcode- specific	rsvd	rxqid															

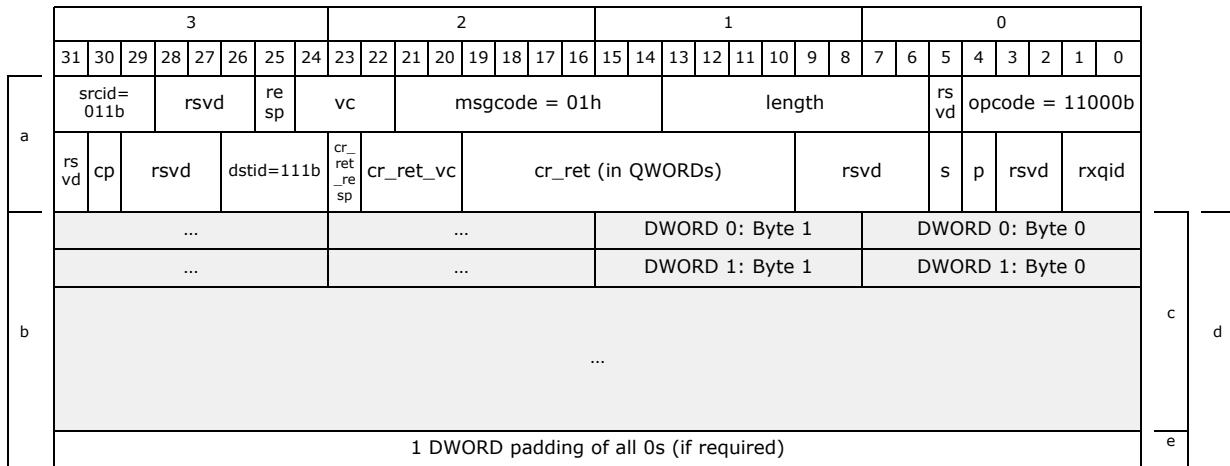
Table 7-13. Common Fields in MPM Header of all MPM with Data Messages on Sideband

Field	Description
opcode	11000b: MPM with Data.
length	MPM Payload length (i.e., 0h for 1 QWORD, 1h for 2 QWORDs, 2h for 3 QWORDs, etc.).
msgcode	Message code as defined in Table 7-12 .
vc	Virtual Channel ID.
resp	0: Request MPM. 1: Response MPM. For a Vendor-defined Management Port Gateway Message with Data, this bit is always 0 (see Section 7.1.2.4.3).
srcid	011b: Indicates Management Port Gateway as source.
dstid	111b: Indicates Management Port Gateway as target.
cp	Control parity for all bits in the sideband packet header.
rxqid	RxQ-ID to which this packet is destined, and RxQ-ID associated with any credits returned in the packet (see Section 8.2.3.1.2 for RxQ details).

7.1.2.4.2 Encapsulated MTP Message

Encapsulated MTP on sideband is an MPM with Data message with a msgcode of 01h.

Figure 7-6. Encapsulated MTP on Sideband



- a. MPM Header.
- b. MPM Payload.
- c. Management Transport Packet (MTP).
- d. Length in MPM Header.
- e. DWORD padding.

Table 7-14. Encapsulated MTP on Sideband Fields

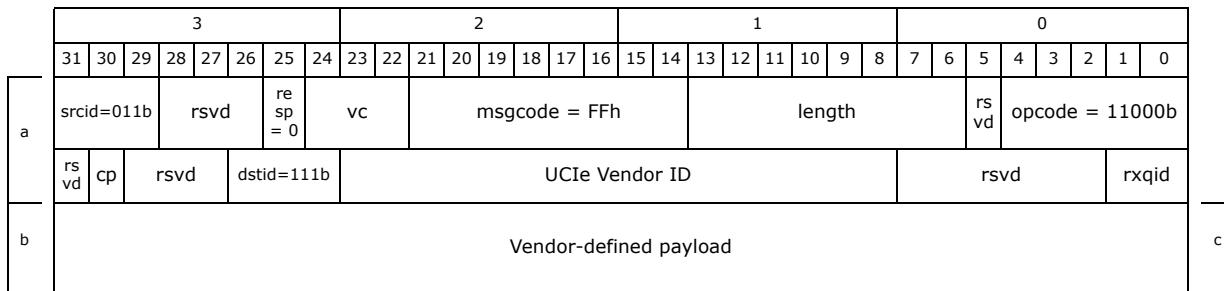
Location	Bit	Description
MPM Header ^a	s	Segmented MTP (see Section 8.2.4.2). The first and middle segments in a segmented MTP have this bit set to 1. The last segment in a segmented MTP will have this bit cleared to 0. An unsegmented MTP also has this bit cleared to 0.
	p	If this is set to 1, there is 1-DWORD padding of all 0s added at the end of the packet, to align to a QWORD boundary.
	cr_ret	Value of RxQ credits being returned to the MPG receiving this message, indicated by the rxqid value and its VC:Resp channel indicated via cr_ret_vc/cr_ret_resp fields. 000h indicates 0 credits returned. 001h indicates 1 credit returned. ... 3FEh indicates 1022 credits returned. 3FFh is reserved. If there is no credit being returned, cr_ret fields must be set to 0h.
	cr_ret_vc	VC associated with the credit returned.
	cr_ret_resp	Resp value associated with the credit returned. 0=Request channel credit. 1=Response channel credit.
MPM Payload	—	See Section 8.2 for details. Note that DWORDx:Bytey in Figure 7-6 refers to the corresponding DWORD, Byte defined in the Management Transport Packet in Figure 8-5 .

- a. See [Section 7.1.2.4.1](#) for details of header fields common to all MPMs with data on the sideband.

7.1.2.4.3 Vendor-defined Management Port Gateway Message

The Vendor-defined Management Port Gateway message with data is defined for custom communication between MPGs on the two ends of a UCIe sideband link. These messages are not part of the Management transport protocol, and these messages start at an MPG and terminate at the MPG on the other end of the UCIe sideband link. These messages share the same RxQ-ID request buffers and credits as encapsulated MTP messages. If an MPG does not support these messages or does not support vendor-defined messages from a given vendor (identified by the UCIe Vendor ID in the header), the MPG silently drops those messages. Length of these Vendor defined messages is subject to the same rules stated in [Section 8.2.5.1.2](#). Ordering of these messages sent over multiple sideband links is subject to the same rules presented in [Section 8.2.4.3](#) for encapsulated MTPs.

Figure 7-7. Vendor-defined Management Port Gateway Message with Data on Sideband



- a. MPM Header.
- b. MPM Payload.
- c. Length in MPM Header.

Table 7-15. Vendor-defined Management Port Gateway Message with Data on Sideband Fields

Location	Field	Description
MPM Header ^a	UCIe Vendor ID	UCIe Consortium-assigned unique ID for each vendor.
MPM Payload	—	Vendor-defined.

a. See [Section 7.1.2.4.1](#) for details of header fields common to all MPMs with data on the sideband.

7.1.2.5 MPMs without Data

Bits [21:14] in the first DWORD of the MPM header of an MPM without Data message form an 8b msgcode that denotes a specific MPM without Data message. [Table 7-16](#) lists the supported msgcodes.

Table 7-16. Supported MPM without Data Messages on Sideband

msgcode	Message
01h	Management Port Gateway Capabilities Message
02h	Credit Return Message
03h	Init Done Message
04h	PM Message
FFh	Vendor-defined Management Port Gateway Message
Others	Reserved

7.1.2.5.1 Common Header Fields of MPM without Data Messages on Sideband

Figure 7-8 shows and Table 7-17 describes the common fields in the MPM header of MPM without data messages on the sideband.

Figure 7-8. Common Fields in MPM Header of all MPM without Data Messages on Sideband

3			2						1						0																
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
srcid=011b		rsvd						msgcode						msgcode-specific						rs vd	opcode = 10111b										
rs vd	cp	rsvd	dstid=111b	msgcode-specific												rsvd	msgcode- specific														

Table 7-17. Common Fields in MPM Header of all MPM without Data Messages on Sideband

Field	Description
opcode	10111b: MPM without Data.
msgcode	Message code as defined in Table 7-16.
srcid	011b: Indicates Management Port Gateway as source.
cp	Control parity for all bits in the sideband packet header.
dstid	111b: Indicates Management Port Gateway as target.

7.1.2.5.2 Management Port Gateway Capabilities Message

See Section 8.2.3.1.2 for usage of this message during sideband management transport path initialization.

Figure 7-9 shows and Table 7-18 describes the Management Port Gateway Capabilities message format on the sideband.

Figure 7-9. Management Port Gateway Capabilities MPM on Sideband

3			2						1						0																
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
srcid=011b		rsvd						msgcode = 01h						NumVC	rsvd						opcode = 10111b										
a	rs vd	cp	rsvd	dstid=111b	Port ID[15:0]												rsvd														

a. MPM Header.

Table 7-18. Management Port Gateway Capabilities MPM Header Fields on Sideband^a

Field	Description
NumVC	Number of VCs supported by the Management Port Gateway that is transmitting the message.
Port ID	Port ID number value of the Management port associated with the Management Port Gateway that is issuing the message (see Section 8.1.3.6.2.1).

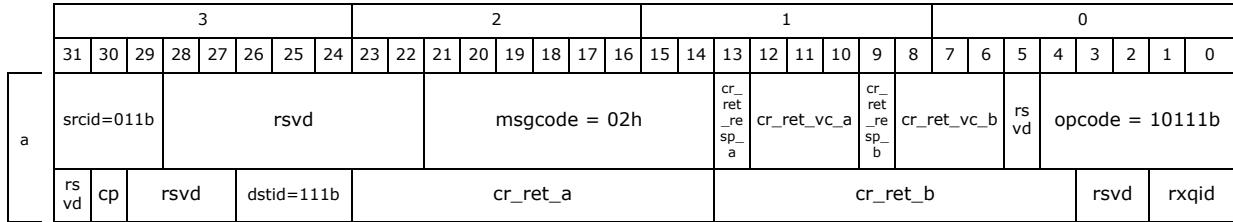
a. See Table 7-17 for details of header fields common to all MPMS without data on the sideband.

7.1.2.5.3 Credit Return Message

See Section 8.2.3.1.2 for usage of this message during sideband management transport path initialization.

Figure 7-10 shows and Table 7-19 describes the Credit Return message format on the sideband. If credit returns a and b carry the same vc:resp fields, then the total credit returned for that rxqid:vc:resp credit type is the sum of cr_ret_a and cr_ret_b.

Figure 7-10. Credit Return MPM on Sideband



a. MPM Header.

Table 7-19. Credit Return MPM Header Fields on Sideband^a

Field	Description
cr_ret_vc_a(b)	VC for which the credit is being returned.
cr_ret_resp_a(b)	Resp value associated with the credit returned. 0=Request channel credit. 1=Response channel credit.
cr_ret_a(b)	Value of credits returned for the RxQ (in the Management Port Gateway transmitting this message) indicated by the rxqid field and the associated VC:Resp channel indicated via cr_ret_vc_a(b)/cr_ret_resp_a(b) fields. 000h indicates 0 credits returned. 001h indicates 1 credit returned. ... 3FEh indicates 1022 credits returned. 3FFh indicates infinite credits. 3FFh value is legal only on credit returns that happen during VC initialization (i.e., before Init Done message is sent) and cannot be used after initialization until the transport path is renegotiated/initialized again. If a receiver detects infinite credit returns after VC initialization and during runtime, it silently ignores it.
rxqid	RxQ-ID of the receiver queue for which the credits are being returned (see Section 8.2.3.1.2 for RxQ details).

a. See Table 7-17 for details of header fields common to all MPMs without data on the sideband.

7.1.2.5.4 Init Done Message

See Section 8.2.5.1.4 for usage of this message during sideband management transport path initialization.

Figure 7-11 shows and Table 7-20 describes the Init Done message format on the sideband.

Figure 7-11. Init Done MPM on Sideband

			3								2								1								0															
			31 30 29 28 27 26 25 24								23 22 21 20 19 18 17 16								15 14 13 12 11 10 9 8								7 6 5 4 3 2 1 0															
			srcid=011b								rsvd								msgcode = 03h								rsvd								opcode = 10111b							
			rs vd		cp		rsvd		dstid=111b		rsvd																									rxqid						

a. MPM Header.

Table 7-20. Init Done MPM Header Fields on Sideband^a

Field	Description
rxqid	RxQ-ID of the receiver queue that has completed initializing credits (see Section 8.2.3.1.2 for RxQ details).

a. See Table 7-17 for details of header fields common to all MPMs without data on the sideband.

7.1.2.5.5 PM Message

See Section 8.2.5.1.4 for usage of this message during sideband management transport PM flows.

Figure 7-12 shows and Table 7-21 describes the PM message format on the sideband.

Figure 7-12. PM MPM on Sideband

			31 30 29 28 27 26 25 24								23 22 21 20 19 18 17 16								15 14 13 12 11 10 9 8								7 6 5 4 3 2 1 0											
			srcid=011b								rsvd								msgcode = 04h								pmcode		rsvd								opcode = 10111b	
			rs vd		cp		rsvd		dstid=111b		rsvd																								rxqid			

a. MPM Header.

Table 7-21. PM MPM Header Fields on Sideband^a

Field	Description
pmcode	1h: Wake Req; 2h: Wake ack; 3h: Sleep Req; 4h: Sleep ack; 5h: Sleep nak; Others: Rsvd.
rxqid	RxQ-ID of the receiver queue to which the message applies (see Section 8.2.3.1.2 for RxQ details).

a. See Table 7-17 for details of header fields common to all MPMs without data on the sideband.

7.1.2.5.6 Vendor-defined Management Port Gateway Message

The Vendor-defined Management Port Gateway message without data is defined for custom communication between the MPGs on both ends of a PCIe sideband link. These messages are not part of the management transport protocol, and these messages start at an MPG and terminate at the MPG on the other end of the PCIe sideband link. These messages share the same RxQ-ID request buffers as encapsulated MTP messages. If an MPG does not support these messages or does not

support these messages from a given vendor (identified by the UCIe Vendor ID in the header), the MPG silently drops those messages.

The Vendor-defined Management Port Gateway message without data on the sideband has the format shown in Figure 7-13.

Figure 7-13. Vendor-defined Management Port Gateway MPM without Data on Sideband

3			2								1								0												
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
a	srcid=011b	rsvd	resp = 0	vc	msgcode = FFh								Vendor-defined								rs vd	opcode = 10111b									
	rs vd	cp	rsvd	dstid=111b	UCIe Vendor ID												rsvd				rxqid										

a. MPM Header.

Table 7-22. MPM Header Vendor-defined Management Port Gateway Message without Data on Sideband^a

Field	Descriptions
Vendor-defined	Defined by the vendor specified in the UCIe Vendor ID field.
vc	Virtual Channel ID.
resp	Vendor-defined Management Port Gateway message without data always uses the Request channel. The value must be 0.
UCIe Vendor ID	UCIe Consortium-assigned unique ID for each vendor.
rxqid	RxQ-ID of the receiver queue to which the message belongs (see Section 8.2.3.1.2 for RxQ details).

a. See Table 7-17 for details of header fields common to all MPMs without data on the sideband.

7.1.3 Flow Control and Data Integrity

Sideband packets can be transferred across FDI, RDI or the UCIe sideband Link. Each of these have independent flow control.

7.1.3.1 Flow Control and Data Integrity over FDI and RDI

For each Transmitter associated with FDI or RDI, a design time parameter of the interface is used to determine the number of credits advertised by the Receiver, with a maximum of 32 credits. Each credit corresponds to 64 bits of header and 64 bits of potentially associated data. Thus, there is only one type of credit for all sideband packets, regardless of how much data they carry. Every Transmitter/Receiver pair has an independent credit loop. For example, on RDI, credits are advertised from Physical Layer to Adapter for sideband packets transmitted from the Adapter to the Physical Layer; and credits are also advertised from Adapter to the Physical Layer for sideband packets transmitted from the Physical Layer to the Adapter.

The Transmitter must check for available credits before sending Register Access requests and Messages. The Transmitter must not check for credits before sending Register Access Completions, and the Receiver must guarantee unconditional sinking for any Register Access Completion packets. Messages carrying requests or responses consume a credit on FDI and RDI, but they must be guaranteed to make forward progress by the Receiver and not get blocked behind Register Access requests. Both RDI and FDI give a dedicated signal for sideband credit returns across those interfaces.

All Receivers associated with RDI and FDI must check received messages for data or control parity errors, and these errors must be mapped to Uncorrectable Internal Errors (UIE) and transition RDI to LinkError state.

When supporting Management Port Messages over sideband, the Physical Layer maintains separate credited buffers (which is a design time parameter) per RxQ-ID it supports to which it can receive Management Port Messages from Management Port Gateway over the RDI configuration bus. Whether received over FDI or RDI, Management Port Messages are always sunk unconditionally in the Management Port Gateway.

7.1.3.2 Flow Control and Data Integrity over UCIE sideband Link between dies

The BER of the sideband Link is 1e-27 or better. Hence, no retry mechanism is provided for the sideband packets. Receivers of sideband packets must check for Data or Control parity errors, and any of these errors is mapped to a fatal UIE.

7.1.3.3 End-to-End flow control and forward progress for UCIE Link sideband

It is important for deadlock avoidance to ensure that there is sufficient space at the Receiver to sink all possible outstanding requests from the Transmitter, so that the requests do not get blocked at any intermediate buffers that would thereby prevent subsequent completions from making progress.

Sideband access for Remote Link partner's Adapter or Physical Layer registers is only accessible via the indirect mailbox mechanism, and the number of outstanding transactions is limited to four at a time. Although four credits are provisioned, there is only a single mailbox register, and this limits the number of outstanding requests that can use this mechanism to one at a time. The extra credits allow additional debug-related register access requests in case of register access timeouts. These credits are separate from local FDI or RDI accesses, and thus the Physical Layer must provision for sinking at least one register access request and completion each from remote die and local Adapter in addition to other sideband request credits (see Implementation Note below). The Adapter provisions for at least four remote register access requests from remote die Adapter. Each credit corresponds to 64b of header and 64b of data. Even requests that send no data or only send 32b of data consume one credit. Register Access completions do not consume a credit and must always sink.

If Management Transport Protocol is not supported, the Adapter credit counters for register access request are initialized to 4 on Domain Reset exit OR whenever RDI transitions from Reset to Active.

If Management Transport Protocol is supported, the Adapter credit counters for register access request are initialized to 4 on [Domain Reset exit] OR whenever [RDI transitions from Reset to Active AND SB_MGMT_UP=0].

It is permitted to send an extra (N-4) credit returns to remote Link partner if a UCIE implementation is capable of sinking a total of N requests once RDI has transitioned to Active state. The Adapter must implement a saturating credit counter capable of accumulating at least 4 credits, and hence prevent excess credit returns from overflowing the counter.

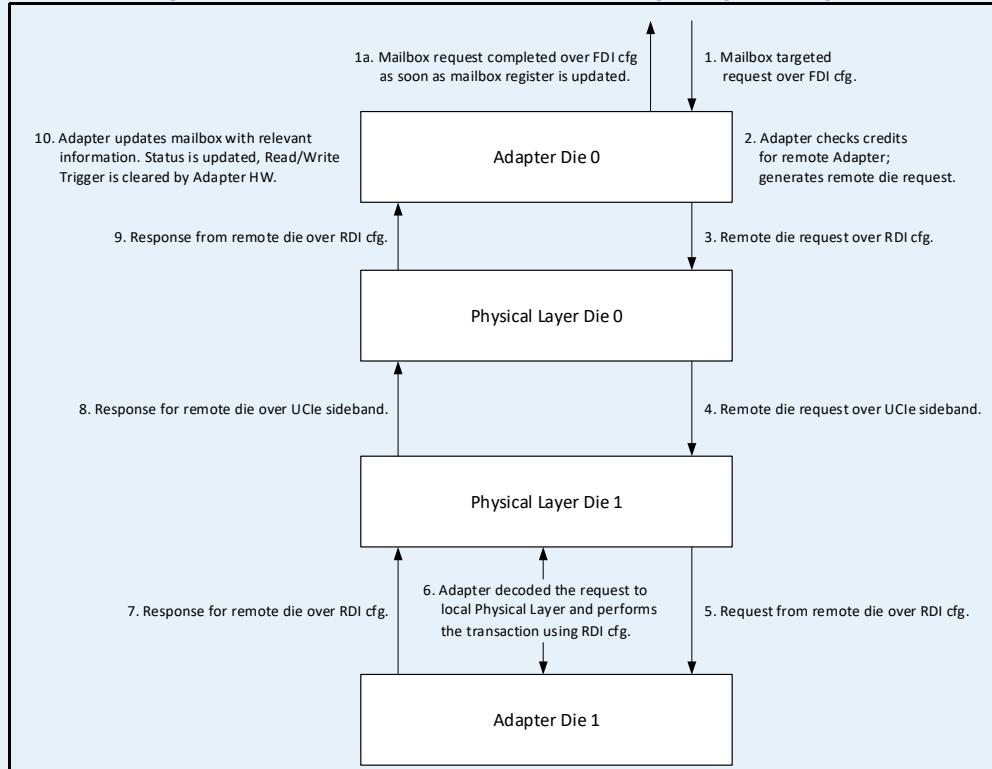
All other messages except Vendor Defined messages must always sink and make forward progress, and not block any messages on the sideband interface behind them. All Link Management message requests have an associated response, and the source of these messages must only have one outstanding request at a time (i.e., one outstanding message per "Link Management Request" MsgCode encoding).

For vendor defined messages, there must be a vendor defined cap on the number of outstanding messages, and the Receiver must guarantee sufficient space so as to not block any messages behind the vendor defined messages on any of the interfaces.

IMPLEMENTATION NOTE

Figure 7-14 shows an example of an end-to-end register access request to remote die and the corresponding completion returning back.

**Figure 7-14. Example Flow for Remote Register Access Request
(Local FDI/RDI Credit Checks Are Not Explicitly Shown)**



In Step 1 shown in Figure 7-14, the Protocol Layer checks for FDI credits before sending the request to Adapter Die 0. Adapter Die 0 completes the mailbox request as soon as the mailbox register is updated (shown in Step 1a). FDI credits are returned once its internal buffer space is free. In Step 2, Adapter Die 0 checks credits for remote Adapter as well as credits for local RDI before sending the remote die request to Physical Layer Die 0 in Step 3. Physical Layer schedules the request over UCIe sideband and returns the RDI credit to Adapter Die 0 once it has freed up its internal buffer space.

IMPLEMENTATION NOTE

Continued

In Step 5, Physical Layer Die 1 checks for Adapter Die 1 credits on RDI before sending the request over RDI. Adapter Die 1 decodes the request to see that it must access a register on Physical Layer Die 1; Adapter Die 1 checks for RDI credits of Physical Layer Die 1 before sending the request over RDI in Step 6. Adapter Die 1 must remap the tag for this request, if required, and save off the original tag of the remote die request as well as pre-allocate a space for the completion. Physical Layer Die 1 completes the register access request and responds with the corresponding completion. Because a completion is sent over RDI, no RDI credits need to be checked or consumed. Adapter Die 1 generates the completion for the remote die request and sends it over RDI (no credits are checked or consumed for completion over RDI) in Step 7. The completion is transferred across the different hops as shown in [Figure 7-14](#) and finally sunk in Adapter Die 0 to update the mailbox information. No RDI credits need to be checked for completions at the different hops.

For forward progress to occur, the Adapters and Physical Layers on both die must ensure that they can sink sufficient requests, completions, and messages to guarantee that there is no Link Layer level dependency between the different types of sideband packets (i.e., remote register access requests, remote register access completions, Link state transition messages for Adapter LSM(s), Link state transition messages for RDI, and Link Training related messages). In all cases, because at most one or two outstanding messages are permitted for each operation, it is relatively easy to provide greater than or the same number of buffers to sink from RDI. For example, in the scenario shown in [Figure 7-14](#), Physical Layer Die 1 must ensure that it has dedicated space to sink the request in Step 6 independent of any ongoing remote register access request or completion from Die 1 to Die 0, or any other sideband message for state transition, etc. Similarly, Physical Layer Die 1 must have dedicated space for remote die register access completion in Step 7.

Dynamic coarse clock gating is permitted in Adapter or Physical Layer in a subset of the RDI states (see [Chapter 10.0](#)). Thus, when applicable, any sideband transfer over RDI or FDI must follow the clock gating exit handshake rules as defined in [Chapter 10.0](#). It is recommended to always perform the clock exit gating handshakes for sideband transfers if implementations need to decouple dependencies between the interface status and sideband transfers.

Implementations of the Physical Layer and Adapter must ensure that there is no receiver buffer overflow for messages being sent over the UCIe sideband Link. This can be done by either ensuring that the time to exit clock gating is upper bounded and less than the time to transmit a sideband packet over the UCIe sideband Link, OR that the Physical Layer has sufficient storage to account for the worst-case backup of each sideband message function (i.e., remote register access requests, remote register access completions, Link state transition messages for Adapter LSM(s), Link state transition messages for RDI, and Link Training related messages). The latter offers more-general interoperability at the cost of buffers.

7.1.4 Operation on RDI and FDI

The same formats and rules of operation are followed on the RDI and FDI. The protocol is symmetric — requests, completions, and messages can be sent on **lp_cfg** as well as on **pl_cfg** signals. Implementations must ensure deadlock-free operation by allowing a sufficient quantity of sideband packets to sink and unblock the sideband bus for other packets. At the interface, these transactions are packetized into multiple phases depending on the configuration interface width (compile time parameter). Supported interface widths are 8, 16, or 32 bits. **lp_cfg_vld** and **pl_cfg_vld** are asserted independently for each phase. They must be asserted on consecutive clock cycles for transferring consecutive phases of the same packet. They may or may not assert on consecutive clock cycles when transferring phases of different packets. For packets with data, 64b of data is always transmitted over RDI or FDI; for 32b of valid payload, the most-significant 32b (Phase 4) of the packet are assigned to 0b before transfer.

§ §

8.0 System Architecture

8.1 UCIe Manageability

8.1.1 Overview

UCIe Manageability is optional and defines mechanisms to manage a UCIe-based SiP that is independent of UCIe mainband protocols. This accelerates the construction of a UCIe-based SiP by allowing a common manageability architecture and hardware/software infrastructure to be leveraged across implementations.

Examples of functions that may be performed using UCIe Manageability include the following:

- Discovery of chiplets that make up an SiP and their configuration,
- Initialization of chiplet structures, and parameters (i.e., serial EEPROM replacement),
- Firmware download,
- Power and thermal management,
- Error reporting,
- Performance monitoring and telemetry,
- Retrieval of log and crash dump information,
- Initiation and reporting of self-test status,
- Test and debug, and
- Various aspects of chiplet security.

UCIe manageability has been architected with the following goals:

- Support for management using mainband or sideband is mainband protocol agnostic allowing it to be used with chiplets that implement existing mainband protocols, mainband protocols that may be standardized in the future, and vendor-specific mainband protocols.
- The required core capabilities of UCIe manageability may be realized in hardware allowing simple chiplets to remain simple while providing advanced manageability capabilities (i.e., those that may require firmware implementation) to be implemented in chiplets that require these capabilities.
- UCIe Chiplets that support manageability may be used to realize products for a variety of markets. These markets may have vastly different manageability and security requirements. UCIe manageability defines a menu of optional management and security capabilities that build on top of the required core capabilities.
- UCIe manageability is intended to foster an open chiplet ecosystem where SiPs may be constructed from chiplets produced by different vendors. This means that common features and capabilities that are generally useful across market segments are standardized. Mechanisms are supplied for vendor-specific extensions.

- Manageability capabilities are discoverable and configurable, allowing a common firmware base to be rapidly used across SiPs.
- UCIe manageability builds on top of applicable industry standards.

8.1.2 Theory of Operation

A UCIe-based System-in-Package (SiP) that supports manageability contains one or more UCIe Chiplets that support UCIe manageability. Chiplets in the SiP that support UCIe manageability form a Management Domain. The SiP may also contain chiplets that do not support UCIe manageability and these chiplets are outside the Management Domain. If the Management Domain contains more than one chiplet, then the chiplets are interconnected through Management Ports using chiplet-to-chiplet management links to form a Management Network. The Management Network is fully connected meaning that there is a path from any chiplet in the Management Domain to any other chiplet in the Management Domain.

A UCIe Chiplet that supports manageability contains a Management Fabric and one or more Management Entities. A Management Entity is a Management Element, a Management Port, or a Management Bridge. An example UCIe chiplet that supports manageability is shown in [Figure 8-1](#) and an example SiP that supports manageability consisting of four UCIe chiplets is shown in [Figure 8-2](#).

Figure 8-1. Example UCIe Chiplet that Supports Manageability

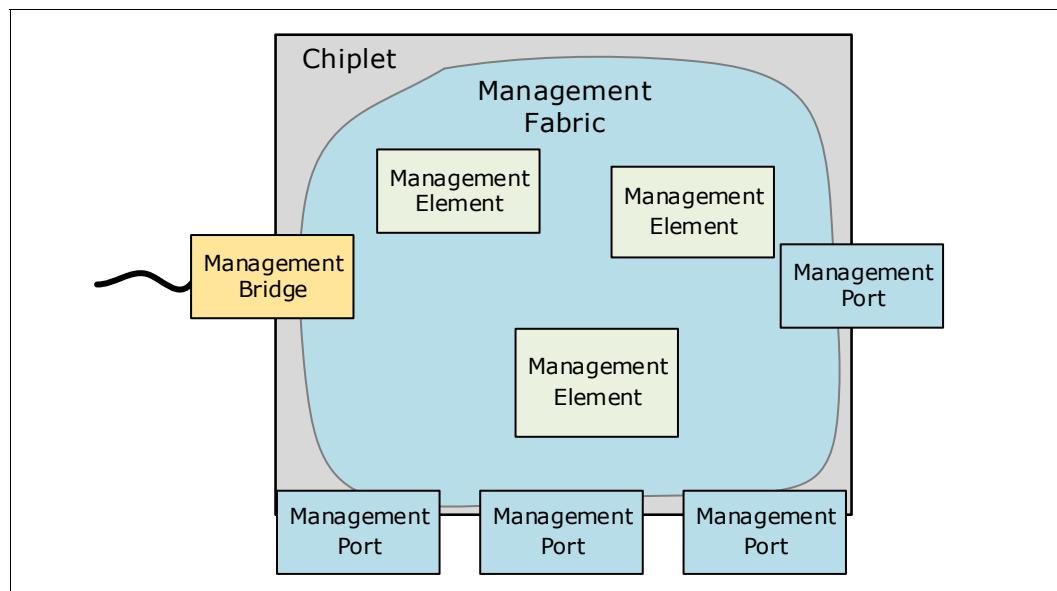
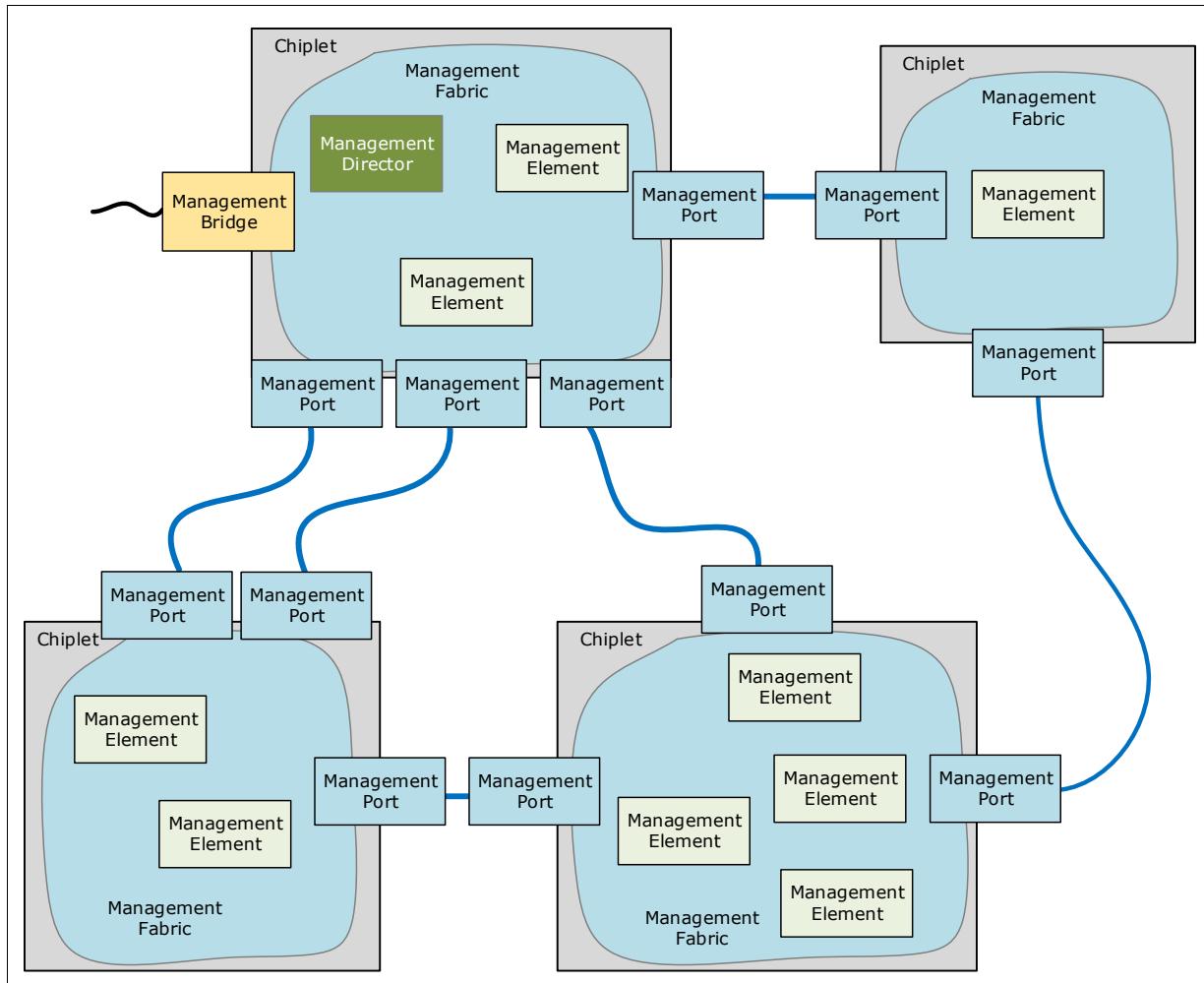
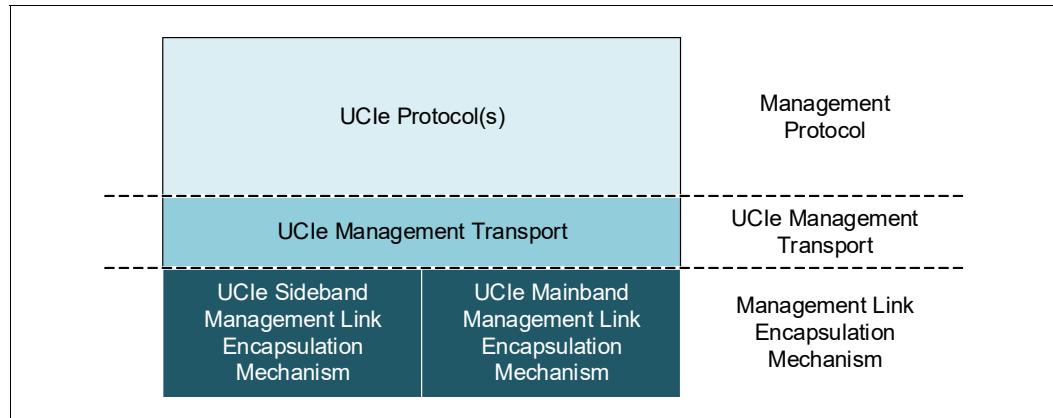


Figure 8-2. Example SiP that Supports Manageability

The UCIe Management Transport is an end-to-end media-independent protocol for management communication on the Management Network. This includes between Management Entities within a chiplet as well as between Management Entities in different chiplets. As shown in [Figure 8-3](#), the Management Protocols above the UCIe Management Transport are used to implement management services. An example of a Management Protocol is the UCIe Memory Access Protocol.

Figure 8-3. UCIe Manageability Protocol Hierarchy

A Management Port is a Management Entity that acts as the interface between the Management Fabric within a chiplet and a point-to-point management link that interconnects two chiplets. As shown in [Figure 8-3. UCIe Manageability Protocol Hierarchy](#), below the UCIe Management Transport is a Management Link Encapsulation Mechanism that defines how UCIe Management Transport packets are transferred across a point-to-point management link. Two Management Link Encapsulation Mechanisms are defined, one for the UCIe sideband and one for the UCIe mainband. See [Section 8.2](#) for additional details of Management Link Encapsulation Mechanisms. Whether a specific UCIe sideband or mainband link in a chiplet may function as a point-to-point management link is implementation specific.

A chiplet that supports manageability should support at least one UCIe sideband Management Port. To enable broad interoperability, it is strongly recommended that a chiplet support enough UCIe sideband Management Ports to enable construction of an SiP with a single management domain using only UCIe sideband. If a chiplet supports management applications that require high bandwidth, such as test, debug, and telemetry, then it is strongly recommended that the chiplet support UCIe mainband Management Ports.

A Management Fabric within a UCIe chiplet facilitates communication between Management Entities inside the chiplet. A Management Entity is a Management Element, a Management Port, or a Management Bridge. The Management Fabric may be realized using one or more on-die fabrics and the implementation of the Management Fabric is beyond the scope of this specification.

A Management Element is a Management Entity that implements a management service. A Management Element must support the UCIe Management Transport protocol and one or more Management protocols.

A Management Bridge is a Management Entity that interconnects the Management Network to another interconnect, allowing agents on the Management Network and the other interconnect to communicate. The interconnect associated with a bridge may be internal to an SiP or external to the SiP.

One of the Management Elements within an SiP is designated as the Management Director. An SiP may contain multiple Management Elements that may act as a Management Director; however, there can only be one active Management Director at a time. How the Management Director is selected in such SiPs is beyond the scope of this specification. The roles of the Management Director include the following:

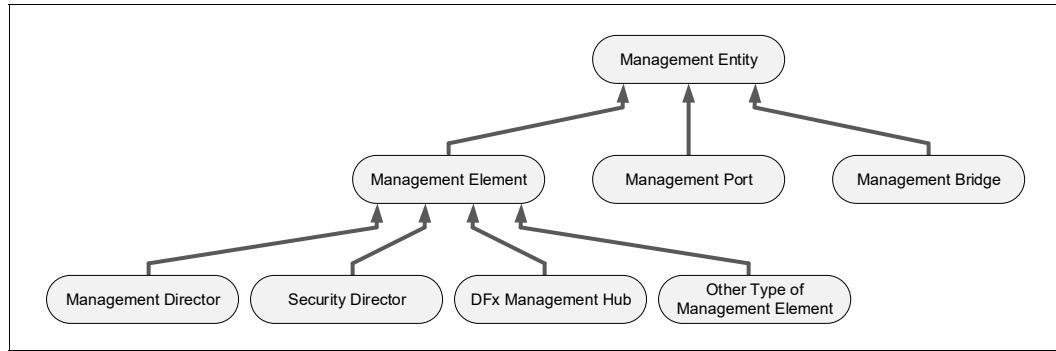
- Discovering chiplets and configuring Chiplet IDs,
- Discovering and configuring Management Elements,

- Discovering and configuring Management Ports,
- Discovering and configuring Management Bridges,
- Acting as the manageability Root of Trust (RoT), and
- Coordinating overall management of the SiP.

One or more of the Management Elements within an SiP may function as a Security Director. A Security Director is responsible for configuring security parameters.

The relationship between the various type of Management Entities is shown in [Figure 8-4](#).

Figure 8-4. Relationship Between the Various Types of Management Entities



Unless otherwise specified, UCIe manageability, Management Entities, and all associated manageability structures in a chiplet (e.g., those in a Management Capability Structure) are reset on a Management Reset. A Management Reset occurs on initial application of power to a chiplet. Other conditions that cause a Management Reset are implementation specific.

8.1.3 UCIe Management Transport

The UCIe Management Transport is a protocol that facilitates communication between Management Entities on an SiP's Management Network. UCIe Management Transport packets are produced and consumed by Management Entities on the Management Network and the packets flow unmodified through the network.

8.1.3.1 UCIe Management Transport Packet

[Figure 8-5](#) shows the fields that make up a UCIe Management Transport packet. The first two DWORDs contain the packet header. This is followed by a protocol specified field defined by the Management Protocol carried in the packet. Finally, a UCIe Management Transport packet may optionally contain a Packet Integrity value computed over the previous contents of the packet. If present, Packet Integrity is one DWORD in size.

Reserved fields in a UCIe Management Transport packet must be filled with 0s when the packet is formed. Reserved fields must be forwarded unmodified on the Management Network and ignored by receivers. An implementation that relies on the value of a reserved field in a packet is non-compliant.

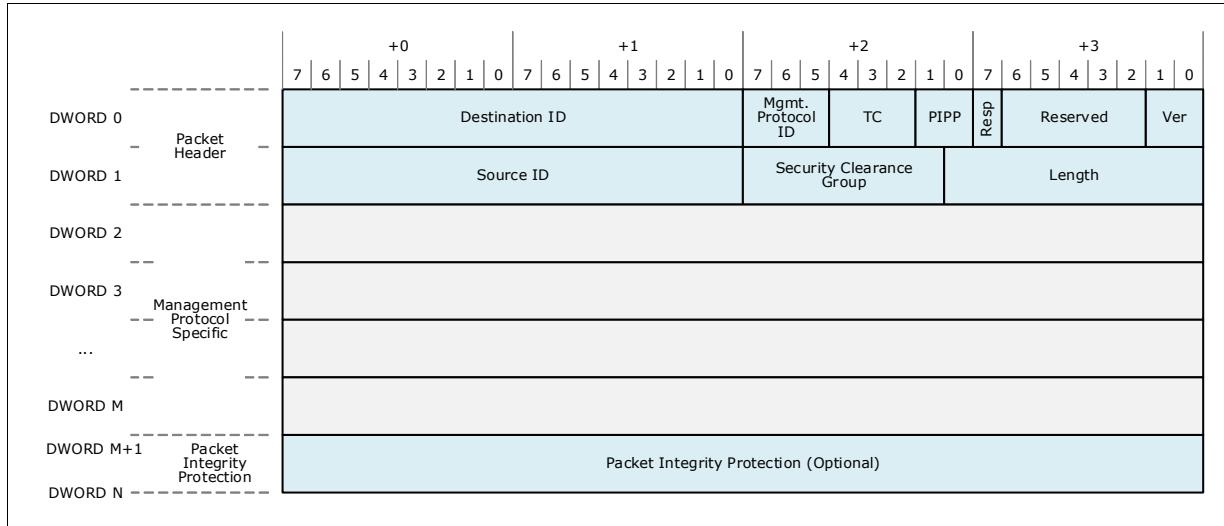
Figure 8-5. UCIe Management Transport Packet

Table 8-1 defines the fields of a UCIe Management Transport packet. All fields in the table have little endian bit ordering (e.g., Destination ID bits 0 through 7 are in Byte 1 with bit 0 of the Destination ID in Byte 1 bit 0, and Destination ID bits 8 through 15 are in Byte 0 with Destination ID bit 8 in Byte 0 bit 0).

Table 8-1. UCIe Management Transport Packet Fields (Sheet 1 of 2)

Field Name	Field Size	Description
Destination ID	16 bits	Destination ID This field specifies the Management Network ID of the entity on the Management Network that is to receive the packet.
Mgmt. Protocol ID	3 bits	Management Protocol ID This field contains an ID that corresponds to a Management Protocol and specifies the type of payload contained in the Management Protocol Specific field. See Section 8.1.3.1.3 for ID values.
TC	3 bits	Traffic Class This field is a packet label used to enable different packet servicing policies. Each Traffic Class is a unique ordering domain with no ordering guarantees between packets in different Traffic Classes. See Section 8.1.3.1.1 .
PIPP	2 bits	Packet Integrity Protection Present (PIPP) A nonzero value in this field indicates that the packet contains a packet integrity protection value in the Packet Integrity Protection field. The type of packet integrity is indicated by the nonzero value. See Section 8.1.3.4 for more information. 00b: Packet Integrity Protection field is not present in the packet 11b: CRC Integrity (one DWORD in size) Others: Reserved
Resp	1 bit	Request or Response This field indicates whether the packet is a request or a response. 0: Request packet 1: Response packet
Reserved	5 bits	Reserved
Ver	2 bits	UCIE Management Transport Packet Version This field indicates the version of the UCIe Management Transport packet. This field must be set to 00b in this version of the specification. If a Management Entity receives a packet with a UCIe Management Transport packet version that it does not support, then Management Entity must discard the packet.

Table 8-1. UCIe Management Transport Packet Fields (Sheet 2 of 2)

Field Name	Field Size	Description
Source ID	16 bits	Source ID This field indicates the Management Network ID of the entity on the Management Network that originated the packet.
Security Clearance Group	7 bits	Security Clearance Group This field is used by the UCIe Management Transport access control mechanism. In request packets, this field is set to the security clearance group value associated with the request. In response packets, this field is not used and must be set to 00h.
Length	9 bits	Length This field indicates the length of the entire packet in DWORDs. This includes the UCIe Management Network Header, the Management Protocol Specific field, and the Packet Integrity Protection field if present. The length of the packet in DWORDs is equal to the value of this field plus 1 (e.g., a value of 000h in this field indicates a packet length of one DWORD, a value of 001h in this field indicates a packet length of two DWORDs, and so on).
Management Protocol Specific	Varies	Management Protocol Specific This field contains Management Protocol specific packet contents. The format of this field is indicated by the Management Protocol field. This field must be an integral number of DWORDs.
Packet Integrity Protection	Varies	Packet Integrity Protection If the Packet Integrity Protection Present (PIPP) field in the packet has a nonzero value, then this field is present in the packet and corresponds to the packet integrity value. See Section 8.1.3.4 for more information. This field must be an integral number of DWORDs. In this version of the specification only CRC integrity protection is supported which is always one DWORD.

A request packet is a packet with the Resp field in the UCIe Management Transport packet header assigned to 0. A requester is a Management Entity that first introduces a request packet into a Management Fabric. A responder is the Management Entity that performs the actions associated with a request that consists of one or more request packets and is the ultimate destination of these request packet(s). A response packet is a packet with the Resp field in the UCIe Management Transport packet header assigned to 1. As a result of performing the actions associated with a request, the responder may introduce one or more response packets into the Management Fabric destined to the requester.

A Management entity that receives a malformed packet must discard the packet.

- A packet with an incorrect length in the Length field is malformed.
 - An example of a malformed packet with an incorrect length in the Length field is one where the Length field in the UCIe Management Transport packet header indicates a length of 65 DWORDs but the actual length of the packet is 64 DWORDs.
- A packet whose size exceeds the Management Transport Packet size supported by a chiplet is malformed.
- A packet that violates a requirement outlined in this specification is malformed.
 - An example of a malformed packet due to a requirement violation is a response packet with a nonzero value in the Security Clearance Group field.

8.1.3.1.1 Traffic Class and Packet Ordering Requirements

Traffic classes (TCs) are used to enable different packet servicing policies. The UCIe Management Transport supports eight traffic classes, and the characteristics of each traffic class are described in [Table 8-2](#). The Management Director configures management fabric routes and may configure different routes for different traffic classes (e.g., TC0 traffic may be routed to UCIe sideband Management Port A and TC2 traffic may be routed to UCIe mainband Management Port B).

Table 8-2. Traffic Class Characteristics

Traffic Class	Description
0	Default Ordered Lossless Traffic Class Traffic class optimized for packets that require high reliability and availability. Example: Configuration and health monitoring packets
1	Low Latency Ordered Lossless Traffic Class This traffic class has the same characteristics as the Default Ordered Lossless Traffic Class but is intended to be lightly loaded and used for packets that require low latency and should bypass congested Default Ordered Lossless Traffic Class packets. Example: Debug cross trigger and low latency power management packets
2	Bulk Lossless Ordered Performance Traffic Class Traffic class optimized for packets associated with bulk traffic. In a properly functioning system without errors, packets in this traffic class are never dropped. Example: Firmware download packets
3	Bulk Lossy Ordered Performance Traffic Class Traffic class optimized for packets associated with bulk traffic that may be dropped in the presence of congestion. Example: Debug trace packets
4	Unordered Lossless Traffic Class Traffic class optimized for packets that require high performance. Request packets in this traffic class may be delivered out-of-order. Response packets in this traffic class may be delivered out-of-order. Example: High performance PCIe Memory Access protocol accesses.
5 to 6	Reserved
7	Vendor-Specific Traffic Class

A request packet and an associated response packet need not have the same traffic class. Which traffic classes are allowed and how they are mapped between a request and response are Management Protocol specific.

An implementation shall ensure forward progress on all traffic classes. Quality of service between traffic classes is implementation specific and beyond the scope of this specification.

Each traffic class is a unique ordering domain and there are no ordering guarantees for packets in different traffic classes and there are no ordering guarantees between request and response packets in the same traffic class. Regardless of the traffic class, a response packet associated with any traffic class must be able to bypass a blocked request packet associated with any traffic class.

Within an ordered traffic class, request packets are delivered in-order from a requester to a responder and response packets are delivered in-order from a responder to the requester. There are no ordering guarantees between requests to different responders and there are no ordering guarantees between responses from different responders to a requester.

Within an unordered traffic class there are no ordering guarantees between packets of any type and the chiplet's Management Fabric is free to arbitrarily reorder packets. While packets may be reordered on an unordered traffic class, there is no requirement that they be reordered (i.e., an implementation is free to maintain ordering as in an ordered traffic class).

Packets associated with a lossy traffic class may be dropped during normal operation. The policy used to determine when a lossy traffic class packet is dropped is vendor defined and beyond the scope of this specification (e.g., due to exceeding a vendor defined congestion threshold). Lossless traffic classes are reliable, and packets associated with a lossless traffic class are not dropped during normal operation; however, packets associated with a lossless traffic class may be dropped due to an error

condition. The detection of lost packets and recovery from lost packets is the responsibility of a Management Protocol.

To maintain forward compatibility, a UCIe Management Transport packet with a reserved traffic class value is treated in the same manner as a packet with a traffic class value of zero (i.e., Default Ordered Lossless Traffic Class).

To maintain interoperability, all implementations are required to support all traffic classes; however, an implementation is only required to maintain the ordered and lossless characteristics of a traffic class. All other traffic class characteristics may be ignored. For example, an implementation may treat all traffic classes in the same manner as the Default Ordered Lossless Traffic Class. Under no circumstances may packets in an ordered traffic class be reordered between a requester and a responder. Similarly, under no circumstances may a packet in a lossless traffic class be dropped in a properly functioning system without errors.

IMPLEMENTATION NOTE — CHIPLET ROUTE THROUGH

Chiplet route through occurs when a UCIe Management Transport packet flows through a chiplet (i.e., the packet enters a chiplet through a UCIe Management Port, is not destined to any Management Entity within the chiplet, and the packet matches a Route Entry that causes it to be routed out a UCIe Management Port). Due to chiplet route through it is desirable that chiplets implement the packet servicing policies associated with a TC even if none of the Management Entities within the chiplet utilize that TC. Chiplets are always required to support chiplet route through for all traffic classes.

8.1.3.1.2 Packet Length

The length of a Packet is indicated by the Length field, is an integral number of DWORDs, and consists of the entire length of the packet (i.e., the UCIe Management Network Header, the Management Protocol Specific field, and Packet Integrity Protection field if present.).

The maximum packet length architecturally supported by the UCIe Management Transport is 512 DWORDs (i.e., 2048B). A chiplet may support a maximum packet length that is less than the architectural limit. The Maximum Packet Size (MPS) field in the Chiplet Capability Structure indicates the maximum packet size supported by the chiplet. If a packet larger than that advertised by MPS is received on a Management Port or Management Bridge, then it is silently discarded and not emitted on the chiplet's Management Fabric.

The Configured Maximum Packet Size (CMPS) field in the Chiplet Capability Structure controls the maximum UCIe Management Transport packet size generated by a Management Entity within the chiplet. The initial value of this field corresponds to 8 DWORDs (i.e., 64B). The CMPS field does not affect UCIe Management Transport packets emitted by Management Ports and Management Bridges when forwarding packets into a chiplet's Management Fabric. This allows packets to be routed through the chiplet (e.g., between Management Ports) that are larger than the size of packets generated by Management Entities within the chiplet.

8.1.3.1.3 Management Protocol

The Management Protocol field in a packet contains a Management Protocol ID that indicates the format of the Management Protocol Specific field. [Table 8-3](#) lists the Management Protocols supported by the UCIe Management Transport.

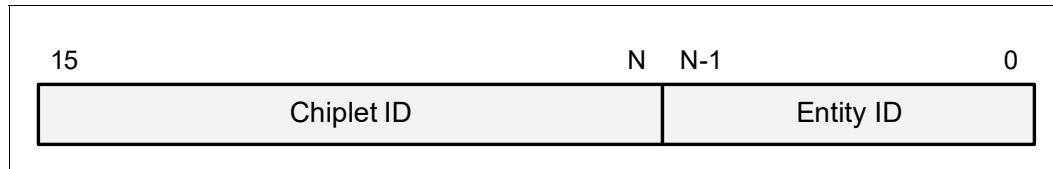
Table 8-3. Management Protocol IDs

Management Protocol ID	Description
0	Reserved
1	PCIe Memory Access Protocol This protocol is used to access memory mapped structures and memory.
2	PCIe Test and Debug Protocol
3 to 6	Reserved
7	Vendor defined

8.1.3.2 Management Network ID

Management Network IDs are used to uniquely identify Management Entities in the Management Network and to route PCIe Management Transport packets between Management Entities.

As shown in [Figure 8-6](#), a Management Network ID is a 16-bit field that consists of the concatenation of a Chiplet ID and an Entity ID. The Chiplet ID uniquely identifies a chiplet within an SiP and the Entity ID is a fixed value that uniquely identifies a Management Entity within a chiplet. Together the Chiplet ID and Entity ID uniquely identify each Management Entity in an SiP.

Figure 8-6. Management Network ID Format

The size of the Chiplet ID in bits is chiplet implementation specific and may be 2-bits to 15-bits in size. The size of the Entity ID in bits is also chiplet implementation specific and may be 1-bit to 14-bits in size. For each chiplet, the sum of the size of the Chiplet ID and the Entity ID must be 16-bits. The size of the Chiplet ID and Entity ID fields may be different in different chiplets in an SiP (i.e., it is not a requirement that all chiplets of an SiP have the same Chiplet ID field size). To facilitate interoperability an implementation should make the Chiplet ID field as large as possible (i.e., make the Entity ID field only as large as needed to address Management Entities in the chiplet).

The Management Director initializes the Chiplet ID value associated with each chiplet during SiP initialization. This is done by writing the Chiplet ID value to the Chiplet field in the Chiplet Capability Structure and setting the Chiplet ID Valid bit in that structure. The Management Director may determine the size of the Chiplet ID associated with a chiplet by examining the initial value of the Chiplet ID field in the Chiplet Capability Structure or by determining which bits are read-write in this field.

Each Management Entity within a chiplet has a fixed Entity ID. Entity ID zero within each chiplet must be a Management Element that is used to initialize and configure the chiplet. Entity IDs within a chiplet that map to Management Entities may be sparse. For example, a chiplet may contain Management Entities at Management Entity IDs zero, one, and three. The other Entity IDs are unused. A PCIe Management Packet that targets an unused Entity ID within a chiplet is silently discarded by the chiplet's Management Fabric.

IMPLEMENTATION NOTE — CONFIGURING ROUTING IN AN SiP WITH DIFFERENT CHIPLET ID SIZES

A System-in-Package (SiP) may be composed of chiplets with varying sizes for the Chiplet ID portion of the Management Network ID. To establish routing within such an SiP, one approach is to identify the smallest Chiplet ID size among all chiplets in the SiP. Subsequently, Route Entries (as described in [Section 8.1.3.6.2.2](#)) are configured as if each chiplet possessed a Chiplet ID size equivalent to this minimum. For chiplets with a Chiplet ID size exceeding the minimum, any unused Chiplet ID bits are assigned 0 in the Base ID field of a Route Entry and set to 1 in the Limit ID field of a Route Entry.

8.1.3.3 Routing

UCIE Management Transport packets are routed based on the Destination ID field in the UCIE Management Network Header.

The routing of UCIE Management Transport packets differs depending on whether the Chiplet ID value is initialized or uninitialized. The Chiplet ID value is initialized when the Chiplet ID Valid bit is set to 1 in the Chiplet Capability Structure. The Chiplet ID value is uninitialized when the Chiplet ID Valid bit is cleared to 0 in the Chiplet Capability Structure.

A UCIE Management Transport request packet is one where the Resp field is cleared to 0. A UCIE Management Transport response packet is one where the Resp field is set to 1.

While the Chiplet ID and Entity ID size of chiplets may vary in an SiP, all packet routing associated with a chiplet is performed using the Chiplet ID size of the chiplet performing the routing.

The method used to configure UCIE Management Transport routing within an SiP is beyond the scope of this specification. The routing may be pre-determined during SiP design and this static configuration may be used by the Management Director to configure routing. Alternatively, the Management Director may discover the SiP configuration (i.e., chiplets, Management Network topology, and Chiplet ID size of each chiplet) and use this information to dynamically configure routing.

Because the Management Network may contain redundant management links between chiplets as well as links that form cycles, care must be exercised to ensure that the UCIE Management Transport routing is acyclic and deadlock free. In the absence of faults, request packets and response packets are delivered in order; however, it is possible to configure UCIE Management Transport routing such that the path used by a request packet from a requester to a responder is different from path used by a response packet from the responder back to the requester.

8.1.3.3.1 Routing of a Packet from a Management Entity within the Chiplet

This section describes routing of a UCIE Management Transport packet generated by a Management Entity within the chiplet.

- If the chiplet's Chiplet ID value is initialized, then the packet is routed as follows.
 - If the Chiplet ID portion of the packet's Destination ID matches the Chiplet ID value of the chiplet performing the routing, the packet is routed within the chiplet based on the Entity ID portion of the packet's Destination ID.
 - If the Entity ID portion of the packet's Destination ID matches that of a Management Entity within the chiplet, then the packet is routed to that Management Entity.

- o If the Entity ID portion of the packet's Destination ID does not match that of any Management Entity within the chiplet, then the packet is discarded.
- If the Chiplet ID portion of the packet's Destination ID does not match the Chiplet ID value of the chiplet, then the packet is routed based on Management Port Route Entries.
 - o If the packet's Destination ID matches a Route Entry associated with a Management Port, then the packet is routed out that Management Port. See [Section 8.1.3.6.2.2](#) for Route Entry matching rules.
 - o If the packet's Destination ID matches multiple Route Entries within the chiplet and the packet is associated with an ordered traffic class, then the packet is discarded.
 - o If the packet's Destination ID matches multiple Route Entries within the chiplet and the packet is associated with an unordered traffic class, then the packet is routed out one of the Management Ports with a matching Route Entry. Which matching Route Entry is selected in the unordered traffic class case is vendor defined.
 - o If the packet's Destination ID does not match any Route Entries within the chiplet, then the packet is discarded.
- If the chiplet's Chiplet ID value is uninitialized, then packet is routed as follows.
 - If the packet is a UCIe Management Transport Response packet and the corresponding UCIe Management Transport Request packet was received from a Management Port, then the packet is routed as follows.
 - o If the link is up that is associated with Management Port on which the corresponding UCIe Management Transport Request packet was received, then the response packet is routed out that same Management Port on virtual channel zero (VC0). This allows a Management Entity outside the chiplet to configure the chiplet before the chiplet's Chiplet ID value is initialized.
 - o If the link associated with Management Port on which the corresponding UCIe Management Transport Request packet was received is not up, then the response packet is discarded.
 - Otherwise, the packet is routed within the chiplet based on the Entity ID portion of the packet's Destination ID.
 - o If the Entity ID portion of the packet's Destination ID matches that of a Management Entity within the chiplet, then the packet is routed to that Management Entity.
 - o If the Entity ID portion of the packet's Destination ID does not match that of any Management Entity within the chiplet, then the packet is discarded.

IMPLEMENTATION NOTE — ROUTING FOR UNINITIALIZED CHIPLET IDs

When a chiplet's Chiplet ID value is uninitialized, then a packet received on a chiplet's Management Port is routed based on the Entity ID portion of the packet's Destination ID to a Management Entity within the chiplet, if one exists. When a response packet is emitted by this Management Entity, the packet is routed out the same Management Port on which the corresponding request was received. This allows an agent external to the chiplet to configure the chiplet before the Chiplet ID and Route Entries are initialized (e.g., following a Management Reset).

When a response packet is routed out the Management Port on which the corresponding request was received, it is routed out the Management Port on virtual channel zero. While there is no requirement that requests to uninitialized chiplets use virtual channel zero (VC0), it is strongly encouraged that virtual channel zero be used.

8.1.3.3.2 Routing of a Packet Received on a Management Port

This Section describes routing of a UCIe Management Transport packet received on a chiplet's Management Port.

- If the chiplet's Chiplet ID value is initialized, then the packet is routed in the same manner as a packet generated by a Management Entity within the chiplet. See [Section 8.1.3.3.1](#) for how such a packet is routed.
- If the chiplet's Chiplet ID value is uninitialized, then the packet is routed within the chiplet based on the Entity ID portion of the packet's Destination ID.
 - If the Entity ID portion of the packet's Destination ID matches that of a Management Entity within the chiplet, then the packet is routed to that Management Entity.
 - If the Entity ID portion of the packet's Destination ID does not match that of any Management Entity within the chiplet, then the packet is discarded.

8.1.3.4 Packet Integrity Protection

A UCIe Management Transport packet may optionally contain a Packet Integrity Protection field that is used to protect the integrity of the packet. The presence and type of packet integrity are indicated by the Packet Integrity Protection Present (PIPP) field in the packet.

CRC protection, defined in [Section 8.1.3.4.1](#), is the only packet integrity protection currently defined and is one DWORD in size.

8.1.3.4.1 CRC Integrity Protection

When the PIPP field in a packet is set to 11b, then the Packet Integrity Protection field in the packet is one DWORD in size and contains a 32-bit CRC computed over the previous contents of the packet (i.e., the UCIe Management Network Header and Management Protocol Specific field). Each bit of the Packet Integrity Protection field is set to the corresponding bit of the computed CRC (e.g., bit 31 of the computed CRC corresponds to bit 31 of the Packet Integrity Protection field).

The 32-bit CRC required by this specification is CRC-32C (Castagnoli) which uses the generator polynomial 1EDC6F41h. The CRC is calculated using the following Rocksoft™ Model CRC Algorithm parameters:

Name : "CRC-32C"
Width : 32

```

Poly : 1EDC6F41h
Init : FFFFFFFFh
RefIn : True
RefOut : True
XorOut : FFFFFFFFh
Check : E3069283h

```

When the PIPP field in a packet is set to 11b and the CRC contained in the Packet Integrity Protection field is incorrect, then the packet is discarded.

8.1.3.5 Access Control

The Management Network in an SiP may contain multiple Management Entities that issue requests and multiple Management Entities that expose assets (e.g., structures, keys, or memory). The UCIe Management Transport supports an access control mechanism that may be used by a Management Protocol to prevent unauthorized access to assets by Management Entities on the Management Network.

Some Management Protocols have their own security architecture, so the use of the access control mechanism is Management Protocol specific. [Table 8-4](#) shows which Management Protocols use the access control mechanism.

Table 8-4. Management Protocol use of Access Control Mechanism

Management Protocol	Uses Access Control Mechanism
UCIe Memory Access Protocol	Yes
UCIe Test and Debug Protocol	Yes ^a
Vendor Defined	Vendor Defined

a. The UCIe Test and Debug Protocol uses the UCIe Memory Access Protocol for configuration and status field accesses and as a result uses the Access Control Mechanism.

The access control mechanism is based on a 7-bit security clearance group value contained in request packets. When a Management Entity emits a request packet on the Management Network, it populates the Security Clearance Group field in the packet with a value that corresponds to the security clearance group associated with the requester. When a Management Entity receives a request packet, it determines whether the asset(s) accessed by the request are allowed or prohibited by that security clearance group.

A Management Entity may emit packets with different security clearance group values. How the security clearance group values that a Management Entity emits are configured or selected is beyond the scope of this specification (see [Section 8.1.3.6.4.1](#)).

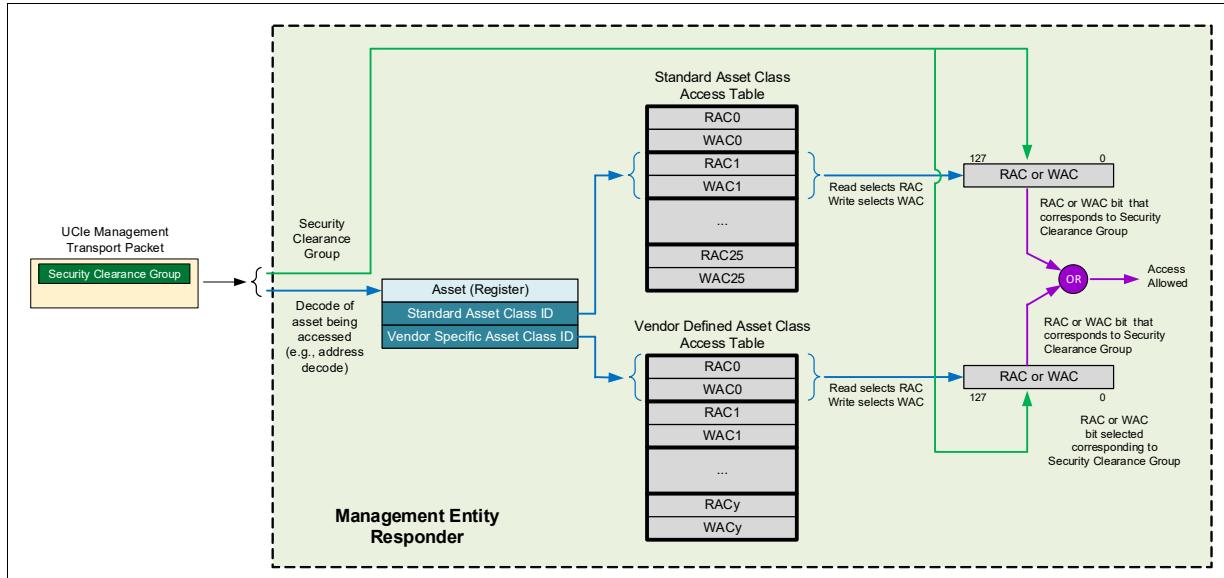
Although the security clearance group value is seven bits in size, an implementation may choose to restrict the number of security clearance groups. When an implementation restricts the number of security clearance groups to a value N, then security clearance group values from 0 to (N-1) are allowed, and security clearance group values from N to 127 are not allowed. Security Clearance Group 0 is dedicated for use by Security Directors (see [Section 8.1.3.5.2](#)).

The method a Management Entity uses to determine whether a request packet is allowed access to an asset is shown in [Figure 8-7](#) and consists of the following steps.

1. The standard asset class ID or the vendor defined asset class ID of the asset being accessed by the request packet is determined.

- UCIe defines standard asset classes (see [Section 8.1.3.5.1](#)) and supports vendor defined asset classes. Each asset must map to a standard asset class, a vendor defined asset class, or both.
 - Associated with each asset class is an asset class ID. The mapping associated with this step produces a standard asset class ID, a vendor defined asset class ID, or both.
 - The manner and granularity in which an implementation maps assets to asset class IDs are beyond the scope of this specification and may be done as part of address decoding, tagging of assets, or some other mechanism.
2. Each asset class ID determined in the previous step is mapped to a 256-bit access control structure.
- Associated with each asset class ID is a 128-bit Read Access Control (RAC) structure and a 128-bit Write Access Control (WAC) structure. If an asset's state is being read, then the RAC structure is selected as the access control structure. If an asset's state is being written/modified, then the WAC structure is selected as the access control structure.
 - o RAC and WAC structures are contained in the Access Control Capability Structure (see [Section 8.1.3.6.3](#)).
 - o If a Management Entity does not have any assets that correspond to an asset class ID, then the RAC and WAC structures associated with that asset class ID must be hardwired to 0 (i.e., all the bits on both the RAC and WAC structures are read-only with a value of 0) so no accesses would be allowed.
 - o If an implementation restricts the number of security clearance groups, then RAC and WAC structure bits associated with security clearance groups that are not supported must be hardwired to 0. For example, if an implementation only supports Security Clearance Groups 0 through 3, then bits 4 through 127 must be read-only with a value of 0 in all RAC and WAC structures.
3. The bit corresponding to the security clearance group value in the request packet is examined in each access control structure determined in the previous step to determine whether access to the asset is allowed.
- The 7-bit security clearance group value in the request packet is a value from 0 to 127 and this value maps to a corresponding bit in a 128-bit access structure (e.g., security clearance group value 27 maps to access control structure bit 27).
 - If a bit corresponding to the security clearance group value in an access control structure is set to one, then access to that asset is allowed by that security clearance group. If a bit corresponding to the security clearance group value in an access control structure is set to 0, then access to that asset is prohibited by that security clearance group.
4. If access to the asset is allowed by either the standard asset class or the vendor defined asset, then access to the asset is granted and the request is processed. If access to the asset class is prohibited by both the standard asset class and a vendor defined asset class, then access to the asset is prohibited and the request is not processed.
- How a request that attempts to access a prohibited asset is handled is Management Protocol specific.

If a request packet requires access to multiple assets for the request to be serviced, then [Step 1](#) through [Step 3](#) are performed and the request is processed only if access is granted to all assets. If access is prohibited to any asset associated with the request, then no asset is accessed by the request.

Figure 8-7. Access Control Determination in a Responder Management Entity

8.1.3.5.1 Standard Asset Classes

The objective of the standard asset classes is to provide a consistent classification of assets across chiplet implementations and applications for access control. To achieve this, it must be possible for a chiplet implementer to map chiplet assets that are accessible over the Management Network into asset classes without understanding the underlying architecture, roles, or applications associated with an SiP that uses the chiplet.

Standard asset class 0 is for SiP security configuration (e.g., reading and writing the RAC and WAC structures). The remaining standard asset classes are constructed by taking the Cartesian product of a set of asset types and asset contexts and removing elements that are not applicable in practice. Asset types used to construct the standard asset class are shown in [Table 8-5](#) and asset contexts used to construct the standard asset class are shown in [Table 8-6](#). The standard asset classes are shown in [Table 8-7](#).

In some cases, an asset may logically map to two or more standard asset classes. For example, a memory region may contain both SiP data and chiplet data. When this occurs, the asset should be mapped to the standard asset class with the lowest ID value.

In some cases, further access control granularity may be desired beyond what is offered by the standard asset classes. This granularity may be accomplished by separating the assets into different Management Elements within the chiplet. For example, a chiplet may contain two global secrets and the chiplet implementor may wish to allow one set of requesters access to the first global secret and a separate set of requesters access to the second global secret. By putting the two global assets into two different Management Elements, different security clearance groups may be given access to each global secret. In another example, a chiplet may contain an I/O controller and a memory controller and the chiplet implementor may wish to allow one security clearance group to configure the I/O controller and a different security clearance group to configure the memory controller. This granularity may be achieved by putting the I/O controller and memory controller in different Management Elements.

Table 8-5. Asset Types

Asset Type	Description
Persistent One-Time Secret	Data or configuration that is persistent, one-time programmable, and considered a secret or sensitive configuration. Example: one-time programmable device secret
Secret	A secret, data, or status that may compromise a secret, or configuration that may control the exposure of a secret. Example: security key
Permanent Denial of Service (PDOS)	Data or configuration that could potentially cause permanent denial of service. Example: thermal, power, or voltage controls
Sensitive	Volatile or persistent data that should have limited exposure, status that could expose information that should have limited exposure, or configuration that may control exposure or modification of sensitive information. Examples: error injection capabilities, sensitive state machines, private memory space
Permanent	Data or configuration that is one-time programmable and is not sensitive or a secret. Example: general fuses
Data	General data or user data that is not sensitive or a secret. Example: application space
Configuration	General configuration that cannot be used to expose user, sensitive, or secret information.
Status	General status that cannot be used to expose user, sensitive, or secret information. Example: boot status

Table 8-6. Asset Contexts

Asset Context	Description
Global	Asset associated with or which affects chiplets or SiPs produced by a manufacturer. The definition of the manufacturer is beyond the scope of the specification and may be the SiP integrator, the SiP designer, or an IP provider. All that matters is that the asset is the same in SiPs of that type (i.e., a global key).
SiP	Asset associated with or which affects a specific SiP.
Chiplet	Asset associated with or which affects a specific chiplet.
Partition	Asset associated with or which affects a partition. The definition of a partition is vendor defined but is broadly defined as a collection of hardware resources that act as an independent unit.

Table 8-7. Standard Security Asset Classes (Sheet 1 of 2)

Standard Security Asset Class ID	Asset Context	Asset Type
0	SiP	SiP Security Configuration
1	Global	Secret
2	SiP	Persistent One-Time Secret
3	SiP	Secret
4	SiP	Permanent Denial of Service (PDOS)
5	SiP	Sensitive
6	SiP	Permanent
7	SiP	Data
8	SiP	Configuration

Table 8-7. Standard Security Asset Classes (Sheet 2 of 2)

Standard Security Asset Class ID	Asset Context	Asset Type
9	SiP	Status
10	Chiplet	Permanent Secret
11	Chiplet	Secret
12	Chiplet	Permanent Denial of Service (PDOS)
13	Chiplet	Sensitive
14	Chiplet	Permanent
15	Chiplet	Data
16	Chiplet	Configuration
17	Chiplet	Status
18	Partition	Permanent Secret
19	Partition	Secret
20	Partition	Permanent Denial of Service (PDOS)
21	Partition	Sensitive
22	Partition	Permanent
23	Partition	Data
24	Partition	Configuration
25	Partition	Status

8.1.3.5.2 Security Director

A Management Element within an SiP that may configure security parameters is designated as a Security Director. An SiP may contain multiple Security Directors. When an SiP contains multiple Security Directors, coordination between security directors is beyond the scope of this specification.

The Security Clearance Group value of 0 is reserved for Security Directors and must not be used for any other purpose.

The Management Director may also be a Security Director. While it is not recommended that the Management Director operate using the Security Clearance Group value reserved for Security Directors (i.e., 0) during normal operation, it is required to operate with this value during initial configuration. When and how a Management Director changes the Security Clearance Group used for transactions is beyond the scope of this specification.

8.1.3.6 Initialization and Configuration

UCIE Management Transport initialization and configuration are performed through read and write operations using the UCIE Memory Access protocol to Management Entity fields. Management Entity fields are grouped by function into Management Capability Structures.

Unless otherwise specified, Management Capability Structures and any sub-structures must be read or written as single DWORD quantities (i.e., the Length field in the UCIE Memory Access Request must be 0h which represents a data length of one DWORD). All fields in a Management Capability Structure and any sub-structures have little endian bit ordering.

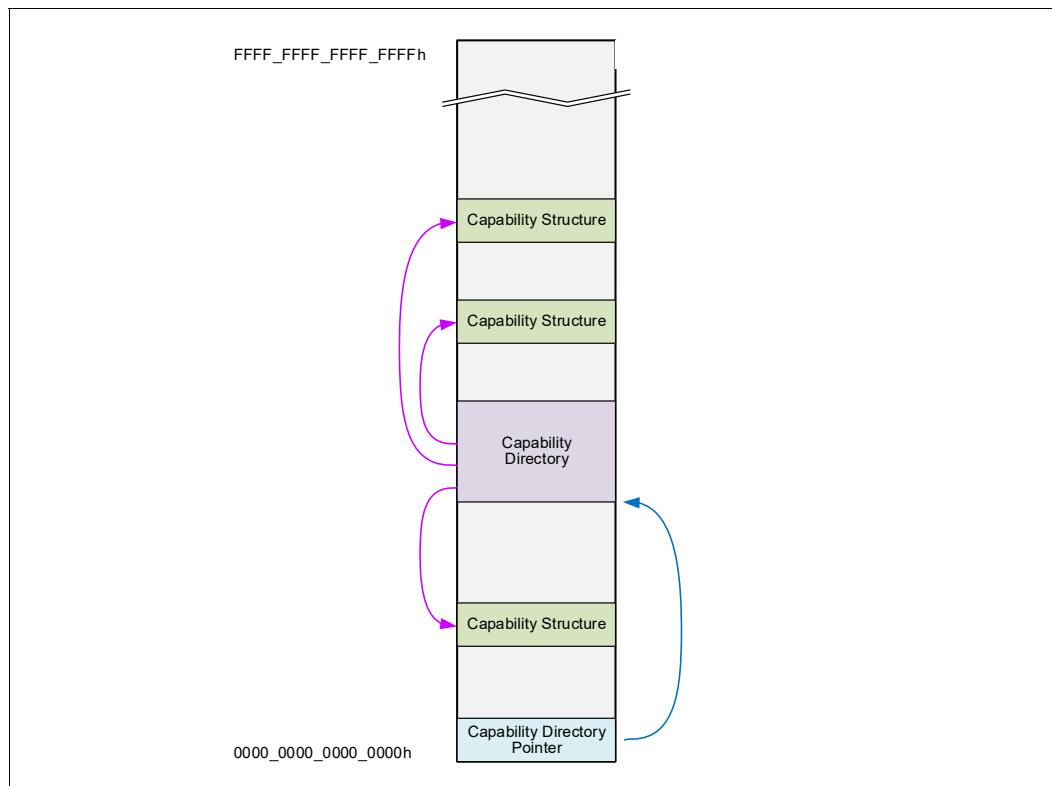
A Management Entity may support the UCIE Memory Access protocol (see [Section 8.1.4](#)) which exposes a 64-bit address space associated with the Management Entity containing fields that allow configuration by another Management Entity, such as the Management Director.

Each chiplet must support Management Element 0. Chiplet initialization and configuration are performed through Management Element 0 using the Chiplet Capability Structure and as a result Management Element 0 must support the UCIe Memory Access protocol. A chiplet may contain other Management Entities and the number of such Management Entities is implementation specific. These additional Management Entities may support the UCIe Memory Access protocol.

Figure 8-8 shows the UCIe Memory Access protocol memory map associated with a Management Entity that supports the UCIe Memory Access Protocol. The contents and organization of the memory map are implementation specific except for a 64-bit Capability Directory Pointer value located at address 0. If the Management Entity implements any Management Capability Structures, then the Management Capability Directory Pointer contains the address of a Management Capability Directory. If the Management Entity does not implement any Management Capability Structures, then the Management Capability Directory Pointer contains a value of 0.

The Management Capability Directory, described in Section 8.1.3.6.1, contains a list of pointers (i.e., 64-bit UCIe Memory Access protocol addresses) to Management Capability Structures supported by the Management Entity and contains a pointer (i.e., the Element ID) of the next Management Entity in the chiplet if one exists.

Figure 8-8. Memory Map for Management Entities



The organization that all Management Capability Structures follow is shown in [Figure 8-9](#). A Management Capability Structure is at least two DWORDS in size and may be larger. The size of a Management Capability Structure is Management Capability Structure specific. Associated with each Management Capability Structure is a Management Capability ID that identifies the capability. The list of Management Capability IDs defined by UCIE are listed in [Table 8-8](#).

Figure 8-9. Management Capability Structure Organization

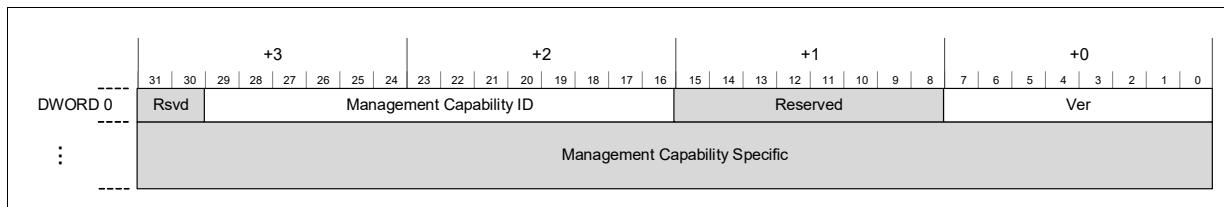
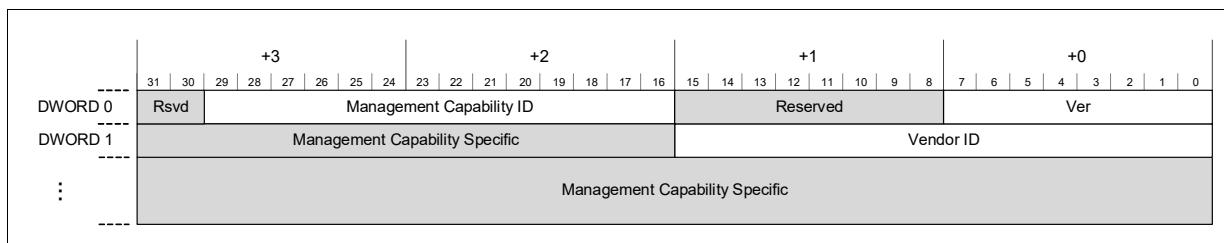


Table 8-8. UCIE-defined Management Capability IDs

Management Capability ID	Management Capability Structure Name	Description
0	Chiplet	See Section 8.1.3.6.2
1	Access Control	See Section 8.1.3.6.3
2	UCIE Memory Access Protocol	See Section 8.1.4.2
3	DFx Management Hub	See Section 8.3.1.1
4	Security Clearance Group	
5 to 12,287	Reserved	
12,288 to 16,383	Vendor defined	See Figure 8-10

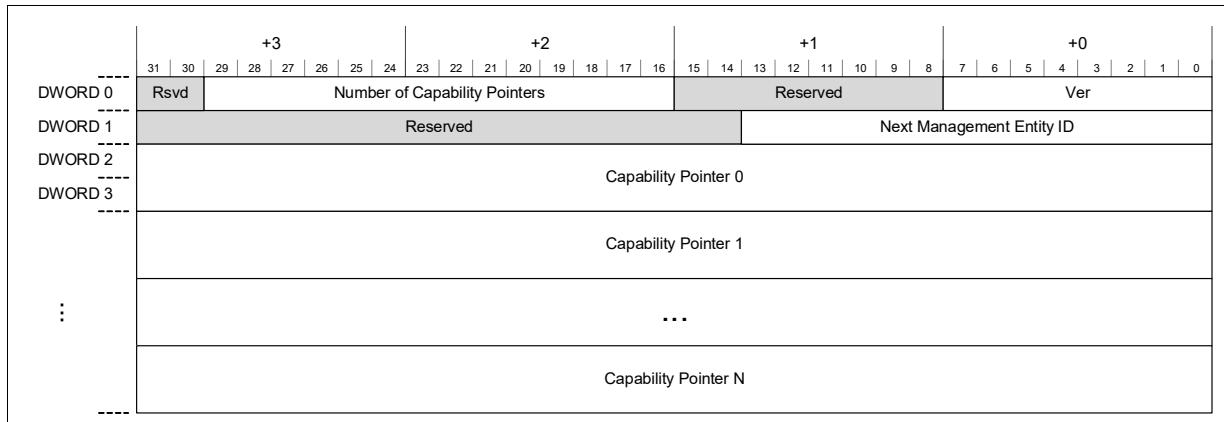
The top 4096 Management Capability IDs are available for vendor-defined use. The organization of a Vendor Defined Management Capability Structure is shown in [Figure 8-10](#). Bits 0 through 15 of WORD 1 contain the UCIE-assigned identifier of the vendor that specified the Management Capability Structure.

Figure 8-10. Vendor Defined Management Capability Structure Organization



8.1.3.6.1 Management Capability Directory

The Management Capability Directory provides a method for discovery of Management Capability Structures associated with a Management Entity. The structure of the Management Capability Directory is shown in [Figure 8-11](#) and described in [Table 8-9](#).

Figure 8-11. Management Capability Directory**Table 8-9. Management Capability Directory Fields**

Field Name	DWORD & Bit Location	Standard Security Asset Class ID ^a	Attribute	Description
Ver	0 [7:0]	17	RO	Capability Directory Version This field is a UCIE-defined version number that indicates the version of the capability directory. This field must be 00h in this version of the specification
Number of Capability Pointers	0 [29:16]	17	RO	Number of Capability Pointers This field indicates the number of Capability Pointers in the Capability Directory. Since the UCIE Memory Access Protocol must be supported in order to access the Management Capability Directory, this field cannot be zero.
Next Management Entity ID	1 [13:0]	17	RO	Next Management Entity ID Each chiplet contains a list of Management Entities that support the UCIE Memory Access Protocol starting with Management Element 0. This field contains the Management Entity ID of the next Management Entity in the chiplet that supports the UCIE Memory Access Protocol. If this is the last Management Entity in the chiplet that supports the UCIE Memory Access Protocol, then this field has a value of 0000h. Management Entity IDs in this list must be ordered from lowest to highest and may sparse (i.e., there may be gaps). The Management Entity list may be viewed as a list of Management Network IDs for Management Entities that support the UCIE Memory Access Protocol contained within the chiplet with the Chiplet ID field set to 0.

a. See Table 8-7 for a description of Standard Security Asset Class IDs.

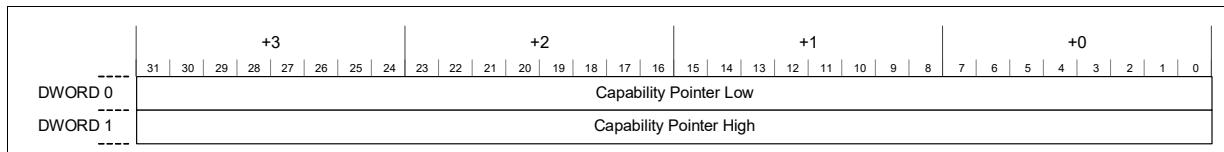
Figure 8-12. Capability Pointer

Table 8-10. Capability Pointer Fields

Field Name	DWORD & Bit Location	Standard Security Asset Class ID ^a	Attribute	Description
Capability Pointer Low	0 [31:0]	17	RO	<p>Capability Pointer Low</p> <p>Bits 0 to 31 of the 64-bit address of the first byte of the Capability Structure associated with the Capability pointed to by this Capability Pointer.</p> <p>A value of all 0s in both the Capability Pointer Low and High fields indicates that this is a Null Capability Pointer and should be skipped.</p> <p>Because capability structures must be DWORD-aligned, bits 0 and 1 must be 00b.</p>
Capability Pointer High	1 [31:0]	17	RO	<p>Capability Pointer High</p> <p>Bits 32 to 63 of the 64-bit address of the first byte of the Capability Structure associated with the Capability pointed to by this Capability Pointer.</p>

a. See [Table 8-7](#) for a description of Standard Security Asset Class IDs.

8.1.3.6.2 Chiplet Capability Structure

The Chiplet Capability Structure must be implemented in Management Element 0 of each chiplet and must not be implemented in any other Management Entity in a chiplet.

[Figure 8-13](#) shows the organization of the Chiplet Capability Structure. The Chiplet Capability Structure describes the basic characteristics of the chiplet. It points to a list of Management Port Structures that describe the characteristics of chiplet Management Ports. Each Management Port Structure contains one or more Route Entries that control the routing of UCIe Management Transport packets from the Management Fabric of the chiplet out the corresponding Management Port to an adjacent chiplet in the SiP.

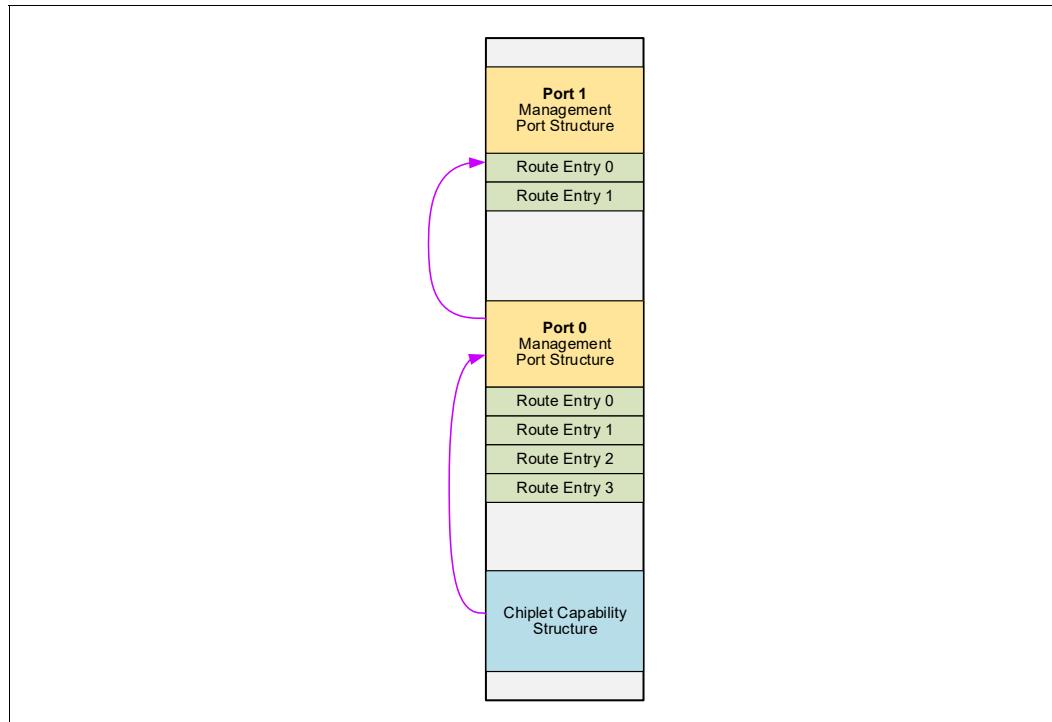
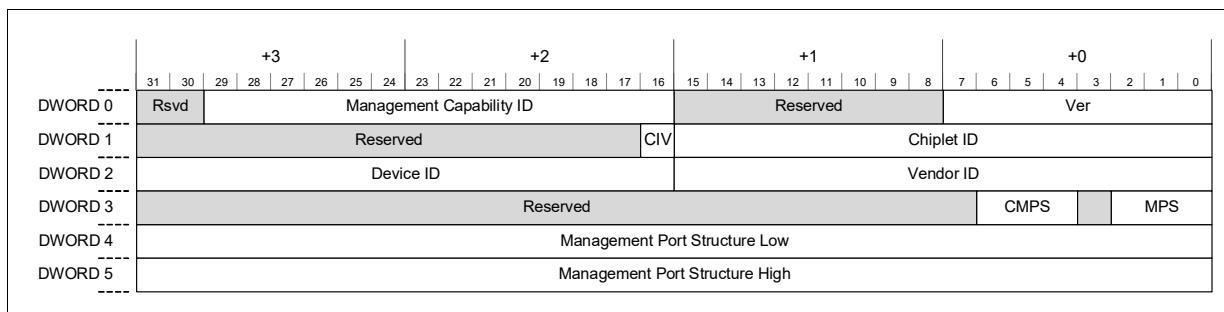
Figure 8-13. Chiplet Capability Structure Organization**Figure 8-14. Chiplet Capability Structure**

Table 8-11. Chiplet Capability Structure Fields (Sheet 1 of 2)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID ^a	Attribute	Description
Ver	0 [7:0]	17	RO	Capability Structure Version This field indicates the version of this capability structure. This field has a value of 00h in this specification.
Management Capability ID	0 [29:16]	17	RO	Management Capability ID This field specifies the Capability ID of this Management Capability Structure. The Chiplet Capability Structure has a Management Capability ID of 000h.
Chiplet ID	1 [15:0]	8	RW/RO	Chiplet ID This field is used to configure the Chiplet ID portion of the 16-bit Management Network ID associated with Management Element zero in the chiplet. A Management Network ID is partitioned into a Chiplet ID field in the upper bits and an Entity ID field in the lower bits (see Section 8.1.3.2). The lower bits of this field associated with the Entity ID portion of the Management Network ID are hardwired to 0 (i.e., RO). Since bits 0 and 1 are only associated with an Entity ID, they are always hardwired to zero. The upper bits of this field associated with the Chiplet ID portion of the Management Network ID may be read and written (i.e., RW). These upper bits must be initialized with the Chiplet ID value associated with the chiplet. The initial value of these upper bits is all ones (i.e., 1). The value of the Chiplet ID portion of the Management Network ID is used for PCIe Management Transport packet routing only when the Chiplet ID Valid (CIV) field is set to 1.
CIV	1 [16]	8	RW	Chiplet ID Valid When this bit is set to 1, the Chiplet ID value in the Chiplet ID field is used for PCIe Management Transport packet routing.
Vendor ID	2 [15:0]	17	RO	Vendor ID PCIe assigned identifier of the vendor that produced the chiplet.
Device ID	2 [31:16]	17	RO	Device ID Vendor assigned identifier that identifies the type of chiplet produced by that vendor. The tuple (Vendor ID, Device ID) uniquely identifies a type of chiplet.
MPS	3 [2:0]	17	RO	Maximum Packet Size This field indicates the maximum PCIe Management Transport packet size supported by the chiplet (see Section 8.1.3.1.2). 000b: 4 DWORDs 001b: 8 DWORDs 010b: 16 DWORDs 011b: 32 DWORDs 100b: 64 DWORDs 101b: 128 DWORDs 110b: 256 DWORDs 111b: 512 DWORDs

Table 8-11. Chiplet Capability Structure Fields (Sheet 2 of 2)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID ^a	Attribute	Description
CMPS	3 [6:4]	16	RW	<p>Configured Maximum Packet Size This field indicates the configured maximum UCIe Management Transport packet size of the chiplet (see Section 8.1.3.1.2). The Configured Packet Size must be less than or equal to the Maximum Packet Size. Setting the Configured Packet Size to a value greater than the Maximum Packet Size blocks all Management Entities in the chiplet from emitting packets. Management Entities in the chiplet never generate UCIe Management Transport packets that are larger than the Configured Packet Size. This field has no effect on how a Management Entity handles receipt of a packet or transfer of packets on the Management Fabric. These behaviors are only affected by the Maximum Packet Size. The initial value of this field is 001b. 000b: 4 DWORDs 001b: 8 DWORDs 010b: 16 DWORDs 011b: 32 DWORDs 100b: 64 DWORDs 101b: 128 DWORDs 110b: 256 DWORDs 111b: 512 DWORDs</p>
Management Port Structure Low	4 [31:0]	17	RO	<p>Management Port Structure Bits 0 to 31 of the 64-bit address of the first Management Port Structure. Because the Management Port Structure must be DWORD-aligned, bits 0 and 1 must be 00b and are ignored. If the chiplet implements zero Management Ports, then this field must be 0.</p>
Management Port Structure High	5 [31:0]	17	RO	<p>Management Port Structure Bits 32 to 63 of the 64-bit address of the first Management Port Structure. If the chiplet implements zero Management Ports, then this field must be 0.</p>

a. See [Table 8-7](#) for a description of Standard Security Asset Class IDs.

8.1.3.6.2.1 Management Port Structure

The Management Port Structure provides a mechanism to discover and configure the characteristics of a chiplet Management Port. The structure contains Route Entries associated with the port and points to the next Management Port if one exists.

Figure 8-15. Management Port Structure

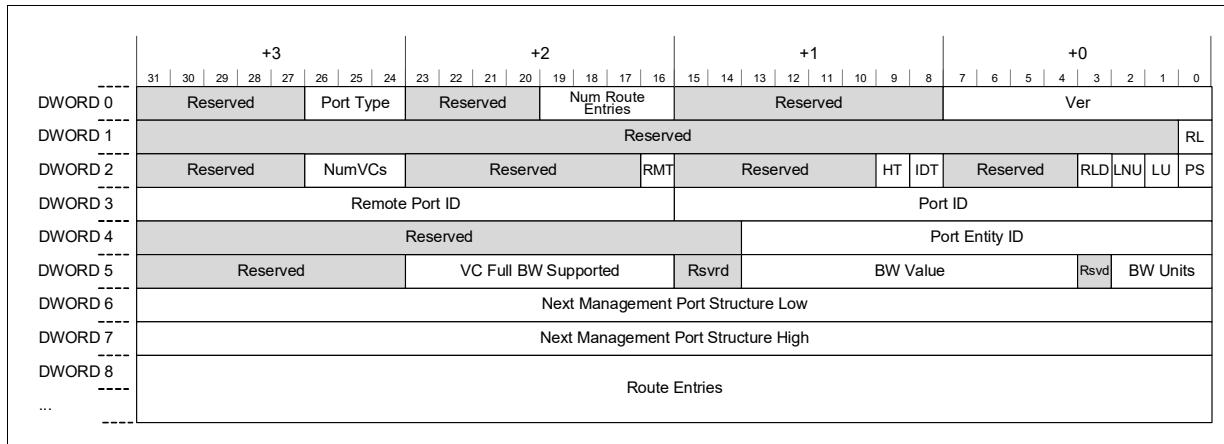


Table 8-12. Management Port Structure Fields (Sheet 1 of 4)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID ^a	Attribute	Description
Ver	0 [7:0]	17	RO	Management Port Structure Version This field indicates the version of the Management Port Structure. Must be 00h for this version of the specification.
Num Route Entries	0 [19:16]	17	RO	Number of Route Entries This field indicates the number of Route Entries associated with this Management Port. The number of Route Entries associated with the Management Port is equal to the value in this field plus 1. A Management Port must have at least one Route Entry associated with it. A value of 0h in this field indicates one Route Entry.
Port Type	0 [26:24]	17	RO	Management Port Type This field indicates the management port type. 000b: Not Implemented (skip) 001b: UCIE Sideband 010b: UCIE Mainband 111b: Vendor Defined Others: Reserved A value of 000b indicates that the management port is not implemented and should be skipped.

Table 8-12. Management Port Structure Fields (Sheet 2 of 4)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID ^a	Attribute	Description
RL	1 [0]	8	RW	<p>Retrain Link Writing a 1 to this bit initiates retraining of the link associated with the Management Port. Because the UCIE Management Transport may be multiplexed with other protocols on the link, retraining a port link may affect SiP operation. Retraining a UCIE Sideband link will also retrain the corresponding UCIE Mainband link if one exists. Retraining a link may take time to complete after this bit is written. Status in this structure does not reflect the result of a link retraining until the operation completes. The Retrain Link Done (RLD) field may be used to determine when the operation has completed.</p>
PS	2 [0]	17	RO	<p>Port Status This field indicates the current Management Port Status. 0: Link Not Up 1: Link Up</p>
LU	2 [1]	17	RW1C	<p>Link Up This bit is set to 1 when the link transitions from a link not up to a link up state. Writing to this field has no effect on the link.</p>
LNU	2 [2]	17	RW1C	<p>Link Not Up This bit is set to 1 when the link transitions from a link up to a link not up state. Writing to this field has no effect on the link.</p>
RLD	2 [3]	17	RW1C	<p>Retrain Link Done This bit is set to 1 when a 1 is written to the Retrain Link (RL) field and the corresponding retrain operation has completed.</p>
IDT	2[8]	17	RW1C	<p>Init Done Timeout This bit is set to 1 when an Init Done timeout is detected (see Section 8.2.4.4 for details).</p>
HT	2[9]	17	RW1C	<p>Heartbeat Timeout This bit is set to 1 when a Heartbeat Timeout is detected (see Section 8.2.5.1.3 for details). Heartbeat Timeout is implemented only on the UCIE sideband.</p>
RMT	2[16]	17	RW1C	<p>Remote Management Transport This bit is set to 1 when management transport support is advertised by the remote chiplet associated with this Management Port (see Section 4.5.3.3.1.1).</p>
Num VCs	2 [26:24]	17	RO	<p>Number of Virtual Channels If the Port Status field indicates that the link is up, then the value of this field indicates the number of virtual channels available on the Management Port minus 1 (i.e., a value of 0 means one VC, a value of 1 means two VCs, and so on). Because implemented virtual channels must always start at 0 and increase sequentially. A value of N in this field indicates that Virtual Channels 0 through N are available. If the Port Status field indicates that the link is not up, then this field has a value of 000b.</p>

Table 8-12. Management Port Structure Fields (Sheet 3 of 4)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID ^a	Attribute	Description
Port ID	3 [15:0]	17	RO	<p>Port Identifier This field indicates the chiplet's unique 16-bit identifier associated with the corresponding Management Port. Port identifiers are statically assigned by the chiplet manufacturer, never change, and need not be assigned sequentially (i.e., their assignment may be sparse) except as outlined below.</p> <p>PCIe mainband and sideband ports associated with the same physical connection share all port ID bits in common except bit 0. Bit 0 has a value of 0 in the mainband port identifier and a value of 1 in the corresponding sideband port identifier. For example, if a PCIe mainband port has a port identifier of N, then N is even and the PCIe sideband port associated with that mainband port is odd and has a port identifier of (N+1).</p> <p>The port identifier FFFFh is reserved.</p>
Remote Port ID	3 [31:16]	17	RO	<p>Remote Port Identifier This field indicates the remote chiplet unique 16-bit port identifier associated with the remote Management Port (i.e., the Port ID of the adjacent chiplet). The value of this field is only valid when the Port Status field indicates that the link is up; otherwise, the value of this field is FFFFh.</p> <p>The value of this field is obtained from the remote chiplet during management transport path negotiation (see Section 4.5.3.3.1.1).</p>
Port Entity ID	4 [13:0]	17	RO	<p>Port Entity ID This field indicates the Entity ID associated with the Management Port (see Section 8.1.3.2).</p>
BW Units	5 [2:0]	17	RO	<p>Port Bandwidth Units If the Port Status field indicates that the link is up, then this field indicates the units associated with the BW Value field.</p> <p>Support for port bandwidth reporting is optional.</p> <ul style="list-style-type: none"> 000b: Port bandwidth not reported 001b: KB/s 010b: MB/s 011b: GB/s 100b: TB/s Others: Reserved <p>If the port Status field indicates that the link is not up or port bandwidth reporting is not supported, then this field has a value of 000b.</p>
BW Value	5 [13:4]	17	RO	<p>Port Bandwidth Value If the Port Status field indicates that the link is up, then this field indicates the maximum port bandwidth value. The units associated with the value are specified by the BW Units field.</p> <p>If the port bandwidth may change (e.g., because a chiplet port supports multiple link speeds), then this field reflects the current port bandwidth.</p> <p>Support for port bandwidth reporting is optional.</p> <p>If the port Status field indicates that the link is not up or port bandwidth reporting is not supported, then this field has a value of 000h.</p>

Table 8-12. Management Port Structure Fields (Sheet 4 of 4)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID ^a	Attribute	Description
VC Full BW Supported	5 [23:16]	17	RO	<p>Virtual Channel Full Bandwidth Supported Each bit in this field corresponds to a virtual channel. If the Port Status field indicates that the link is up and NUM VCs field indicates that a virtual channel is available, then a value of 1 in the bit corresponding to the virtual channel indicates that the virtual channel supports full link bandwidth reported by the BW Value field from the adjacent chiplet to this chiplet. A value of 0 in the bit corresponding to the virtual channel indicates that the virtual channel does not support full link bandwidth from the adjacent chiplet to this chiplet. Whether full link bandwidth is supported from this chiplet to the adjacent chiplet may be determined from the Management Port Structure in the adjacent chiplet.</p> <p>If the Port Status field indicates that the link is not up, then none of the virtual channels associated with the port are available.</p> <p>If a virtual channel is not available, then the value of the bit corresponding to the virtual channel has a value of 0.</p>
Next Management Port Structure Low	6 [31:0]	17	RO	<p>Next Management Port Structure Low Bits 0 to 31 of the 64-bit address of the first byte of the next Management Port Structure. Because Management Port Structures must be DWORD-aligned, bits 0 and 1 must be 00b.</p> <p>A value of all 0s in both the Management Port Structure Low and High fields indicates that there are no more Management Port Structures.</p>
Next Management Port Structure High	7 [31:0]	17	RO	<p>Next Management Port Structure High Bits 32 to 63 of the 64-bit address of the first byte of the Management Port Structure. A value of all 0s in both the Management Port Structure Low and High fields indicates that there are no more Management Port Structures.</p>

a. See [Table 8-7](#) for a description of Standard Security Asset Class IDs.

8.1.3.6.2.2 Route Entry

A Route Entry is used to specify a route from the Management Fabric within a chiplet out the Management Port associated with the Route Entry.

The TC Select field selects traffic classes that are filtered out from matching a Route Entry.

A Route Entry may specify a normal route or a default route. The type of route is determined by the RT field.

While the Chiplet ID and Entity ID size of chiplets may vary in an SiP, all Route Entry matching associated with a chiplet is performed using the Chiplet ID size of that chiplet.

Packet Route Entry matching is performed as follows.

- If a Route Entry has the Route Type field set to Normal Route, then a packet matches the Route Entry when all the following are true:
 - The link is up,
 - The packet is associated with a traffic class that has the corresponding bit of the TC Select field in the Route Entry set to 1,
 - The Chiplet ID portion (using the Chiplet ID width for this chiplet) of the packet's Destination ID field is greater than or equal to the value in the Base ID field, and

- The Chiplet ID portion (using the Chiplet ID width for this chiplet) of the packet's Destination ID field is less than or equal to the value in the Limit ID field.
- If a Route Entry has the Route Type field set to Default Route, then a packet matches the route when all the following are true:
 - The link is up,
 - The packet is associated with a traffic class that has the corresponding bit of the TC Select field in the Route Entry set to 1, and
 - The packet does not match any other Route Entry within the chiplet.

If a packet on the Management Fabric of a chiplet matches the route specified by the Route Entry, then the packet is transmitted out the Management Port associated with the Route Entry. The virtual channel used by the packet is specified by the VC ID field in the matching Route Entry.

Route Entries associated with the Management Ports on either side of a point-to-point link that interconnects two chiplets may be configured differently. This means that the TC-to-VC mapping in each direction on the link may be different.

Figure 8-16. Route Entry

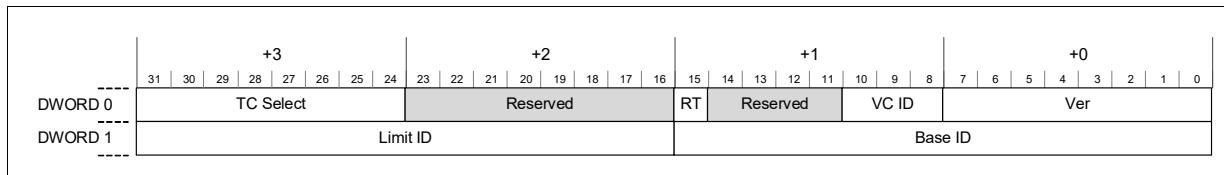


Table 8-13. Route Entry Fields (Sheet 1 of 2)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID ^a	Attribute	Description
Ver	0 [7:0]	8	RO	Route Entry Version This field indicates the version of the Route Entry. Must be 0h in this version of the specification
VC ID	0 [10:8]	8	RW	Virtual Channel ID This field selects the Virtual Channel (VC) used by packets that match this Route Entry. The default value of this field is 0 which maps all selected traffic classes onto VCO.
RT	0 [15]	8	RW	Route Type This field selects the routing type of this Route Entry. 0: Normal Route 1: Default Route The default value of this field is 0.

Table 8-13. Route Entry Fields (Sheet 2 of 2)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID ^a	Attribute	Description
TC Select	0 [31:24]	8	RW	<p>Traffic Class Select This field selects which traffic classes may match this Route Entry. A packet's traffic class is specified Traffic Classes (TC) in the packet header. Each bit in this field corresponds to a traffic class (i.e., bit 0 corresponds to TC0, bit 1 to TC1, and so on). If a bit in this field is set to 0, then packets with the associated traffic class are filtered out from matching the route specified by the Route Entry. If a bit in this field is set to 1, then packets with the associated traffic class are considered for matching the route specified by the Route Entry. The default value of this field is 0 which filters out all traffic classes.</p>
Base ID	1 [15:0]	8	RW/RO	<p>Base ID This field contains the Base ID value of the Chiplet ID associated with this Route Entry. This field contains a 16-bit Management Network ID. The Management Network ID is partitioned into a Chiplet ID field in the upper bits and an Entity ID field in the lower bits (see Section 8.1.3.2). The lower bits of this field associated with the Entity ID portion of the Management Network ID are hardwired to 0 (i.e., RO). Since bits 0 and 1 are only associated with an Entity ID, they are always hardwired to zero. The upper bits of this field associated with the Chiplet ID portion of the Management Network ID may be read and written (i.e., RW). These upper bits must be initialized with the Chiplet ID value associated with the Base ID. The initial value of these upper bits is all ones (i.e., 1).</p>
Limit ID	1 [31:16]	8	RW	<p>Limit ID This field contains the Limit ID value of the Chiplet ID associated with this Route Entry. This field contains a 16-bit Management Network ID. The Management Network ID is partitioned into a Chiplet ID field in the upper bits and an Entity ID field in the lower bits (see Section 8.1.3.2). The lower bits of this field associated with the Entity ID portion of the Management Network ID are hardwired to 0 (i.e., RO). The upper bits of this field associated with the Chiplet ID portion of the Management Network ID may be read and written (i.e., RW). These upper bits must be initialized with the Chiplet ID value associated with the Base ID. The initial value of these upper bits is all 0s. If the Base ID is greater than the Limit ID, then the Route Entry is disabled.</p>

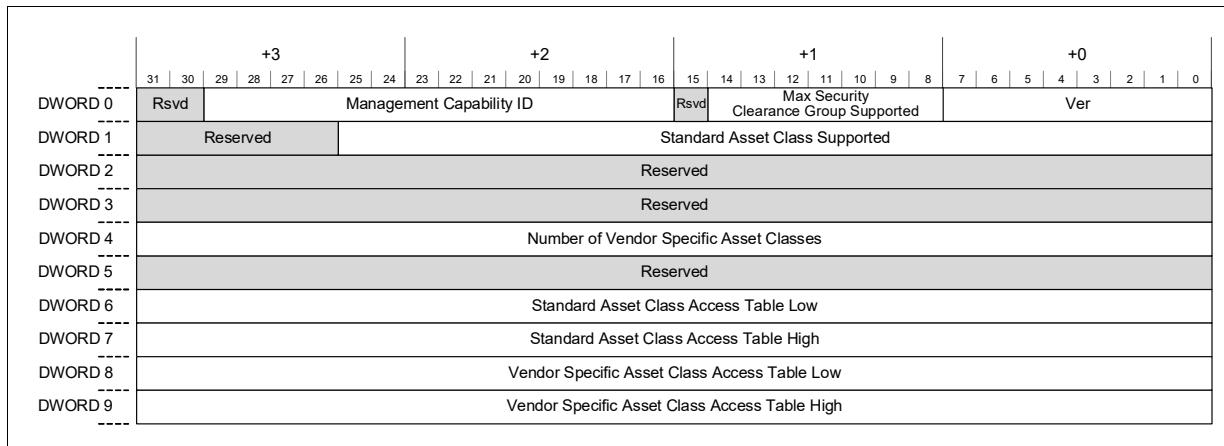
a. See [Table 8-7](#) for a description of Standard Security Asset Class IDs.

8.1.3.6.3 Access Control Capability Structure

A Management Entity must support the access control mechanism outlined in [Section 8.1.3.5](#) and must implement the Access Control Capability Structure described in this section. The Access Control Capability Structure provides access to the Read Access Control (RAC) and Write Access Control (WAC) structures associated with asset classes contained in the Management Entity.

The organization of the Access Control Capability Structure is shown in [Figure 8-17](#). It consists of a 10-DWORD header that contains a pointer to the standard asset class access table and the vendor defined asset class access table.

The standard and vendor defined asset access class tables consists of a sequence of 128-bit (4-DWORD) RAC and 128-bit (4-DWORD) WAC structure pairs. The number of RAC/WAC structure pairs in the standard asset class access table is equal to 26 (i.e., the number of standard asset class IDs) and the number of RAC/WAC structure pairs in the vendor defined asset class access table corresponds to the number of vendor defined asset classes. The fields in the 4-DWORD RAC and WAC structures are described in [Table 8-15](#) and [Table 8-16](#), respectively.

Figure 8-17. Access Control Capability Structure**Table 8-14. Access Control Capability Structure Fields (Sheet 1 of 2)**

Field Name	DWORD & Bit Location	Standard Security Asset Class ID ^a	Attribute	Description
Ver	0 [7:0]	17	RO	Capability Structure Version This field indicates the version of this capability structure. This field has a value of 00h in this specification.
Max Security Clearance Group Supported	0 [14:8]	17	RO	Max Security Clearance Group Supported This field specifies the maximum security clearance group value supported. A value of N in this field indicates that security clearance group values 0 through N are supported. If this capability structure is implemented, then security clearance group 0 must be supported. If the number of security clearance groups is not restricted, then N is equal to 127.
Management Capability ID	0 [29:16]	17	RO	Management Capability ID This field specifies the Capability ID of this Management Capability Structure. The Access Control Capability Structure has a Management Capability ID of 001h.
Standard Asset Class Supported	1 [25:0]	17	RO	Standard Asset Class Supported Each bit in this field represents an asset class ID (see Table 8-5 and Table 8-7). If a bit in this field is set to 1, then the asset class corresponding to the asset class ID represented by the bit is supported. If a bit in this field is set to 0, then the asset class corresponding to the asset class ID represented by the bit is not supported.

Table 8-14. Access Control Capability Structure Fields (Sheet 2 of 2)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID ^a	Attribute	Description
Number of Vendor Defined Asset Classes	4 [31:0]	17	RO	Number of Vendor Defined Asset Classes This field indicates the number of vendor defined asset classes. A value of all 0s in this field indicates that no vendor defined asset classes are supported.
Standard Asset Class Access Table Low	6 [31:0]	17	RO	Standard Asset Class Access Table Low Bits 0 to 31 of the 64-bit address of the base address of the Standard Asset Class Table. Because the Standard Asset Class Access Table must be DWORD-aligned, bits 0 and 1 must be 00b.
Standard Asset Class Access Table High	7 [31:0]	17	RO	Standard Asset Class Access Table High Bits 32 to 63 of the 64-bit address of the base address of the Standard Asset Class Table.
Vendor Defined Asset Class Access Table Low	8 [31:0]	17	RO	Vendor Defined Asset Class Access Table Low Bits 0 to 31 of the 64-bit address of the base address of the Vendor Defined Asset Class Table. Because the Vendor Defined Asset Class Access Table must be DWORD-aligned, bits 0 and 1 must be 00b. A value of zero in the Vendor Defined Asset Class Access Table Low and High fields indicates that there is no Vendor Defined Asset Class Access Table.
Vendor Defined Asset Class Access Table High	9 [31:0]	17	RO	Vendor Defined Asset Class Access Table High Bits 32 to 63 of the 64-bit address of the base address of the Vendor Defined Asset Class Table. A value of zero in the Vendor Defined Asset Class Access Table Low and High fields indicates that there is no Vendor Defined Asset Class Access Table.

a. See [Table 8-7](#) for a description of Standard Security Asset Class IDs.

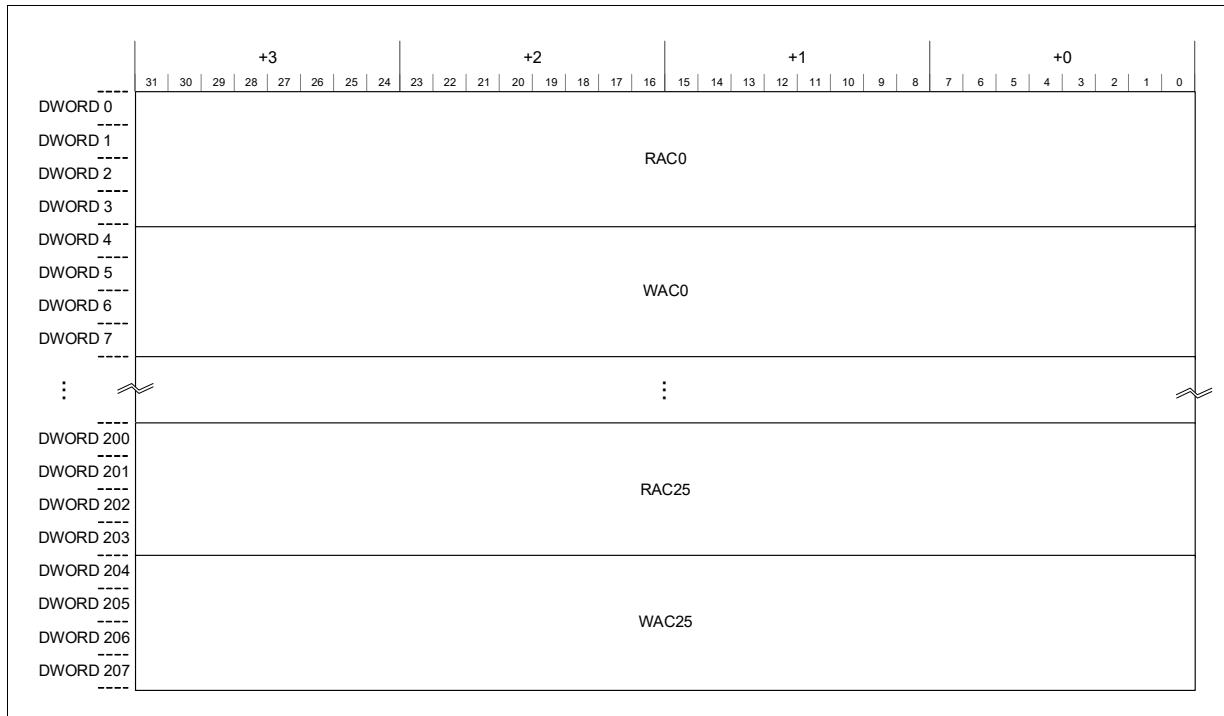
Figure 8-18. Standard Asset Class Access Table

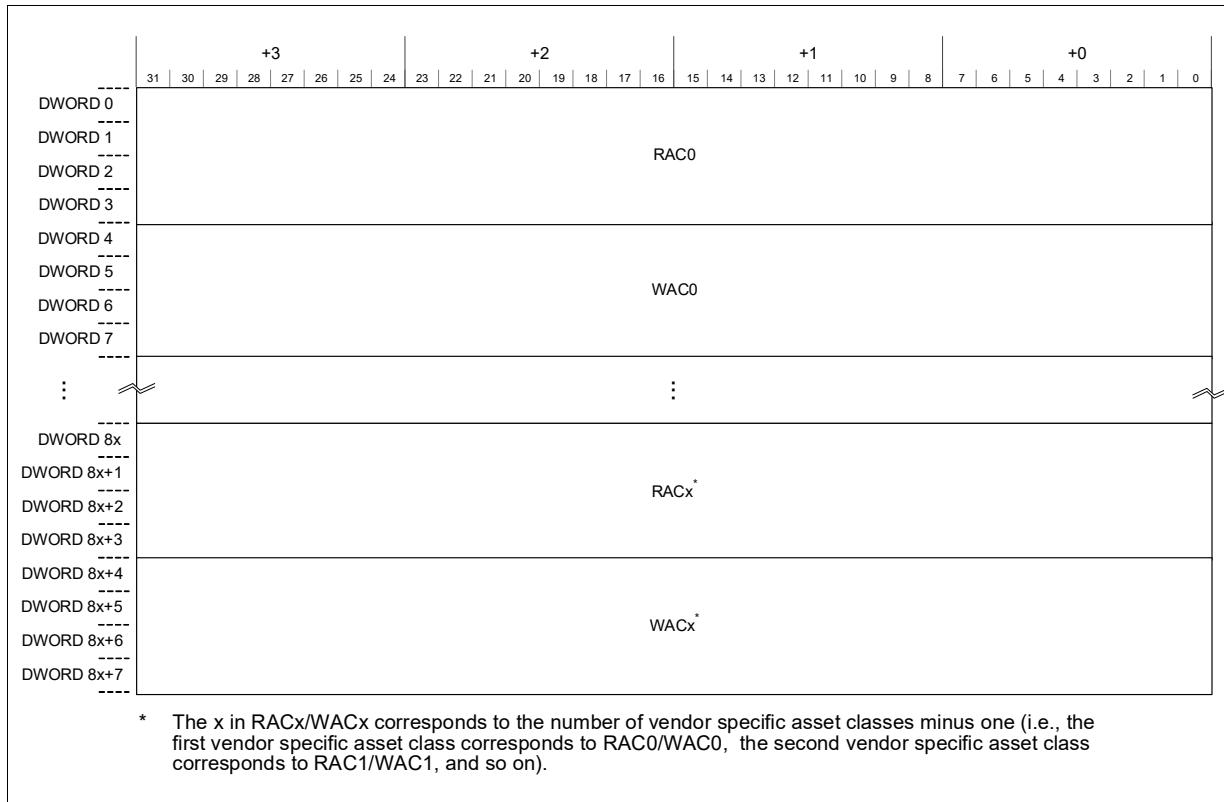
Figure 8-19. Vendor Defined Asset Class Access Table

Table 8-15. Read Access Control (RAC) Structure Field Description

Field Name	DWORD ^a & Bit Location	Standard Security Asset Class ID ^b	Attribute	Description
RACx_SD	8x [0]	0	RW/RO	<p>Read Access Control Security Director This bit provides access control for the Security Director. If this bit is 1, then Security Director read accesses to assets in the Management Entity of the type associated with this RAC structure are allowed. If this bit is 0, then Security Director read accesses to assets in the Management Entity of the type associated with this RAC structure are not allowed. The initial value of this bit is 1. If the Management Entity contains no assets associated with the access class corresponding to this RAC structure, then this field is hardwired to 0.</p>
RACx_LL	8x [31:1]	0	RW/RO	<p>Read Access Control Lower Lower This field provides access control for Security Clearance Groups 1 through 31. Bit x corresponds to Security Clearance Group x. If a bit is 1, then read accesses with the corresponding clearance group to assets in the Management Entity of the type associated with this RAC structure are allowed. If this bit is 0, then read accesses with the corresponding clearance group to assets in the Management Entity of the type associated with this RAC structure are not allowed. The initial value of each bit in this field is 0. If the Management Entity contains no assets associated with the access class corresponding to this RAC structure, then this field is hardwired to 0. If the Management Entity does not support the security clearance group associated with a bit in this field, then the bit is hardwired to 0.</p>
RACx_LM	8x+1 [31:0]	0	RW/RO	<p>Read Access Control Lower Middle This field provides access control for Security Clearance Groups 32 through 63. Bit x corresponds to Security Clearance Group (x + 32). See RACx_LL for a description of this field. The initial value of each bit in this field is 0.</p>
RACx_UM	8x+2 [31:0]	0	RW/RO	<p>Read Access Control Upper Middle This field provides access control for Security Clearance Groups 64 through 95. Bit x corresponds to Security Clearance Group (x + 64). See RACx_LL for a description of this field. The initial value of each bit in this field is 0.</p>
RACx_UU	8x+3 [31:0]	0	RW/RO	<p>Read Access Control Upper Upper This field provides access control for Security Clearance Groups 96 through 127. Bit x corresponds to Security Clearance Group (x + 96). See RACx_LL for a description of this field. The initial value of each bit in this field is 0.</p>

a. DWORD in this table refers to the DWORD offset into asset class access table for RACx. For example, the 128-bit RAC2 structure is at DWORD offsets 16, 17, 18, and 19.

b. See [Table 8-7](#) for a description of Standard Security Asset Class IDs.

Table 8-16. Write Access Control (WAC) Structure Field Description

Field Name	DWORD ^a & Bit Location	Standard Security Asset Class ID ^b	Attribute	Description
WACx_SD	8x+4 [0]	0	RW	<p>Write Access Control Security Director This bit provides access control for the Security Director. If this bit is 1, then Security Director write accesses to assets in the Management Entity of the type associated with this WAC structure are allowed. If this bit is 0, then Security Director write accesses to assets in the Management Entity of the type associated with this WAC structure are not allowed. The initial value of this bit is 1. If the Management Entity contains no assets associated with the access class corresponding to this WAC structure, then this field is hardwired to 0.</p>
WACx_LL	8x+4 [31:1]	0	RW	<p>Write Access Control Lower Lower This field provides access control for Security Clearance Groups 1 through 31. Bit x corresponds to Security Clearance Group x. If a bit is 1, then write accesses with the corresponding clearance group to assets in the Management Entity of the type associated with this WAC structure are allowed. If this bit is 0, then write accesses with the corresponding clearance group to assets in the Management Entity of the type associated with this WAC structure are not allowed. The initial value of each bit in this field is 0. If the Management Entity contains no assets associated with the access class corresponding to this WAC structure, then this field is hardwired to 0. If the Management Entity does not support the security clearance group associated with a bit in this field, then the bit is hardwired to 0.</p>
WACx_LM	8x+5 [31:0]	0	RW	<p>Write Access Control Lower Middle This field provides access control for Security Clearance Groups 32 through 63. Bit x corresponds to Security Clearance Group (x + 32). See WACx_LL for a description of this field. The initial value of each bit in this field is 0.</p>
WACx_UM	8x+6 [31:0]	0	RW	<p>Write Access Control Upper Middle This field provides access control for Security Clearance Groups 64 through 95. Bit x corresponds to Security Clearance Group (x + 64). See WACx_LL for a description of this field. The initial value of each bit in this field is 0.</p>
WACx_UU	8x+7 [31:0]	0	RW	<p>Write Access Control Upper Upper This field provides access control for Security Clearance Groups 96 through 127. Bit x corresponds to Security Clearance Group (x + 96). See WACx_LL for a description of this field. The initial value of each bit in this field is 0.</p>

a. DWORD in this table refers to the DWORD offset into asset class access table for WACx. For example, the 128-bit WAC2 structure is at DWORD offsets 20, 21, 22, and 23.

b. See [Table 8-7](#) for a description of Standard Security Asset Class IDs.

8.1.3.6.4 Security Clearance Group Capability Structure

The Security Clearance Group Capability Structure allows a Security Director to configure the Security Clearance Group value used by a Management Entity when issuing Management Transport requests. This capability structure must be implemented by a Management Entity that is the ultimate source of UCIE Management Transport request packets (i.e., emits request packets) and is not required to be implemented by any other Management Entity.

In some cases, a Management Entity may need to issue requests with different Security Clearance Group values when operating in different contexts. The Security Clearance Group Capability Structure supports multiple Security Clearance Group Contexts to allow a Security Director to configure a Security Clearance Group value for each context. How a Security Director determines the manageability functions provided by these contexts and what Security Clearance Group value should be used is beyond the scope of this specification.

Figure 8-20. Security Clearance Group Capability Structure

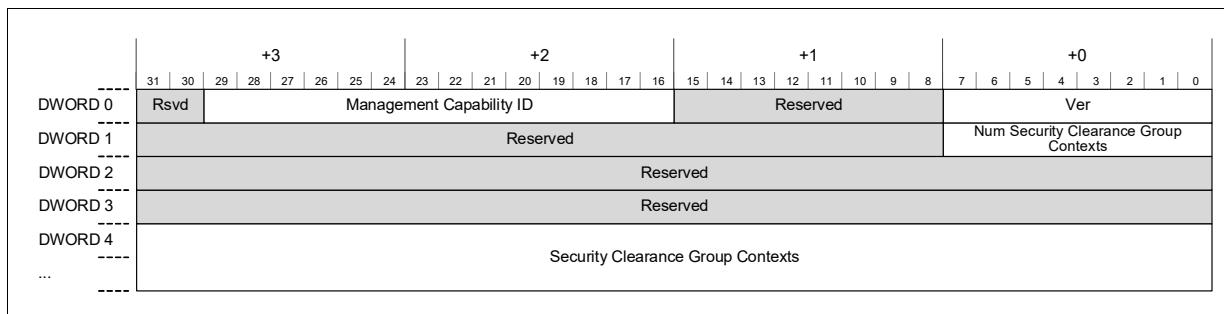


Table 8-17. Security Clearance Group Capability Structure Fields

Field Name	DWORD & Bit Location	Standard Security Asset Class ID ^a	Attribute	Description
Ver	0 [7:0]	17	RO	Capability Structure Version This field indicates the version of this capability structure. This field has a value of 00h in this specification.
Management Capability ID	0 [29:16]	17	RO	Management Capability ID This field specifies the Capability ID of this Management Capability Structure. The Security Clearance Group Capability Structure has a Management Capability ID of 004h.
Num Security Clearance Group Contexts	1 [7:0]	17	RO	Number of Security Clearance Group Contexts This field indicates the number of Security Clearance Group Contexts associated with this Management Entity. The number of Security Clearance Groups Contexts associated with this Management Entity is equal to the value in this field plus 1. A Management Entity that implements this capability structure must have at least one Security Clearance Group Context.

a. See Table 8-7 for a description of Standard Security Asset Class IDs.

8.1.3.6.4.1 Security Clearance Group Context

Figure 8-21. Security Clearance Group Context

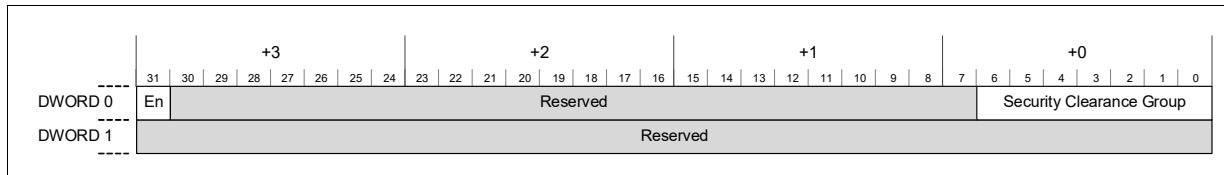


Table 8-18. Security Clearance Group Context Fields

Field Name	DWORD& Bit Location	Standard Security Asset Class ID ^a	Attribute	Description
Security Clearance Group	0 [6:0]	0	RW	Security Clearance Group This field is configured by the Security Director with the Security Clearance Group value used by the Management Entity when issuing a request. The initial value of this field is 7Fh if this context is not associated with a Security Director. The initial value of this field is 00h if this context is associated with a Security Director.
En	0 [31]	0	RW	Request Enable When this field is set to 1 the Management Entity may issue requests associated with this security clearance group context. When this field is set to 0 the Management Entity must not issue requests associated with this security clearance group context. The initial value of this field is 0 if this context is not associated with a Security Director. The initial value of this field is 1 if this context is associated with a Security Director.

a. See [Table 8-7](#) for a description of Standard Security Asset Class IDs.

8.1.4 UCIE Memory Access Protocol

The UCIE Memory Access Protocol provides read and write access to memory mapped structures and memory associated with a Management Entity that supports the UCIE Memory Access Protocol. A Management Entity exposes a 64-bit address space. The relationship of this address space to a system or I/O address map is beyond the scope of this specification.

The address space associated with a Management Entity may be local to that Management Entity or shared across one or more Management Entities in a chiplet. For example, the same address in two Management Entities may reference the same memory location or different memory locations (e.g., a memory location associated with each Management Entity). A Management Entity may have some addresses that are local and some that are shared. For shared addresses, how concurrent accesses, security, and mutual exclusion are handled is beyond the scope of this specification.

The UCIE Memory Access Protocol utilizes the UCIE Management Transport access control mechanism (see [Section 8.1.3.5](#)).

8.1.4.1 UCIe Memory Access Protocol Packets

This section describes UCIe Memory Access Protocol packets. These packets are carried by the UCIe management transport.

8.1.4.1.1 UCIe Memory Request Packet

Memory request packets are issued by a Management Entity to read or write memory mapped structures or memory in another Management entity. The Opcode field indicates the type of operation. When a UCIe Management Transport packet carries a UCIe Memory Request, the Resp field is set to 0 corresponding to a request packet.

UCIe Memory Request packet operations are non-posted. If a UCIe Management Transport packet that carries a UCIe Memory Request packet is not discarded, then a UCIe Memory Response packet is sent in response.

A Management Entity may issue requests on an ordered or unordered traffic class when the Unordered Traffic Class Enable (UE) bit is set to 1 in the UCIe Memory Access Protocol capability structure. When the UE bit is cleared to 0, then the Management Entity may only issue requests on an ordered traffic class and must not issue requests on an unordered traffic class. Whether a Management Entity utilizes an unordered traffic class is implementation specific.

The Tag field in a UCIe Memory Request packet is an 8-bit field populated by the requester, carried in a request packet, and returned by the responder in the corresponding response packet if one is generated. A requester may have multiple outstanding requests with the same Tag field value to the same or different responders. The responder must not assume that Tag field values are unique and must not in any way interpret the Tag field value. The use of the Tag field is requester implementation specific and may be used for applications such as mapping responses to previously issued requests; determining the responder associated with a response packet; and detecting lost, dropped, or discarded packets.

The maximum number of requests that a requester may have outstanding is requester implementation specific.

Figure 8-22 shows the fields of a UCIe Memory Access Request packet. Reserved fields (i.e., ones labeled as Rsvd) must be filled with all 0s when the packet is formed. Reserved fields must be forwarded unmodified on the Management Network and ignored by receivers. An implementation that relies on the value of a reserved field in a packet is non-compliant.

Figure 8-22. UCIe Memory Access Request Packet Format

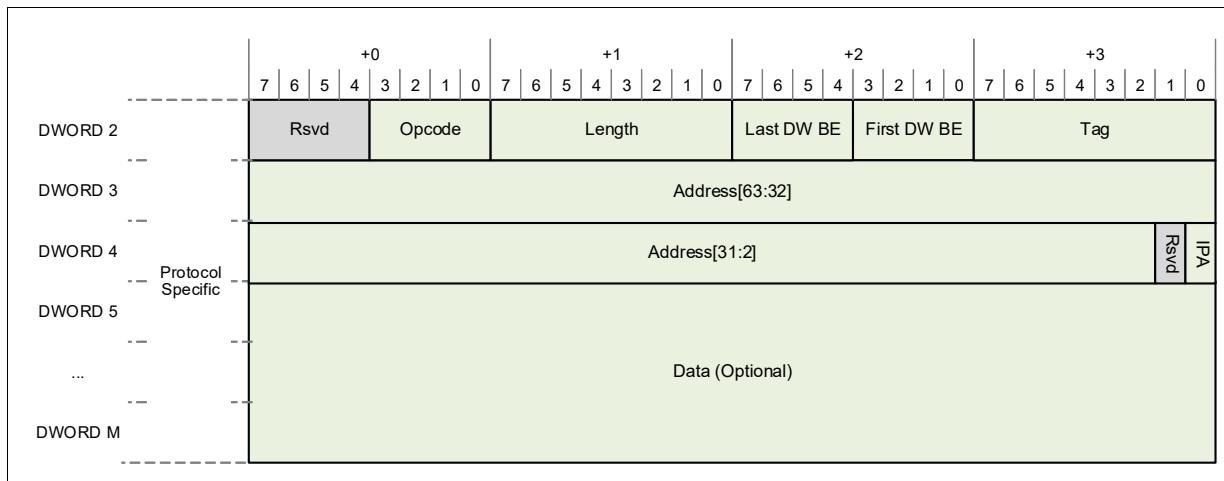


Table 8-19 defines the fields of a UCIe Memory Access Request packet. The packet starts at DWORD 2 because DWORDs 0 and 1 contain the UCIe Management Transport packet header. All fields in the table have little endian bit ordering, similar to [Figure 8-5](#) (e.g., Address bits 32 through 39 are in Byte 3 bits 7 through 0 of DWORD 3, and Address bits 40 through 47 are in Byte 2 bits 7 through 0 of DWORD 2 and so on).

Table 8-19. UCIe Memory Access Request Packet Fields

Field Name	Field Size	Description
Tag	8 bits	Tag This field contains the value of the tag field of the corresponding memory access request.
First DW BE	4 bits	First Data DWORD Byte Enable This field contains byte enables for the first (or only) DWORD referenced. A value of 0 indicates that the corresponding byte is not enabled and a value of 1 indicates that the corresponding byte is enabled. Bit 0 corresponds to Byte 0. Bit 1 corresponds to Byte 1. Bit 2 corresponds to Byte 2. Bit 3 corresponds to Byte 3.
Last DW BE	4 bits	Last Data DWORD Byte Enable This field contains byte enables for the last DWORD referenced. If the LENGTH field has a value of 0, then this field must be 0000b. A value of 0 indicates that the corresponding byte is not enabled and a value of 1 indicates that the corresponding byte is enabled. Bit 0 corresponds to Byte 0. Bit 1 corresponds to Byte 1. Bit 2 corresponds to Byte 2. Bit 3 corresponds to Byte 3.
Length	8 bits	Data Length This field indicates the length of data referenced in DWORDs. The length of the packet in DWORDs is equal to the value of this field plus 1 (e.g., a value of 00h in this field indicates a packet length of one DWORD, a value of 01h in this field indicates a packet length of two DWORDs, and so on).
Opcode	4 bits	Opcode This field indicates the memory access request operation. 0000b: Reserved (used for responses) 0001b: Memory Read (MemRd) 0010b: Memory Write (MemWr) Others: Reserved
Address	62 bits	Address This field contains the DWORD address being referenced in the Management Entity.
IPA	1 bit	Ignore Prohibited Access This bit indicates whether accesses to prohibited assets should be ignored. When access to a prohibited asset is ignored, the asset is not accessed but the request completes successfully. The value of this bit is determined by the requester and should not be set to 1 during normal operation because doing so would cause access violations to not be reported in the Response Status. 0: Do not ignore access to prohibited assets 1: Ignore accesses to prohibited assets
Data	Varies	Data This field is present in Memory Write requests and contains the data being written. This field is not present in Memory Read requests.

8.1.4.1.2 UCIe Memory Access Response Packet

A UCIe Memory Access Response packet is generated by a Management Entity when the processing associated with a UCIe Memory Access Request packet completes. When a UCIe Management Transport packet carries a UCIe Memory Response, the Resp field is set to 1 corresponding to a response packet.

A UCIe Memory Access Protocol responder must always support UCIe Memory Request packets on all traffic classes (TC). The traffic class of a UCIe Memory Access Response packet is the same as the traffic class used in the corresponding UCIe Memory Access Request packet.

As described in [Section 8.1.3.1.1](#), each traffic class is a unique ordering domain. There are no ordering guarantees for UCIe Memory Request packets in different traffic classes.

Within an ordered traffic class, UCIe Memory Request packets are delivered in-order between a requester and a responder and UCIe Memory Response packets are delivered in-order between a responder and the requester. There are no ordering guarantees between requests to different responders and there are no ordering guarantees between responses from different responders to a requester. Within an unordered traffic class there are no packet ordering guarantees and the packets may be delivered in any order.

A Management Entity may process received UCIe Memory Request packets sequentially (i.e., one at a time) or concurrently (i.e., two or more at a time). There are no ordering requirements between requests in different traffic classes; however, the result of processing these requests must be equivalent to some sequential processing of requests performed in an atomic manner.

Regardless of whether UCIe Memory Request packets are associated with an ordered or an unordered traffic class, a responder may send UCIe Memory Access Response Packets out-of-order (i.e., a responder is not required to send response packets in the same order that the corresponding request packets were received by the responder). This means that responses may be received by a requester in an order different from the order in which the requests were sent by the requester.

IMPLEMENTATION NOTE — RESPONSES IN AN ORDERED TRAFFIC CLASS

Because responders are free to send response packets for the UCIe Memory Access protocol in any order, an implementation may reorder these responses when carried in an ordered traffic class. While this conflicts with the ordered traffic class requirements, a requester cannot tell whether this reordering occurred at the responder or in the ordered traffic class of the Management Network.

The Status field in a UCIe Memory Access Response packet indicates the status associated with processing the corresponding UCIe Memory Access Request packet. If a UCIe Memory Access Request packet is processed successfully, then a UCIe Memory Access Response packet is generated with status Success. If the request requires response data, then all the data associated with the successful response is contained in a single response packet.

If a Management Entity receives a well formed UCIe Management Transport packet, but the UCIe Memory Access Request packet is malformed, then no processing of the request occurs and a response with no data and status Packet Error is returned.

- Examples of a malformed UCIe Memory Access Request packet:
 - Receipt of a UCIe Memory Access Request packet with a reserved value in the Opcode field.
 - Receipt of a UCIe Memory Access Request packet with the Length field set to zero and the Last DW BE field set to a nonzero value.

If a request violates the programming model of a Management Entity, then the request is not performed and a response with no data and status Programming Model Violation is returned.

- Examples of programming model violations:
 - Unless otherwise specified all UCIe defined structures must be accessed as DWORDs.

If a Management Entity receives a request, is not capable of processing the request, but will be able to process the request at some point in the future, then a response with no data and status Retry Request is returned. The Retry Request status should not be used during normal operation and implementations are strongly encouraged to only use the Retry Status when absolutely necessary. How long a requester waits after receiving a response with status Retry Request before reissuing the request is implementation specific. The Max Retry Time Units and Max Retry Time Value fields in the UCIe Memory Access Protocol Capability Structure report the maximum duration of time during which a Management Entity may return a response with status Retry Request. A requester may use this time duration to determine how long to poll a responder before declaring that the responder has malfunctioned.

If the Management Entity can process a request, the request does not contain an error, and the request attempts to access an asset that is prohibited, then the asset is not accessed, and no processing associated with the request occurs.

- If the Ignore Prohibited Access (IPA) bit in the request is cleared to 0, then a response with no data and a status of Access Denied is returned.
- If the Ignore Prohibited Access (IPA) bit in the request is set to 1, then the required response data with all values set to zero and status Success is returned. The purpose of this is to allow an address range to be probed without returning errors.

The read of a byte whose corresponding byte enable is 0 in the First DW BE or Last DW BE field should return a value of FFh.

IMPLEMENTATION NOTE — READ VALUE RETURNED ON UNUSED BYTE LANES

If a Management Entity receives a UCIe Memory Access Request packet with a byte enable value of 0 in the First DW BE or Last DW BE field and does not return a value of FFh for the byte in the corresponding response, then care must be exercised to ensure that the data returned in unused bytes does not create a security issue. Implementations are strongly encouraged to align secure information on DWORD or larger boundaries.

Figure 8-23 shows the fields of a UCIe Memory Access Response packet. Reserved fields (i.e., ones labeled as Rsvd) must be filled with 0s when the packet is formed. Reserved fields must be forwarded unmodified on the Management Network and ignored by receivers. An implementation that relies on the value of a reserved field in a packet is non-compliant.

Figure 8-23. UCIe Memory Access Response Packet

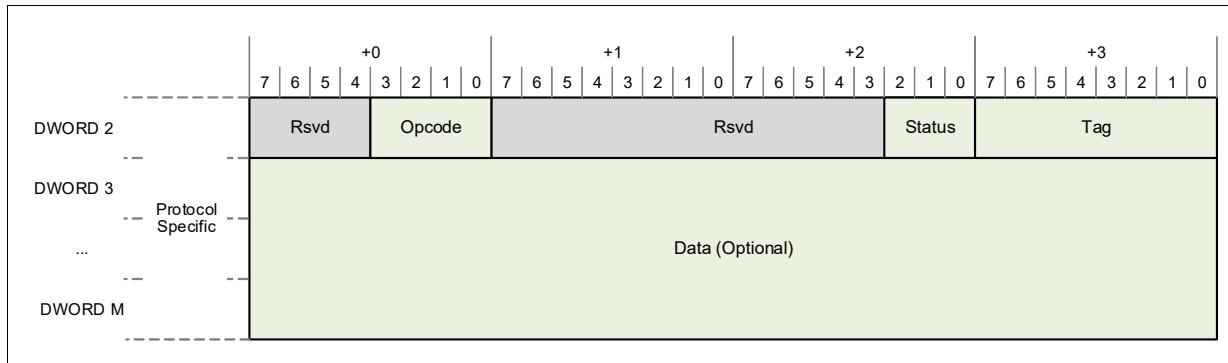


Table 8-20 defines the fields of a UCIe Memory Access Response packet. The packet starts at DWORD 2 because DWORDs 0 and 1 contain the UCIe Management Transport packet header. All fields in the table have little endian bit ordering (e.g., Tag bit 0 is in DWORD 3 bit 0, and Tag bit 7 is in DWORD 3 bit 7).

Table 8-20. UCIe Memory Access Response Packet Fields

Field Name	Field Size	Description
Opcode	4 bits	Opcode This field must be set to 0000b.
Status	3 bits	Response Status This field indicates the memory access response status. 000b: Success (SUCCESS) 001b: Programming Model Violation (PMV) 010b: Retry Request (RR) 011b: Access Denied (AD) 100b: Packet Error (PERR) Others: Reserved
Tag	8 bits	Tag This field contains the value of the tag field of the corresponding memory access request.
Data	Varies	Data If the memory access request was a Memory Read that was processed successfully (i.e., the Response Status field contains Success), then this field contains the data read. This field is not present in Memory Write completions.

8.1.4.2 UCIe Memory Access Protocol Capability Structure

A Management Entity that implements the UCIe Memory Access Protocol must implement the UCIe Memory Access Protocol Capability Structure described in this section.

The Max Buffered Requests field reports the maximum number of requests that the Management Entity is guaranteed to buffer. Issuing more outstanding requests to the Management Entity than this maximum may result in head-of-line blocking in the chiplet Management Fabric and/or a VC associated with a Management Port between chiplets.

The Request Buffer Size field reports the sum of the size of the requests that the Management Entity is guaranteed to buffer. Issuing more outstanding requests to the Management Entity than will fit in this buffer may result in head-of-line blocking in the chiplet Management Fabric and/or a VC associated with a Management Port between chiplets.

The Max Response Time Units and Max Response Time Value fields report the expected maximum time that the Management Entity requires to process a request. This is the expected maximum time with no other outstanding requests from receipt of a UCIe Memory Request packet at the Management Entity to the Management Entity emitting a corresponding UCIe Memory Response packet.

The UCIe Management Access Protocol does not define an architected completion timeout mechanism to detect lost packets or hardware failures; however, a requester may use the time reported in Max Response Time Units and Max Response Time Value fields in this capability structure to implement a vendor defined completion timeout mechanism. When a completion timeout mechanism is implemented, the requester must not declare a completion timeout sooner than the expected maximum response time reported by Response Time Units and Response Time Value fields.

Figure 8-24. UCIe Memory Access Protocol Capability Structure

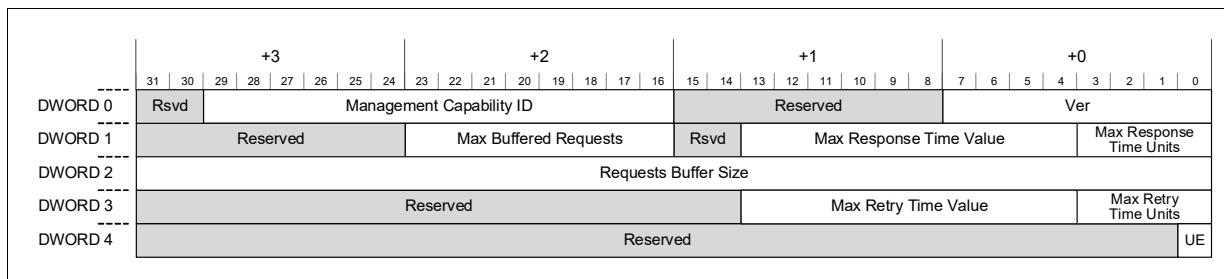


Table 8-21. UCIe Memory Access Protocol Capability Structure Fields (Sheet 1 of 2)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID ^a	Attribute	Description
Ver	0 [7:0]	17	RO	Capability Structure Version This field indicates the version of this capability structure. This field has a value of 00h in this specification.
Management Capability ID	0 [29:16]	17	RO	Management Capability ID This field specifies the Capability ID of this Management Capability Structure. The UCIe Memory Access Protocol Capability Structure has a Management Capability ID of 002h.
Max Response Time Units	1 [3:0]	17	RO	Maximum Response Time Units This field indicates the units associated with the Max Response Time Value field. 0000b: Reserved 0001b: nanoseconds (ns) 0010b: microseconds (us) 0011b: milliseconds (ms) 0100b: seconds (s) Others: Reserved

Table 8-21. UCIe Memory Access Protocol Capability Structure Fields (Sheet 2 of 2)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID ^a	Attribute	Description
Max Response Time Value	1 [13:4]	17	RO	Maximum Response Time Value This field indicates the expected maximum response time value. The units associated with this value are specified by the Max Response Time Units field.
Maximum Buffered Requests	1 [23:16]	17	RO	Maximum Number of Buffered Requests This field reports the maximum number of requests that the Management Entity can buffer. Requests that the Management Entity are currently processing are considered buffered request. A value of zero in this field indicates that the maximum number of buffered requests is not reported.
Request Buffer Size	2 [31:0]	17	RO	Request Buffer Size This field reports the size of the Management Entity request buffer in DWORDs. The number of DWORDs consumed by a request packet in this buffer is equal to the value specified by the Length field in the UCIe Management Transport packet header. A request packet that the Management Entity is currently processing consumes space in this buffer. A value of all 0s in this field indicates that the size of the request buffer is not reported.
Max Retry Time Units	3 [3:0]	17	RO	Maximum Retry Time Units This field indicates the units associated with the Max Retry Time Value field. 0000b: Reserved 0001b: nanoseconds (ns) 0010b: microseconds (us) 0011b: milliseconds (ms) 0100b: seconds (s) Others: Reserved
Max Retry Time Value	3 [13:4]	17	RO	Maximum Retry Time Value This field indicates the maximum duration of time during which a Management Entity may return a response with status Retry Request (i.e., maximum retry time). A value of 000h in this field indicates that the maximum retry time value is not reported.
UE	4 [0]	16	RW	Unordered Traffic Class Enable When this field is set to 1 the Management Entity may issue UCIe Memory Access protocol requests on an ordered or unordered traffic class. When this field is set to 0 the Management Entity may only issue requests on an ordered traffic class and must not issue requests on an unordered traffic class. The initial value of this field is 0 in all Management Entities except the Management Director. The initial value of this field is 1 in the Management Director.

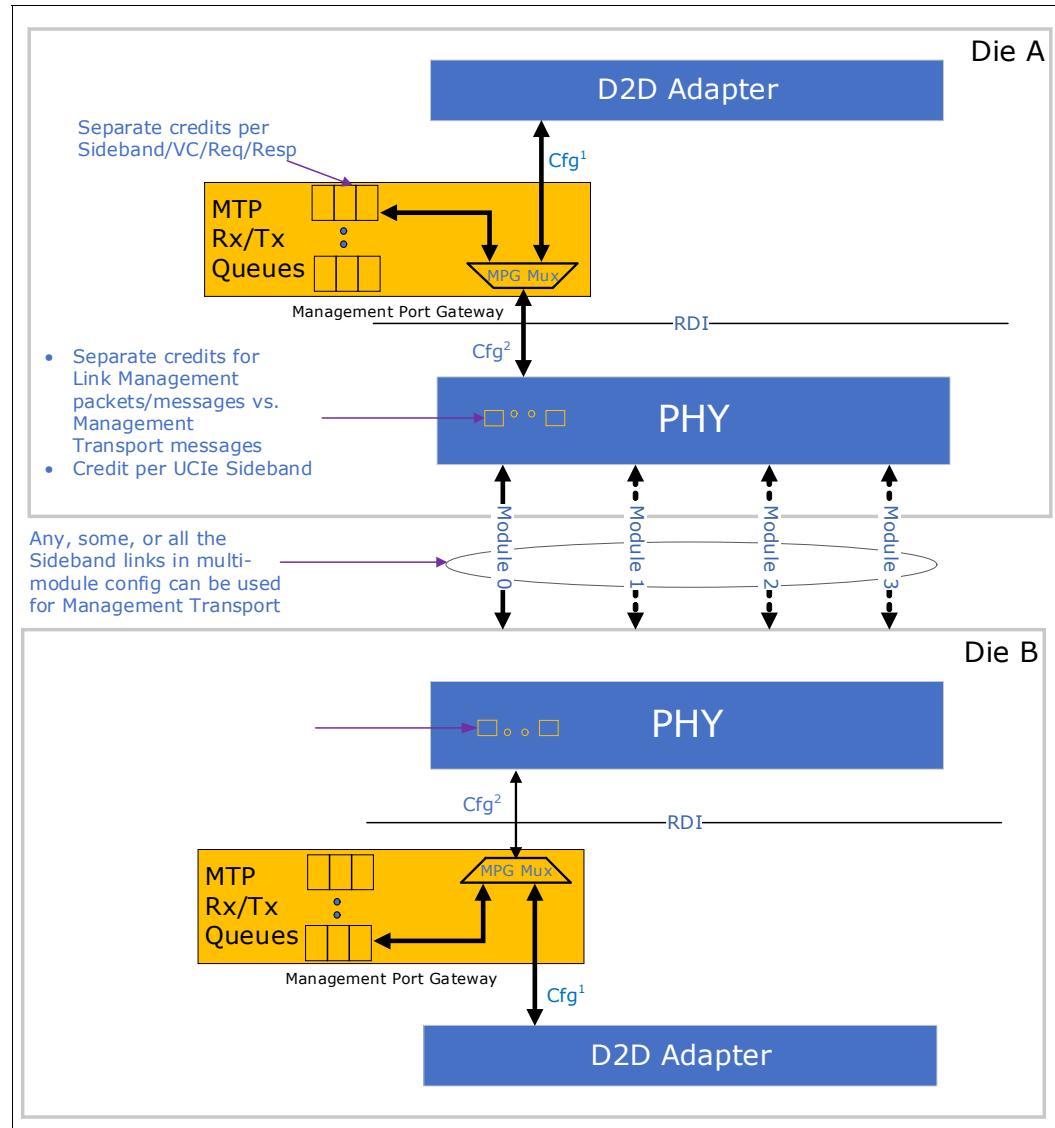
a. See [Table 8-7](#) for a description of Standard Security Asset Class IDs.

8.2 Management Transport Packet (MTP) Encapsulation

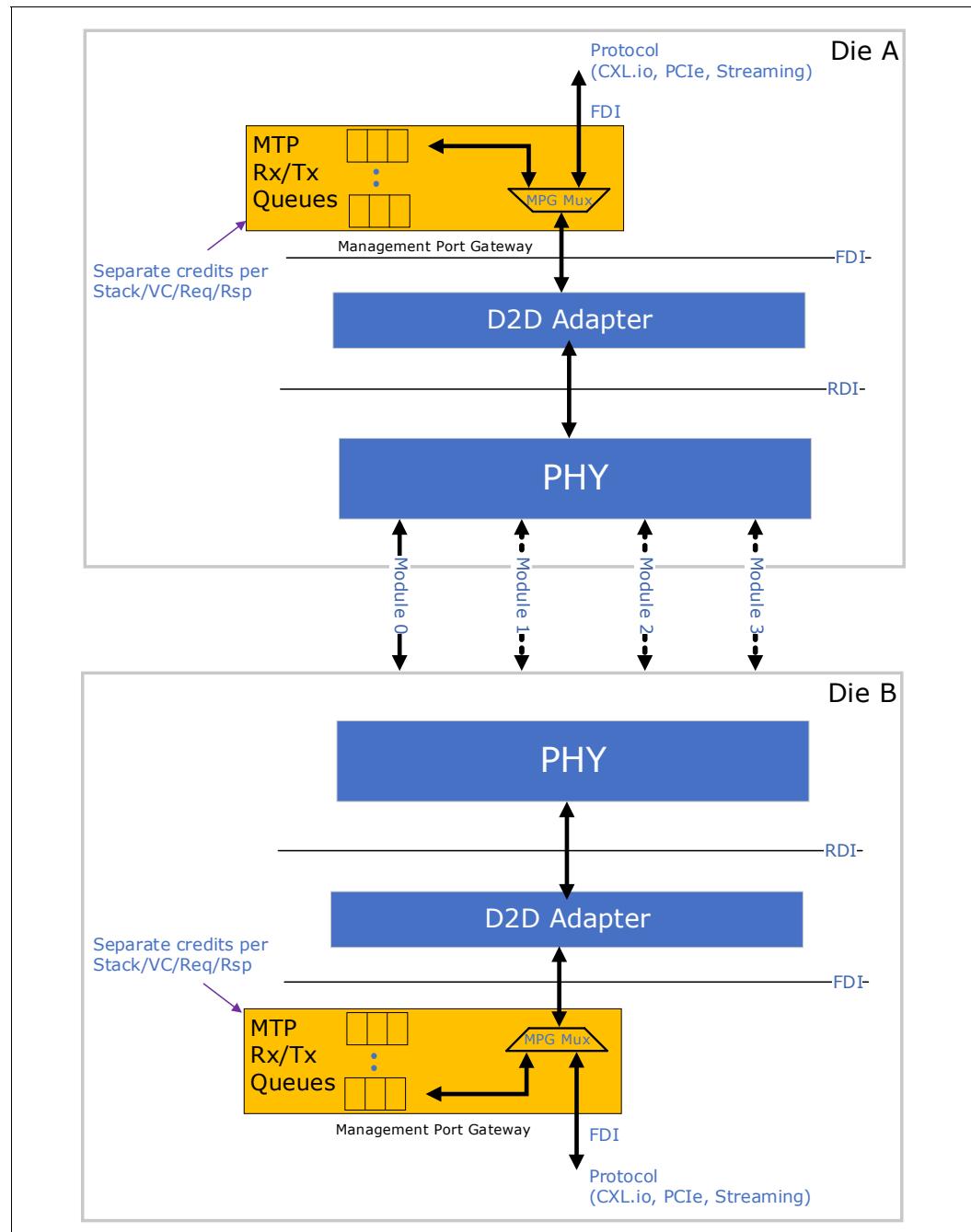
8.2.1 MTP Encapsulation Architecture Overview

Management Transport Packet (MTP) is the message used for management-related functionality on a management network in UCIe-based chiplets (see [Section 8.1.1](#) for details). This section deals with how MTPs are sent/received over UCIe Sideband and Mainband links through the process of Encapsulation. All Management Ports (as defined in [Section 8.1.2](#)) in a chiplet must support Encapsulation. Chiplet support for Management Port on a UCIe sideband link or a UCIe mainband link are independently optional, subject to rules stated in [Section 8.1.2](#). When Management Port is supported on a UCIe link (sideband or mainband) the management path on the link is setup as described in [Section 8.2.3.1](#) for sideband and [Section 8.2.3.2](#) for mainband. After the management path is set up on a link, the Encapsulated MTPs can be sent and received. Throughout this document the term Management Port Messages (MPM) is used to refer to all Sideband and Mainband messages that relate to Encapsulation. For the Sideband interface, these messages are defined in [Table 7-1](#). For the mainband, these messages are defined in [Table 8-22](#) and a "Management Flit" is defined (see [Table 3-4](#) and [Table 3-5](#)) to carry these messages.

See [Figure 8-25](#) and [Figure 8-26](#) for a high-level view of the MTP transport path over UCIe sideband and mainband respectively. In this architecture, MTPs are transported between Management Port Gateways (MPGs) on each end of the UCIe link using either the sideband or the mainband path. In this context, an MPG is an 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. The MPG has credited buffers for receiving MTPs (called RxQ) from the link and their sizes are exchanged during initial link training. These credited buffers are separately maintained for Sideband and Mainband paths when management transport is supported on both. Up to eight VCs can be supported on a Management Port. Dedicated buffering is required for each VC negotiated. Support for VC0 is mandatory when management transport is negotiated, and all other VCs are optional.

Figure 8-25. UCIe Sideband Management Path Architecture^{a b}

- Configuration interface (i.e., `p1_cfg_*` and `lp_cfg_*` signals) described in [Table 10-1](#).
- Configuration interface (i.e., `p1_cfg_*` and `lp_cfg_*` signals) described in [Table 10-1](#) plus extensions described in [Table 10-3](#).

Figure 8-26. UCIe Mainband Management Path Architecture

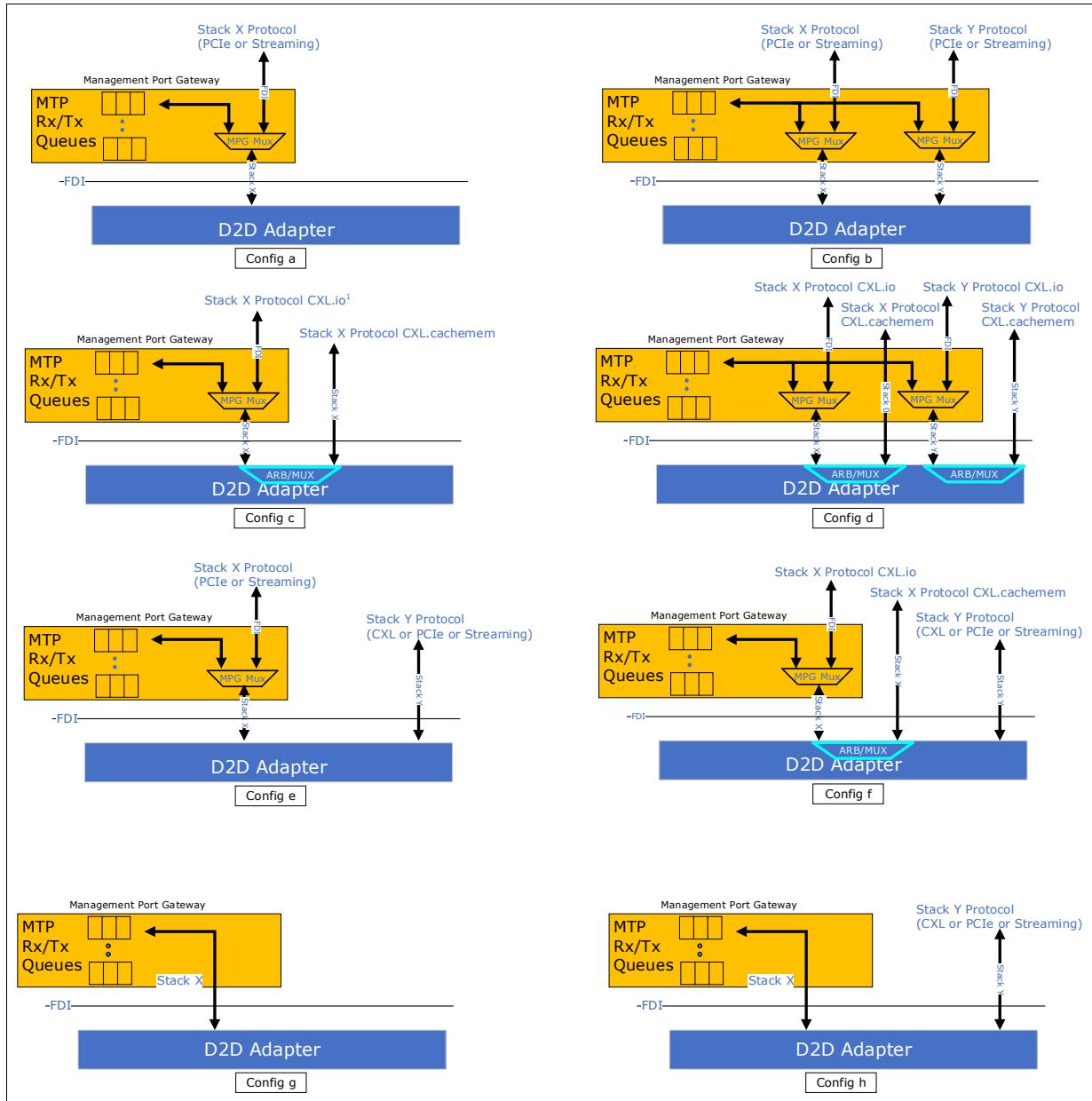
In multi-module or multi-sideband-only link configurations, any, some or all of up to four sideband links can be used for transporting MTPs. Unless stated otherwise, any references to sideband management port behavior/requirements/rules for a multi-module configuration also apply to a multi-sideband-only link configuration. In multi-stack mainband configurations, any or both stacks can be used for transporting MTPs. Ordering is still maintained when transporting packets on multiple sideband links/multiple stacks and this is described in [Section 8.2.4.3](#). Because MTPs can be large (up to 2K payload) and can block the sideband link for regular link management traffic (as described in [Table 7-1](#) except opcodes 10111b and 11000b), there is a mechanism provided to periodically arbitrate the link between link management packets/messages and MPMs over the sideband link. Additionally, to allow management path over sideband (when supported) to operate independent of mainband link status (which is required for certain management use cases such as FW download), UCIE link state machine supports sending/receiving management packets over sideband in all link states including RESET (see [Chapter 4.0](#) for details).

In [Figure 8-25](#) and [Figure 8-26](#), the location of the Management Port Gateway mux is shown for reference purposes only. Implementations can choose to locate the mux elsewhere (e.g., above FDI for sideband path) and details of such implementations are beyond the scope of this specification. Interface definitions for this architecture, seen in [Chapter 10.0](#), and other details discussed around Management Port Gateway integration are with respect to this reference Management Port Gateway mux placement.

The Management Port Gateway interfaces to the D2D Adapter by way of the FDI for mainband transport as shown in [Figure 8-26](#), and FDI is described in [Chapter 10.0](#). The Management Port Gateway can also connect directly to D2D by way of the FDI. Supported configurations of Management Port Gateway connectivity to D2D are shown in [Figure 8-27](#).

The terminology used throughout this chapter will be in reference to the concepts shown in [Figure 8-25](#) and [Figure 8-26](#). In case of CXL protocol, the Management Port Gateway mux is on the CXL.io FDI.

Figure 8-27. Supported Configurations for Management Port Gateway Connectivity to D2D Adapter on Mainband



8.2.2 Management Port Messages

8.2.2.1 Sideband

There are currently two MPM opcodes defined as shown in [Table 7-1, “Opcode Encodings Mapped to Packet Types”](#). See [Section 7.1.2.4](#) for more information.

8.2.2.2 Mainband

All MPMS on mainband carry a 2-DWORD header referred to as “MPM Header” (see [Figure 8-28](#) and [Figure 8-31](#)). In that Header, bits [4:0] in the first DWORD carry the MPM opcode and are defined in [Table 8-22](#). The remainder of this section discusses the format of these opcode messages. See [Section 8.2.5.2.3](#) for details of how these messages are packed in the Management Flit when transmitting over the mainband.

Table 8-22. MPM Opcodes on Mainband

Opcode	Message
10111b	MPM without Data
11000b	MPM with Data
Others	Reserved

8.2.2.2.1 MPMs with Data

Bits [21:14] in the first DWORD of the MPM header (see Figure 8-28) of an MPM with Data message form an 8b msgcode that denotes a specific MPM with Data message. Supported MPM with data messages on the mainband are shown in Table 8-23.

Table 8-23. Supported MPM with Data Messages on Mainband

msgcode	Message
01h	Encapsulated MTP Message
FFh	Vendor-defined Management Port Gateway Message
Others	Reserved

The term “MPM payload” is used in the remainder of this section to refer to the payload in the MPM with Data messages.

8.2.2.2.1.1 Common Fields in MPM Header of MPM with Data Messages on Mainband

Figure 8-28 shows and Table 8-24 describes the common fields in the MPM header of MPM with data messages on the mainband.

Figure 8-28. Common Fields in MPM Header of all MPM with Data Messages on Mainband

3			2						1						0																
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd			resp	vc		msgcode						length						rsvd	opcode = 11000b												
rsvd			msgcode-specific																rsvd	msgcode-specific	rsvd		rxqid								

Table 8-24. Common Fields in MPM Header of all MPM with Data Messages on Mainband (Sheet 1 of 2)

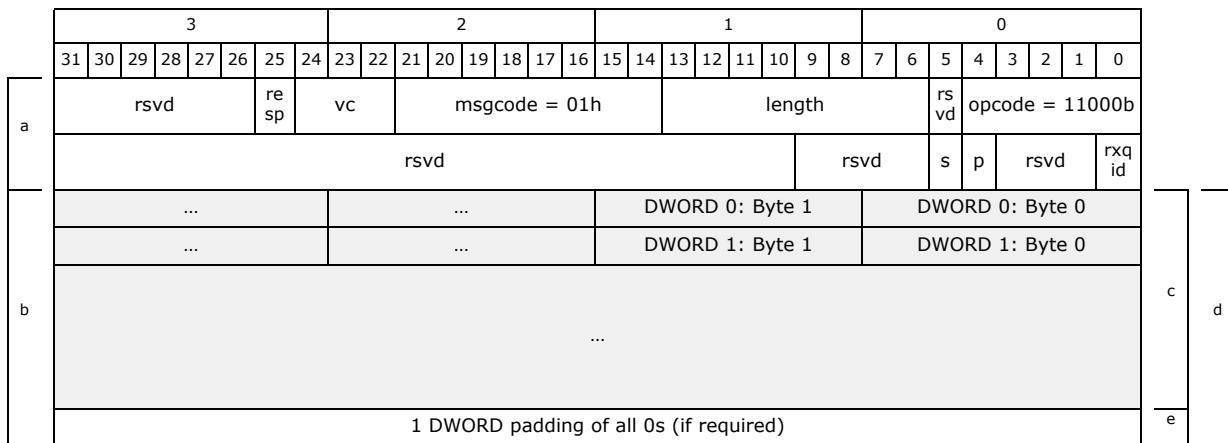
Field	Description
opcode	11000b: MPM with Data.
length	MPM Payload length (i.e., 0h for 1 QWORD, 1h for 2 QWORDs, 2h for 3 QWORDs, etc.).
msgcode	Message code as defined in Table 8-23 .

Table 8-24. Common Fields in MPM Header of all MPM with Data Messages on Mainband (Sheet 2 of 2)

Field	Description
vc	Virtual Channel ID.
resp	0: Request MPM. 1: Response MPM. For a Vendor-defined Management Port Gateway Message with Data, this bit is always 0 (see Section 8.2.2.2.1.3).
rxqid	RxQ-ID to which this packet is destined. See Section 8.2.3.2.2 for RxQ details. rxqid=0 corresponds to Stack 0. rxqid=1 corresponds to Stack 1.

8.2.2.2.1.2 Encapsulated MTP Message

Encapsulated MTP on the mainband is an MPM with Data with a msgcode of 01h.

Figure 8-29. Encapsulated MTP on Mainband

- a. MPM Header.
- b. MPM Payload.
- c. Management Transport Packet (MTP).
- d. Length in MPM Header.
- e. DWORD padding.

Table 8-25. Encapsulated MTP on Mainband Fields

Location	Bit	Description
MPM Header ^a	s	Segmented MTP (see Section 8.2.4.2). The first and middle segments in a segmented MTP have this bit set to 1. The last segment in a segmented MTP will have this bit cleared to 0. An unsegmented MTP also has this bit cleared to 0.
	p	1-DWORD padding of all 0s added at the end of the packet, if required to align to a QWORD boundary.
MPM Payload	—	See Section 8.1.3.1 for details. Note that DWORDx:Bytey in Figure 8-29 refers to the corresponding DWORD, Byte defined in the Management Transport Packet in Figure 8-5 .

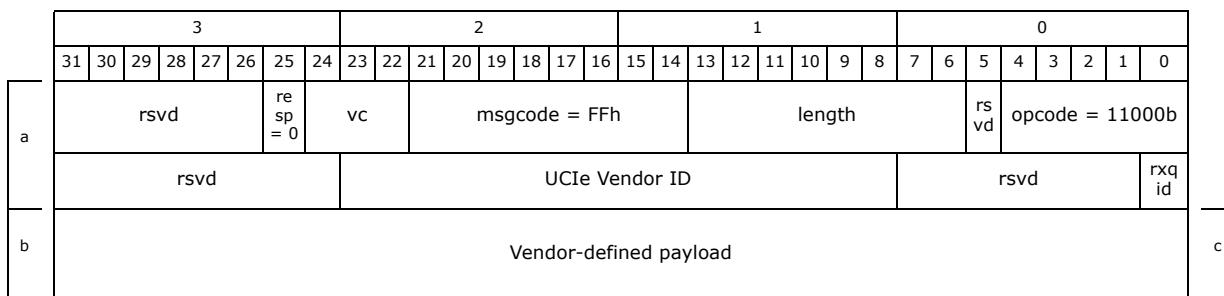
a. See [Section 8.2.2.2.1.1](#) for details of header fields common to all MPMs with data on the mainband.

8.2.2.2.1.3 Vendor-defined Management Port Gateway Message

The Vendor-defined Management Port Gateway message with data is defined for custom communication between MPGs on the two ends of a UCIe mainband link. These messages are not part

of the Management transport protocol, and these messages start at an MPG and terminate at the MPG on the other end of the UCIe mainband link. These messages share the same rxqid buffers as encapsulated MTP messages. If an MPG does not support these messages or does not support vendor-defined messages from a given vendor (identified by the UCIe Vendor ID in the header), the MPG silently drops those messages. Ordering of these messages sent over multiple mainband stacks is subject to the same rules presented in [Section 8.2.4.3](#) for encapsulated MTPs.

Figure 8-30. Vendor-defined Management Port Gateway Message with Data on Mainband



- a. MPM Header.
- b. MPM Payload.
- c. Length in MPM Header.

Table 8-26. Vendor-defined Management Port Gateway Message with Data on Mainband Fields

Location	Field	Description
MPM Header ^a	UCIe Vendor ID	UCIe Consortium-assigned unique ID for each vendor.
MPM Payload	—	Vendor-defined.

a. See [Section 8.2.2.2.1.1](#) for details of header fields common to all MPMs with data on the mainband.

8.2.2.2.2 MPMs without Data

Bits [21:14] in the first DWORD of the MPM header of an MPM without Data message form an 8b msgcode that denotes a specific MPM without Data message. [Table 8-27](#) lists the supported msgcodes.

Table 8-27. Supported MPM without Data Messages on Mainband

msgcode	Message
01h	Management Port Gateway Capabilities Message
03h	Init Done Message
FFh	Vendor-defined Management Port Gateway Message
Others	Reserved

8.2.2.2.2.1 Common Header Fields of MPM without Data Messages on Mainband

Figure 8-31 shows and Table 8-28 describes the common fields in the MPM header of MPM without data messages on the mainband.

Figure 8-31. Common Fields in MPM Header of all MPM without Data Messages on Mainband

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd	msgcode-specific	msgcode	msgcode-specific	rsvd	opcode = 10111b																										
rsvd	msgcode-specific	rsvd	msgcode-specific																												

Table 8-28. Common Fields in MPM Header of all MPM without Data Messages on Mainband

Field	Description
opcode	10111b: MPM without Data.
msgcode	Message code as defined in Table 8-27.

8.2.2.2.2.2 Management Port Gateway Capabilities Message

See Section 8.2.3.2.2 for details of how this message is used during mainband management path initialization.

Figure 8-32 shows and Table 8-29 describes the Management Port Gateway Capabilities message format on the mainband.

Figure 8-32. Management Port Gateway Capabilities MPM on Mainband

a	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd	msgcode = 01h	NumVC	rsvd	opcode = 10111b																											
	rsvd	Port ID[15:0]	rsvd	rsvd																												

a. MPM Header.

Table 8-29. Management Port Gateway Capabilities MPM Header Fields on Mainband^a

Field	Description
NumVC	Number of VCs supported by the Management Port Gateway that is transmitting the message.
Port ID	Port ID number value of the Management port associated with the Management Port Gateway that is issuing the message (see Section 8.1.3.6.2.1).

a. See Section 8.2.2.2.2.1 for details of header fields common to all MPMS without data on the mainband.

8.2.2.2.2.3 Init Done Message

See Section 8.2.3.2.2 for details of how this message is used during mainband management path initialization.

Figure 8-33 shows and Table 8-30 describes the Init Done message format on the mainband.

Figure 8-33. Init Done MPM on Mainband

3			2								1								0																
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
a	rsvd								msgcode = 03h								rsvd								opcode = 10111b										
	rsvd																																	rxqid	d

a. MPM Header.

Table 8-30. Init Done MPM Header Fields on Mainband^a

Field	Description
rxqid	RxQ-ID associated with the message. See Section 8.2.3.2.2 for RxQ details.

a. See Section 8.2.2.2.2.1 for details of header fields common to all MPMs without data on the mainband.

8.2.2.2.2.4 Vendor-defined Management Port Gateway Message

The Vendor-defined Management Port Gateway message without data is defined for custom communication between the MPGs on both ends of a PCIe mainband link. These messages are not part of the management transport protocol, and these messages start at an MPG and terminate at the MPG on the other end of the PCIe mainband link. These messages share the same rxqid buffers as encapsulated MTP messages. If an MPG does not support these messages or does not support these messages from a given vendor (identified by the PCIe Vendor ID in the header), the MPG silently drops those messages.

The Vendor-defined Management Port Gateway message without data on the mainband has the format shown in Figure 8-33.

Figure 8-34. Vendor-defined Management Port Gateway Message without Data on Mainband

3			2								1								0													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
a	rsvd				re sp = 0	vc		msgcode = FFh								Vendor-defined								rs vd	opcode = 10111b							
	rsvd								PCIe Vendor ID																rsvd				rxqid	d		

a. MPM Header.

Table 8-31. MPM Header Vendor-defined Management Port Gateway Message without Data on Mainband^a

Field	Description
vc	Virtual Channel ID.
resp	Vendor-defined Management Port Gateway message without data always uses the Request channel.
PCIe Vendor ID	PCIe Consortium-assigned unique ID for each vendor.
rxqid	RxQ-ID to which this packet is destined. See Section 8.2.3.2.2 for RxQ details.

a. See [Section 8.2.2.2.1](#) for details of header fields common to all MPMS without data on the mainband.

8.2.3 Management Transport Path Setup

Management transport path setup occurs in two distinct phases:

- **Negotiation phase** — In this phase, support for management transport, and when supported, the number of RxQs present in the partner chiplet, are negotiated. This is required for backward compatibility. This negotiation is done separately for sideband and mainband.
- **Initialization phase** — In this phase, the number of VCs supported is negotiated, and the RxQs in the Management Port Gateways on both ends of the link are initialized through credit exchanges for each supported VC. Port IDs are also exchanged.

[Section 8.2.3.1](#) describes the setup process for the sideband. [Section 8.2.3.2](#) describes the setup process for the mainband.

8.2.3.1 Sideband

Sideband Management Transport path setup occurs after a Management Reset or when Software writes 1 to the ‘Retrain Link’ bit in the Sideband Management Port Structure register (see [Section 8.1.3.6.2.1](#)). After setup is complete, management transport path over sideband remains active until the next Management Reset or until a ‘Heartbeat timeout’ is detected (as described in [Section 8.2.5.1.3](#)).

8.2.3.1.1 Negotiation Phase Steps

Negotiation occurs in the MBINIT.PARAM state. See [Section 4.5.3.3.1.1](#) for details.

8.2.3.1.2 Initialization Phase Steps

If the Negotiation phase indicates support for Management transport and the SB_MGMT_UP flag (see [Section 4.5](#)) is cleared, Initialization phase steps are performed as indicated in this section.

A few general rules for RxQs that are initialized in this phase:

- Management Port Gateway maintains separate Rx queues for each sideband link over which it can receive MPMS. The Management Port Gateway can limit the number of Rx queues to be the same or smaller than the number of modules in the design. For example, in a design with four modules, a Management Port Gateway can choose to limit Rx queues to three or two or one.
- Each Rx queue in the Management Port Gateway is assigned a separate RxQ-ID and it is relevant for maintaining ordering when interleaving MTPs across multiple sideband links. See [Section 8.2.4.3](#).
- See [Section 8.2.4.1](#) for details of credit buffers that are required in each Rx queue.

- Number of RxQs finalized for transmitting and receiving MPMs is 0 to MIN{RxQ-Local, RxQ-Remote}, where RxQ-Local and RxQ-Remote are defined in [Section 4.5.3.3.1.1](#).
- Transmission of MPMs with a given RxQ-ID is always associated with a specific local module that is design-specific. For example, an MPM with an RxQ-ID of 0 can be sent on any Module's sideband and that choice is design-specific. However, the choice is static and cannot change after the first MPM with that RxQ-ID is sent.
- Credits associated with each RxQ-ID are exchanged with a remote link partner by way of Credit Return messages as discussed below.

Initialization phase steps:

After **pm_param_done** is asserted and there is >0 module count negotiated for management transport for the local and remote sides, the Management Port Gateway begins the initialization process to the remote MPG for each RxQ-ID that the MPG needs to enable.

1. The initialization phase starts (shown in [Figure 8-35](#), [Figure 8-36](#), and [Figure 8-37](#)) with each Management Port Gateway sending the Management Port Gateway Capabilities message (see [Figure 7-9](#) for message format).
 - Message can be sent on any RxQ-ID path, but sent only once per initialization phase from a chiplet to the partner chiplet.
 - Port ID value in the transmitted message is the value in Port ID field (see [Table 8-12](#)).
 - Port ID value in the received message is recorded in the "Remote Port ID" field (see [Table 8-12](#)).
 - NumVC field is the number of VCs supported by the transmitting Management Port Gateway. The number of VCs supported is the value in the NumVC field + 1. For example, if only one VC (VC0) is supported, NumVC is 0h. If two VCs are supported (VC0, VC1), then NumVC is 1h, etc.
 - MIN{Transmitted NumVC, Received NumVC}+1 number of VCs is enabled by each Management Port Gateway in the subsequent steps. The value of the enabled VCs starts from 0 and increments by 1 for each enabled VC up to MIN{Transmitted NumVC, Received NumVC}.
2. Management Port Gateway then sends credit Return messages for each enabled VC for each type (requests and responses), across all enabled RxQ-IDs. The Management Port Gateway is permitted to send this message. [Figure 8-35](#) shows the flow for the case of only a single RxQ (RxQ-ID=0) and single VC (VC0) negotiated during the negotiation phase. [Figure 8-36](#) shows the flow for the case of two RxQs (RxQ-ID=0, 1) and single VC (VC0) negotiated during the negotiation phase. [Figure 8-37](#) shows the flow for the case of only a single RxQ (RxQ-ID=0) and two VCs (VC0, VC1) negotiated during the negotiation phase.
 - Credit Return message (see [Figure 7-10](#)) contains an "RxQ-ID" field. The field must be assigned starting from 0 to MIN{RxQ-Local, RxQ-Remote}-1.
 - Infinite credits are permitted to be advertised. This is performed by sending a value of 3FFh in the "Rx Credit return in QWORDs" field for that VC and Type before the "Init Done".

Figure 8-35. Sideband Management Transport Initialization Phase Example with RxQ-ID=0 and One VC (VC0)

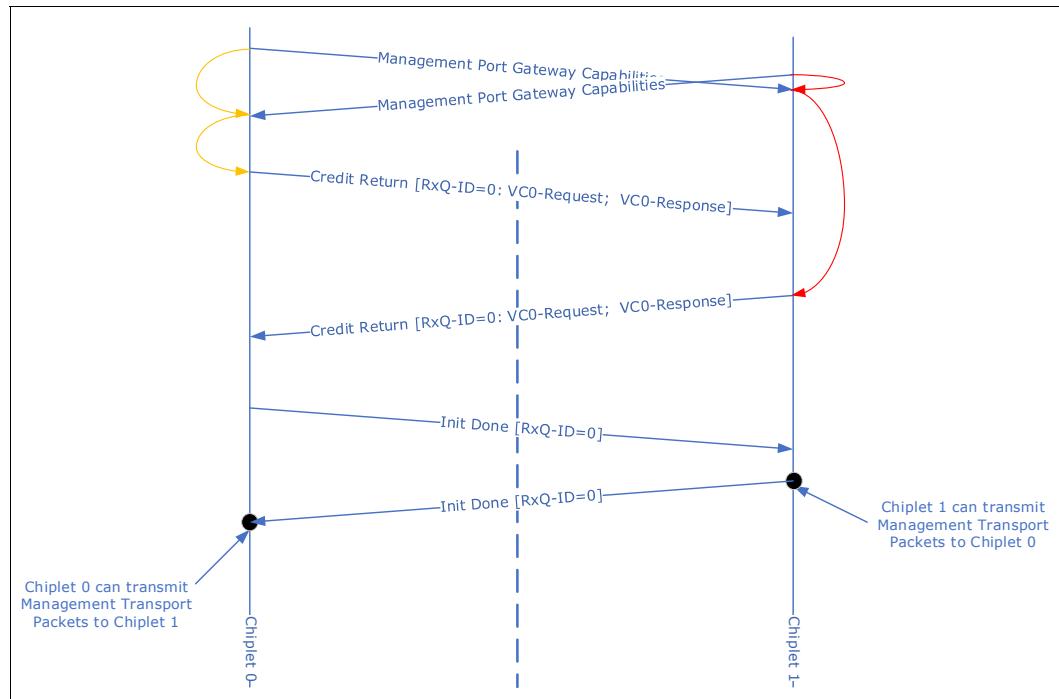


Figure 8-36. Sideband Management Transport Initialization Phase Example with RxQ-ID=0, 1 and One VC (VC0)

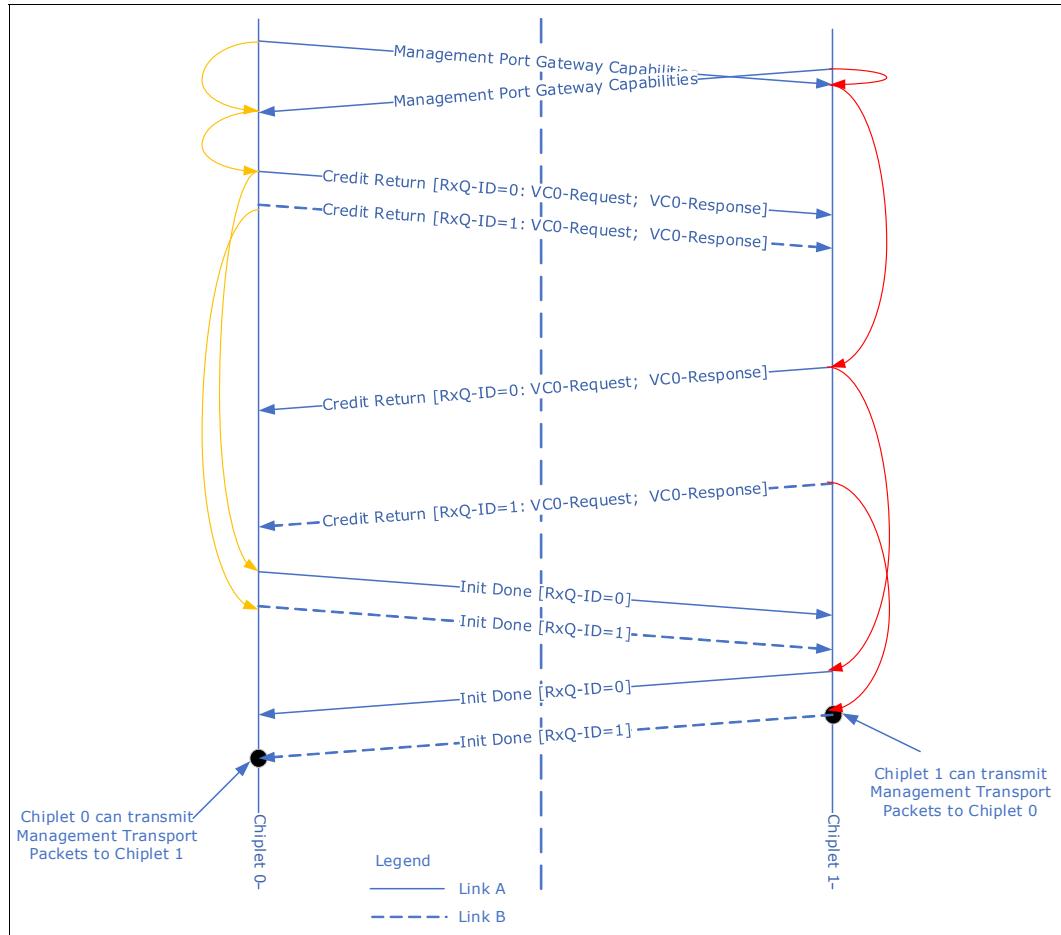
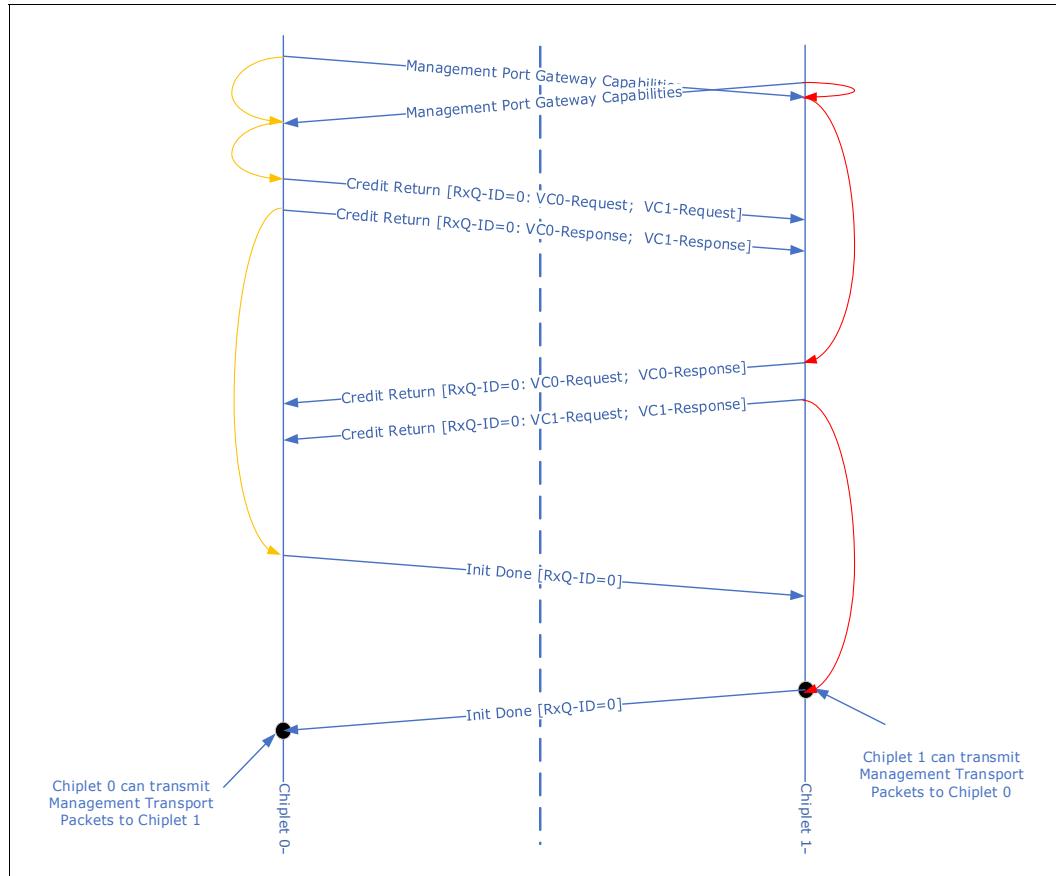


Figure 8-37. Sideband Management Transport Initialization Phase Example with RxQ-ID=0 and Two VCs (VC0, VC1)



3. After the last Credit Return message for a given RxQ-ID, the Management Port Gateway must send an "Init Done" message (see [Figure 7-11](#)) for the corresponding RxQ-ID. This informs the remote Link partner that a receiver has finished advertising credits for enabled VCs for the given RxQ-ID.
 - After "Init Done" has been transmitted and received by a Management Port Gateway for all available RxQ-ID paths, the MPG is ready for sending Management Transport packets.
 - o Sideband should be able to send/receive management transport packets at this point without any dependency on the mainband link status.
 - o Management Port Gateway asserts the `mp_mgmt_up` and `mp_mgmt_init_done` signals to PHY to indicate that the Management Transport Path was successfully initialized. PHY sets the `SB_MGMT_UP` flag when both `mp_mgmt_up` and `mp_mgmt_init_done` are asserted. The flag remains set until the management path goes down. In case of any fatal error (e.g., credit return messages were received for an RxQ-ID that is not expected, a timeout occurred while waiting for the Init Done message, etc.) during RxQ credit exchange, the `mp_mgmt_up` signal will remain de-asserted with the `mp_mgmt_init_done` signal asserted.
 - o Note that the Management Port Gateway is unaware of PHY states and thus, after the `mp_mgmt_up` signal is asserted, the Management Port Gateway assumes that the

management path through the sideband is available unless there is a Management Reset or the MPG detects an error through the mechanism described in [Section 8.2.5.1.3](#).

- o After the SB_MGMT_UP flag is set, sideband link is available for sideband packet (MPMs or any other sideband packets) transmission/reception in all state machine states including RESET/SBINIT.
 - After the Management Port Gateway receives the “Init Done” message for a given RxQ-ID, the MPG must be ready to receive MTPs with that RxQ-ID.

The PHY Layer routes a message with a given RxQ-ID (specified by the `mp_rxqid` signal) to a specific module’s sideband link and that association is design-specific. Note that because RxQ-ID association to a module sideband is design-specific, on the same sideband link, messages with different RxQ-IDs in each direction are possible.

8.2.3.1.3 Other Sideband Management Transport Path Rules

- Sideband interfaces successfully initialized for management transport are available for management transport regardless of the associated mainband module’s state.
 - Note that when management transport is NOT supported and Module 0’s mainband is disabled at runtime, the sideband interface on that module is also disabled and D2D messages are routed to the sideband interface of the next-available lowest-numbered module that is enabled. When management transport is supported and enabled on the sideband link, the sideband link remains active for both management and non-management packets even if the corresponding mainband module is disabled.
- If SW writes 1 to the ‘Retrain Link’ bit in the Management Port Structure register associated with a sideband link when the Management Path is already up on that port, the Management Port Gateway must follow the ‘Heartbeat timeout’ flow (see [Section 8.2.5.1.3](#)) to bring the management path down before instructing the PHY to restart link negotiation (by the `sb_mgmt_init_start` signal).

8.2.3.2 Mainband

Mainband Management Transport path setup occurs when a link trains up. After the setup is complete, the management transport path remains active until a Domain Reset or until the link or the associated stack(s) goes down.

8.2.3.2.1 Negotiation Phase Steps

Mainband Management Transport path negotiation occurs on every mainband link training, thereby leveraging the existing D2D adapter protocol negotiation messages/flows. Support for Management Transport protocol within a stack is explicitly indicated with a new bit in the negotiation flow (see [Table 3-1](#)).

[Section 3.1](#) and [Section 3.2](#) provide Management Transport protocol negotiation details. At the end of protocol negotiation, the D2D adapter indicates the number of D2D stacks that negotiated Management Transport protocol by signals discussed in [Table 10-3](#).

8.2.3.2.2 Initialization Phase Steps

A few general rules for the RxQs that are initialized in this phase:

- Management Port Gateway maintains separate Rx queues for receiving MTPs over each negotiated stack.

- Each Rx queue within the Management Port Gateway is assigned a separate RxQ-ID, which is necessary for maintaining ordering when interleaving packets across multiple stacks. See [Section 8.2.4.3](#).
- See [Section 8.2.4.1](#) for details of credit buffers that are required in each Rx queue.
- RxQ-ID values are either 0 or 1. A value of 0 is used if only one stack is negotiated for management transport (regardless of the stack-id value negotiated) and values of 0 and 1 are used if two stacks are negotiated for management transport. In the latter case, an RxQ-ID value of 0 is used for Stack 0, and an RxQ-ID of 1 is used for Stack 1.

When FDI transitions to active state (`p1_state_sts=Active`) from reset state (`p1_state_sts=Reset`) and Management transport was negotiated, the Management Port Gateway starts the Initialization phase. In a multi-stack implementation where management transport is present on both stacks, the D2D adapter flit negotiation, and protocol negotiation across both stacks must have completed (as indicated by the `dm_param_exchange_done` signal) before the Management Port Gateway starts its initialization sequence.

The initialization flow follows the similar sequence as sideband and some example flows are illustrated below. The credit exchange is not by way of an explicit message as in sideband, but rather by way of a dedicated DWORD, 'CRD', in management flits whose format is shown in [Figure 8-45](#) and further explained in [Chapter 3.0](#). Management Port Gateway Capabilities and Init Done Message formats for the mainband can be seen in [Section 8.2.2.2.2](#). Note that during initialization, the transmitter can return valid credits in the same Management Flit that carries the Init Done message. All protocol layer bytes in the management flit (minus the CRD and Rsvd bytes) carrying the 'Init Done' MPM are driven with NOPs after the 'Init Done' MPM.

In [Figure 8-38](#), [Figure 8-39](#), and [Figure 8-40](#), the labeling

```
Mgmt_Flit {<MPM>, CRD[<credits returned>]}
```

refers to a Management Flit that carries the specified MPM along with credit returns for the indicated credit types. For example:

```
Mgmt_Flit {Init Done, CRD[RxQ-ID=0: VC0-Request, VC0-Response]}
```

indicates a Management Flit that carries the Init Done message along with credit returns for the VC0 request and response credit types for RxQ-ID=0.

Figure 8-38. Mainband Management Transport Initialization Phase Example with RxQ-ID=0 and One VC (VC0)

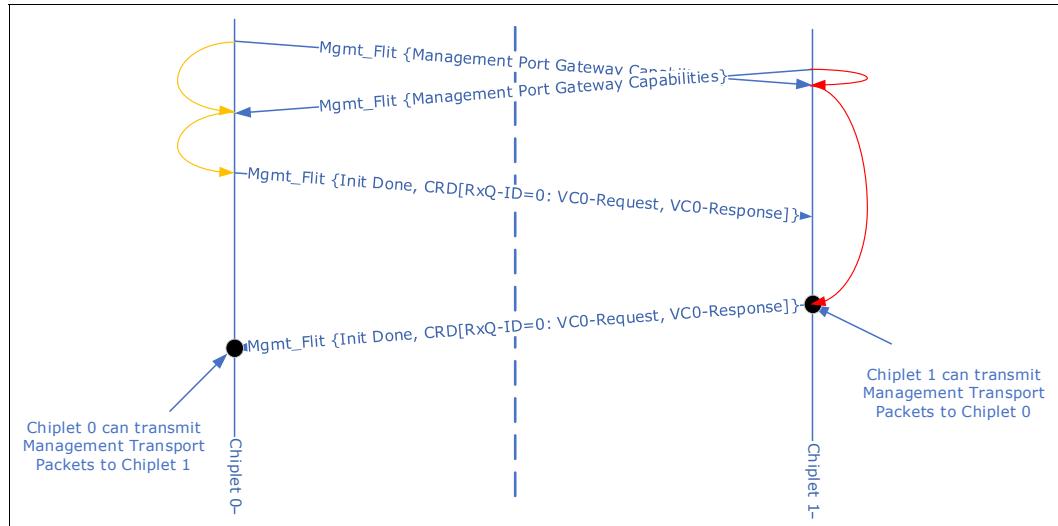


Figure 8-39. Mainband Management Transport Initialization Phase Example with RxQ-ID=0, 1 and One VC (VC0)

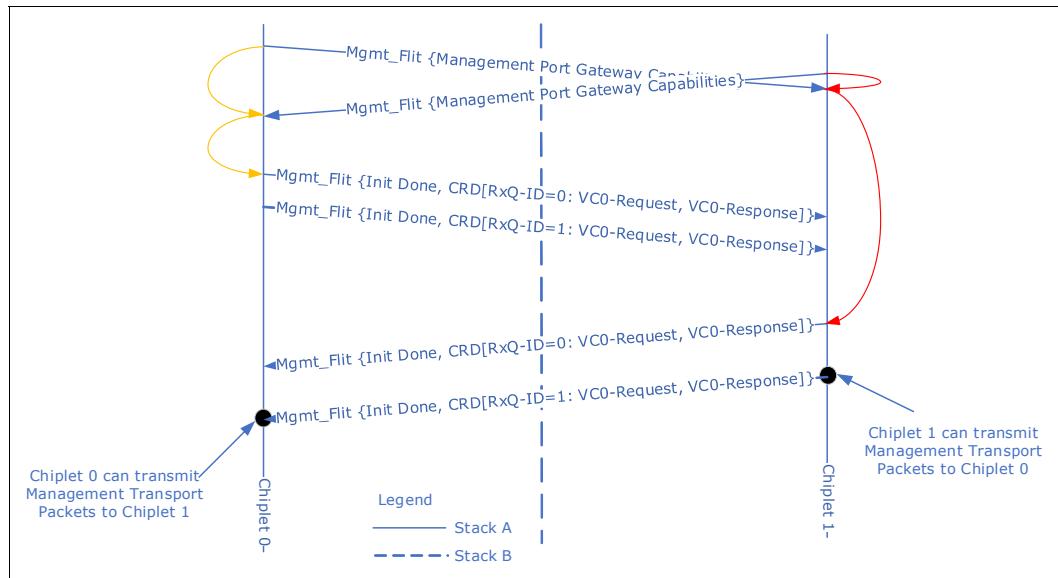
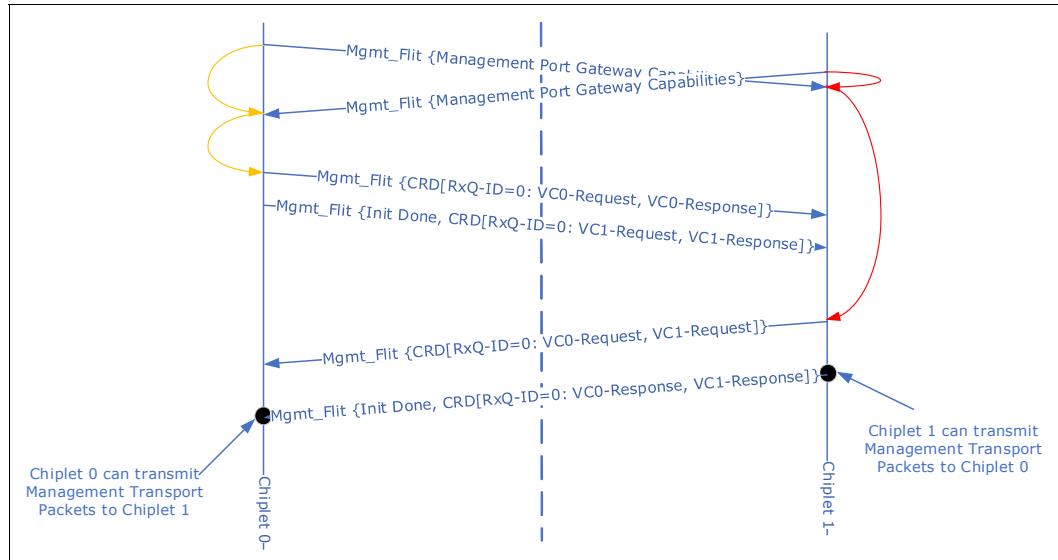


Figure 8-40. Mainband Management Transport Initialization Phase Example with RxQ-ID=0 and Two VCs (VC0, VC1)



8.2.3.2.3 Other Mainband Management Transport Path Rules

The following rules relate to Management Port Gateways and mainband Management Transport:

- During runtime, if the FDI status on any stack that has management traffic negotiated moves to a Link Status=down state, the Management Port Gateway behaves the same as in the 'Init Done' timeout scenario (see [Section 8.2.4.4](#)) across both stacks, if more than one stack had management transport negotiated.
- Arbitration between Management Flits and regular Protocol Layer Flits is implementation-specific.
- When management Software writes 1 to the 'Retrain link' bit in the Management Port Structure register that corresponds to the mainband link, the mainband is retrained, similar to when SW writes 1 to the 'Start UCIe Link Training' bit in the UCIe Link Control register in the UCIe link DVSEC. Note that this retraining of the mainband does not affect the management path on the sideband (if that path had been negotiated), if the path was already set up and active.

8.2.4 Common Rules for Management Transport over Sideband and Mainband

8.2.4.1 Management Packet Flow Control

The rules for management transport flow control are as follows:

- Forward progress and Flow Control are managed by the Management Port Gateways.
- Flow control credits are independent for sideband and mainband paths, if both are implemented in a given UCIe port.
- Encapsulated MTPs are credited, and the credits cover both the header and payload portions of the encapsulated MTP.
- Management Port Gateway Capability message, Credit return messages, Init Done messages, and PM-related messages must sink unconditionally at the destination.

- Although the number of VCs supported in both directions is the same, TC-to-VC mapping can be different in each direction. See the Route Entry description in [Section 8.1.3.6.2.2](#) for how SW controls mapping of TC to VC.
- For each RxQ-ID in the Management Port Gateway:
 - Independent credit management is required for each resp type (Requests vs. Responses), and each supported VC.
 - Credits are in QWORD (64-bit) granularity (i.e., one credit corresponds to one QWORD of storage space at the receiver buffer).
 - Minimum three credits are required for each credit type when nonzero credits are advertised.
 - Header and Data portions of an Encapsulated Management Transport packet and Vendor-defined Management Port Gateway Messages use the same type of credit.
 - Receiver implements separate buffers for Requests and Responses per supported VC and advertises the corresponding credits to the remote Management Port Gateway during initialization. Credits are returned when space is freed up in the receiver buffers.
 - Up to eight VCs are permitted — different VC counts are permitted on sideband vs. mainband.
 - Support for VC0 is mandatory for all implementations.
 - For every VC supported, it is mandatory to initialize credits for Request types and Response types.
 - Credits advertised for a VC:Resp credit type during the initialization phase can be either 0 across all RxQ-IDs or nonzero across all RxQ-IDs.
 - If a VC is initialized, credits for that VC must be advertised on all enabled RxQ-IDs and Resp types. For example, it is NOT permitted to have a configuration where VC1 is supported on RxQ-ID 0 but not on RxQ-ID 1. However, it is not required to advertise the same number of credits on all enabled Paths. This rule is important to simplify Transmitter/Receiver implementations at Management Port Gateways for interleaving MTPs across multiple Links while maintaining ordering across them (see 1.4.3 for concept of interleaving).
 - During management transport initialization and before the Init Done message is received, if multiple credit returns are received for the same VC:Resp credit type, the value from the latest credit return overwrites the previous value.
- Number of RxQs (in the partner chiplet's Management Port Gateway) to which a Management Port Gateway can transmit management messages is always same as the number of RxQs to which the MPG can receive these messages (from the partner chiplet's Management Port Gateway). For example, if two RxQs were negotiated, both send and receive of management traffic must be on two RxQs.
- If the initial credit advertised was infinite for a credit type, there cannot be any credits returned for that type at run time (i.e., after the Init Done message has been sent), with one exception for the "VC0 request infinite credit" scenario for which a runtime credit return of 0 is permitted.
- Credits advertised during the initialization phase are the maximum number of credits that can be outstanding at the transmitter at any point during runtime.
- See [Section 8.2.4.3](#) for the rules for maintaining ordering when interleaving MTPs/MTP Segments across different RxQ-IDs.
- Chiplets can optionally check for the following error conditions during management path initialization flow, and abort the flow when these conditions are detected:
 - Receiving credit returns for more RxQ-IDs than what was negotiated in the Negotiation Phase.

- Receiving credit returns for more VCs than what was implicitly negotiated by the Management Port Gateway Capabilities message.
- Not receiving credit returns or receiving incomplete credit returns for any of the negotiated RxQ-IDs prior to receiving the ‘Init Done’ message for the RxQ-ID.

8.2.4.2 Segmentation

The Management Port Gateway is permitted to break up (i.e., segment) one large MTP and send the individual segments across multiple RxQ-IDs (i.e., interleave; see [Figure 8-41](#) for an example). This is useful for cases in which the MTP message sizes are asymmetric. When segmenting:

- Management Port Gateway sets the s bit in the Encapsulated MTP message header within each individual segment except the last segment that completes the MTP transfer. If an MTP is not segmented, the s bit is 0. Segments with the s bit set to 1 must not also have the p bit set to 1.
- Transmitter must ensure that no other Encapsulated MTP OR no other credited MPM packet (e.g., Vendor-defined Management Port Gateway messages), from the same VC:Resp credit type is interleaved until the segmented management packet completes.

Note that segmentation is visible only from Management Port Gateway-to-Management Port Gateway and is not end-to-end on the UCIe Management Fabric.

See [Section 8.2.4.3](#) for the rules for reassembling the segments and maintaining ordering when interleaving Segments across different RxQ-IDs.

Figure 8-41. Example Illustration of a Large MTP Transmitted over Multiple RxQ-IDs on Sideband with Segmentation^a

Management Transport Packet (MTP)		1st Segment^b — This goes on RxQ-ID=x	
QWORD 0	MTP Header	QWORD 0	MPM Header (s = 1, length = 6h)
QWORD 1	MTP Data 0	QWORD 1	MTP Header
QWORD 2	MTP Data 1	QWORD 2	MTP Data 0
QWORD 3	MTP Data 2	QWORD 3	MTP Data 1
QWORD 4	MTP Data 3	QWORD 4	MTP Data 2
QWORD 5	MTP Data 4	QWORD 5	MTP Data 3
QWORD 6	MTP Data 5	QWORD 6	MTP Data 4
QWORD 7	MTP Data 6	QWORD 7	MTP Data 5
QWORD 8	MTP Data 7		
QWORD 9	MTP Data 8		
QWORD 10	MTP Data 9		
QWORD 11	MTP Data 10		
QWORD 12	MTP Data 11		
QWORD 13	MTP Data 12		
QWORD 14	MTP Data 13		
QWORD 15	MTP Data 14		

2nd Segment^b — This goes on RxQ-ID=MOD((x+1)/N)	
QWORD 8	MPM Header (s = 1, length = 6h)
QWORD 9	MTP Data 6
QWORD 10	MTP Data 7
QWORD 11	MTP Data 8
QWORD 12	MTP Data 9
QWORD 13	MTP Data 10
QWORD 14	MTP Data 11
QWORD 15	MTP Data 12

3rd Segment^b — This goes on RxQ-ID=MOD((x+2)/N)	
QWORD 0	MPM Header (s = 0, length = 1h)
QWORD 1	MTP Data 13
QWORD 2	MTP Data 14

a. N = Number of RxQ-IDs negotiated.

x = Start value of RxQ-ID for an MTP.

b. A segment is an Encapsulated MTP with its s bit set to 1.

8.2.4.3 Interleaving and Multi-module Sideband and Multi-stack Mainband Ordering

When multiple RxQ-IDs are negotiated, the Management Port Gateway must interleave different MTPs of a given VC:Resp credit type across the different RxQ-IDs. For example, when the transmitter does not support Segmentation (see Section 8.2.4.2), if there are two MTPs, Pkt 1 and Pkt 2, both on VC0 and of Resp=0 type and two RxQ-IDs were negotiated, these must be sent on two different RXQ-IDs. This is called interleaving. When the transmitter supports Segmentation, individual Segments are also interleaved. This section discusses transmitter and receiver rules when interleaving so that the original management packet ordering (see Section 8.1.3.1.1) is maintained when the MTPs eventually make it to the management network on the receiving partner chiplet. For the purposes of discussing these rules in this section, the nomenclature of RxQ-ID_x:VC_y:Resp_z is used to refer to the credit buffer of RxQ-ID=x (x=0 to 3), VC_y (y=0-7) and Resp_z (z=0 for Request and 1 for Response) type.

8.2.4.3.1 Transmitter Rules

- First Encapsulated MTP after Management path setup, of a given VC_y:Resp_z credit type is transmitted to the RxQ-ID0:VC_y:Resp_z credit buffers of the partner chiplet.

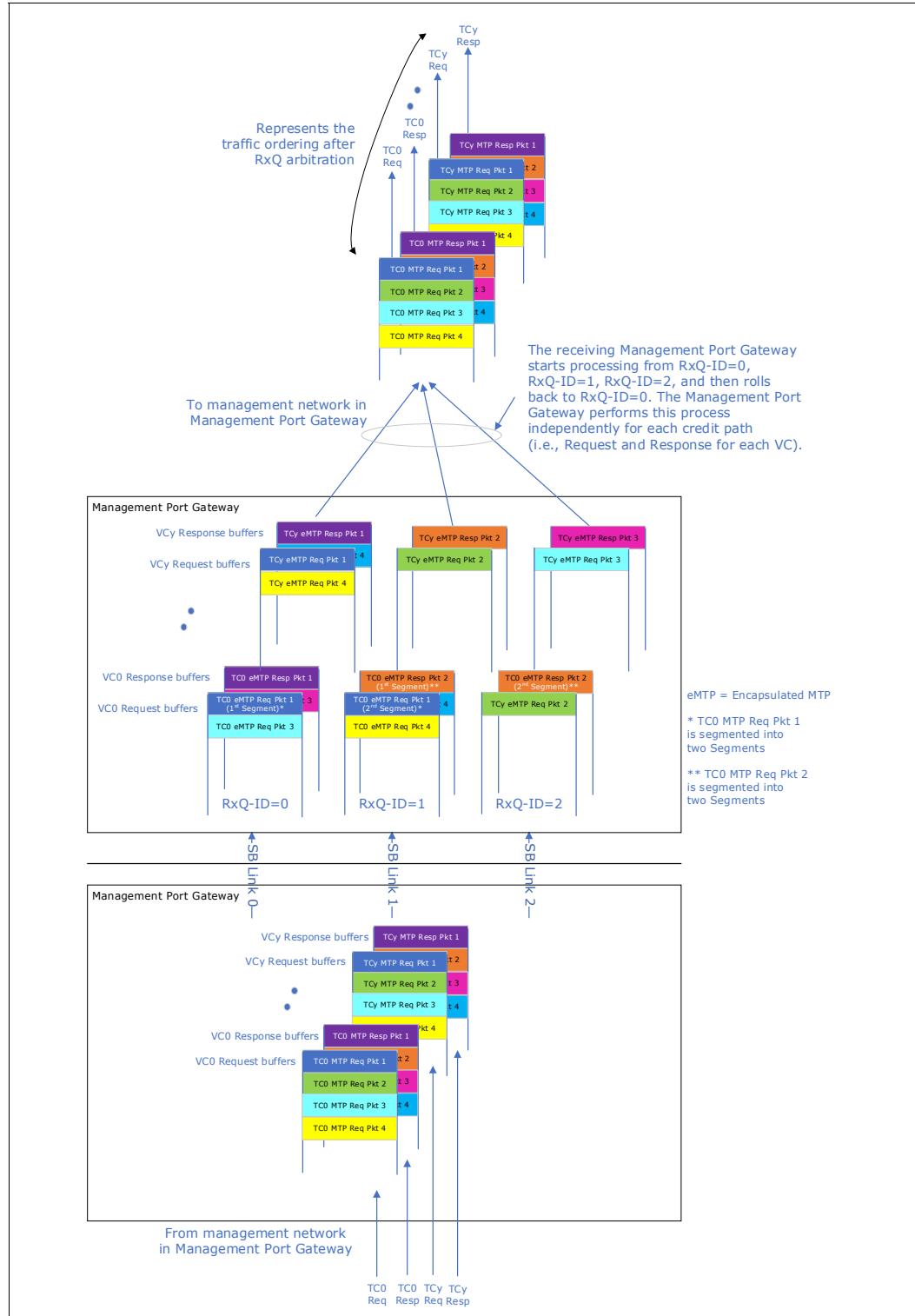
- When the MTP is not segmented, the MTP is fully transmitted to the associated credit buffers and this could take multiple Encapsulated MTPs. In that scenario, each Encapsulated MTP carries the same MPM header but with the length field adjusted for the data length in that message. cr_ret_* fields are also refreshed in every Encapsulated MTP (on the sideband) and indicate 0 if there is no new credit to return. On the mainband path, credits can be refreshed every management flit.
- RxQ-ID is incremented by 1 for transmitting the next MTP of the same VCy:Respz credit type (i.e., the next MTP of VCy:Respz credit type is sent to RxQ-ID1:VCy:Respz credit buffers).
- When the MTP is segmented, a single Encapsulated MTP belonging to the MTP is transmitted to the associated buffers with the “s” bit set to 1. RxQ-ID is incremented by 1 (with wraparound as indicated later in this section) for transmitting each subsequent segment of the same MTP until the MTP is fully sent. After the MTP is fully sent, the RxQ-ID is incremented by 1 again (with wraparound as indicated later in this section) for transmitting the next MTP of the same VCy:Respz credit type.
- The above scheme is repeated independently for traffic within each VCy:Respz credit type. Transmission of packets on different VCy:Respz queues have no dependencies between them.
- RxQ-ID value wraps around after the maximum-negotiated RxQ-ID.
- Transmission to multiple RxQ-ID buffers can occur in parallel on sideband links or mainband stacks.

8.2.4.3.2 Receiver Rules

- On the Rx side, after a Management path setup, the Management Port Gateway services a full MTP (or in the case of Segmentation, one Encapsulated MTP of a MTP) on RxQ-ID0:VCy:Respz queue for a given VCy:Respz credit type. Note that receiving a full MTP could take multiple Encapsulated MTPs.
 - Gateway then services the next MTP (or in case on Segmentation, the next Encapsulated MTP of the management packet) on the RxQ-ID1:VCy:Respz queue, and then on the RxQ-ID2:VCy:Respz queue (if supported), etc.
 - In case of segmentation, the receiver can look at the “s” bit being cleared to 0 (from being set to 1 in prior segments) to know the last segment of an MTP.
 - RxQ-ID value wraps around after the maximum-negotiated RxQ-ID.
- The above receiver scheme applies independently for each VCy:Respz credit type and there are no dependencies between them.
- Messages that do not consume credits must not be allocated into the credited Rx queues (credit returns, PM wake/ack/sleep messages) — and are unconditionally consumed by the Receiver.

Figure 8-42 illustrates the ordering mechanism for an example scenario with three RXQ-IDs, and y VCs (where y=0-7) on the sideband. For the purposes of this illustration — TC0 management port traffic is mapped to VC0 on the sideband management path. TCy management port traffic is mapped to VCy on the sideband management path. Note that in the figure, TC0 Req Pkt 1 and TC0 Resp Pkt 2 are segmented to two segments, to show the impact of segmentation on interleaving and ordering. Other MTPs are not segmented. Similar ordering applies for packets that are interleaved over multiple stacks on the mainband.

Vendor-defined Management Port Gateway messages also use the same credited buffers as MTPs. Transmitter and receiver interleaving rules for these messages are the same as discussed earlier for Encapsulated MTPs.

Figure 8-42. Conceptual Illustration of Sideband Multi-module Ordering with Three RxQs

8.2.4.4 ‘Init Done’ Timeout Flow

During the Management Transport Initialization Phase, a 16-ms timeout (also referred to as ‘Init Done’ timeout) is applied for receiving an “Init Done” MPM from the start of initialization, across all available RxQ-IDs. If an ‘Init Done’ timeout occurs:

- Management Port Gateway cannot schedule any new MPMs across any RxQ-ID and the MPG silently discards any MPMs received, and resets all the RxQ-ID credit counters and pointers.
- Management Port Gateway indicates this status to management FW by way of the management port capability structure (see [Section 8.1.3.6.2.1](#)) and waits for SW to retrigger management path retraining.

8.2.5 Other Management Transport Details

8.2.5.1 Sideband

8.2.5.1.1 Management Port Gateway Flow Control over RD1

See [Section 7.1.3.1](#) for details.

8.2.5.1.2 MPMs with Data Length Rules

When supporting MPMs with Data (see [Section 7.1.2.4](#)) over the sideband, to prevent these messages from occupying the sideband interface for extended periods of time (and thus blocking its usage for mainband link management packets), the following rules must be observed:

- An MPM with Data (e.g., Encapsulated MTP) can have a maximum length field value of seven QWORDS
- Receivers must not check for violation of this transmit rule.
- If the original MTP was larger than seven QWORDS, multiple Encapsulated MTPs are sent until the full MTP is transmitted. It is also permitted to send Encapsulated MTPs smaller than seven QWORDS even when the original MTP is larger than seven QWORDS. This can occur because of credit availability for transmitting the Encapsulated MTP.
- The above rules allow for the link to be arbitrated for any pending Link management packet OR any pending higher priority MEM packet of a different VC:Resp credit type (waiting behind an MPM with Data that is in transmission) with an upper bound on the delay to transmit them. An example of a higher-priority MPM packet that needs to be serviced in a time-bound fashion is a TC1 MTP (see [Section 8.1.3.1.1](#)).
- Segmentation, when performed, must follow the rules described above for each individual Segment of the MTP. See [Section 8.2.4.2](#) for description of segmentation.

[Figure 8-43](#) provides a pictorial representation of splitting a large MTP into multiple smaller Encapsulated MTPs (based on the length rules stated above) and how the Encapsulated MTPs are sent on the sideband link. If the MTP is also segmented, then each Encapsulated MTP is sent on a different RxQ-ID. See [Section 8.2.4.2](#) and [Section 8.2.4.3](#).

See [Section 4.8](#) for how the PHY arbitrates between MPMs and Link Management packets.

Figure 8-43. Example Illustration of a Large MTP Split into Multiple Smaller Encapsulated-MTPs for Transport over Sideband, without Segmentation^a

Management Transport Packet (MTP)		SB Encapsulated MTP 0 — This goes on RxQ-ID=x	
QWORD 0	MTP Header	QWORD 0	MPM Header (s = 1, length = 6h)
QWORD 1	MTP Data 0	QWORD 1	MTP Header
QWORD 2	MTP Data 1	QWORD 2	MTP Data 0
QWORD 3	MTP Data 2	QWORD 3	MTP Data 1
QWORD 4	MTP Data 3	QWORD 4	MTP Data 2
QWORD 5	MTP Data 4	QWORD 5	MTP Data 3
QWORD 6	MTP Data 5	QWORD 6	MTP Data 4
QWORD 7	MTP Data 6	QWORD 7	MTP Data 5
QWORD 8	MTP Data 7		
QWORD 9	MTP Data 8		
QWORD 10	MTP Data 9		
QWORD 11	MTP Data 10		
QWORD 12	MTP Data 11		
QWORD 13	MTP Data 12		
QWORD 14	MTP Data 13		
QWORD 15	MTP Data 14		

SB Encapsulated MTP 1 — This goes on RxQ-ID=x			
QWORD 8	MPM Header (s = 1, length = 6h)		
QWORD 9	MTP Data 6		
QWORD 10	MTP Data 7		
QWORD 11	MTP Data 8		
QWORD 12	MTP Data 9		
QWORD 13	MTP Data 10		
QWORD 14	MTP Data 11		
QWORD 15	MTP Data 12		

SB Encapsulated MTP 2 — This goes on RxQ-ID=x			
QWORD 0	MPM Header (s = 0, length = 1h)		
QWORD 1	MTP Data 13		
QWORD 2	MTP Data 14		

a. N = Number of RxQ-IDs negotiated.
x = Start value of RxQ-ID for an MTP.

8.2.5.1.3 Sideband Runtime Management Transport Path Monitoring — Heartbeat Mechanism

After the management transport path Initialization Phase completes, receiver starts an 8-ms ‘Heartbeat’ timer that restarts whenever an MPM (i.e., opcode 10111b or 11000b) is received. Implementations are permitted to implement this timer as a global timer across all RxQ-IDs or as a timer per RxQ-ID. If the timer times out, the Management Port Gateway de-asserts the `mp_mgmt_up` signal which in turn clears the SB_MGMT_UP flag in the PHY and de-asserts the `mp_mgmt_port_gateway_ready` signal. After a Heartbeat timeout, Management Port Gateway functions similar to what occurs during a ‘Init Done timeout’ (see [Section 8.2.4.4](#) for details). The Heartbeat timer stops after L1/L2 entry negotiation on the sideband path successfully completes, and restarts when L1/L2 exit negotiation starts. See [Section 8.2.5.1.4](#) for details of Management path PM entry/exit flows.

After the ‘Init Done’ message is been transmitted on an RxQ-ID path during the initialization Phase, the Management Port Gateway (MPG) must guarantee an MPM transmission of no more than 4 ms apart on the RxQ-ID path. If there are no scheduled messages to send on an RxQ-ID path, the MPG must send a credit return message (unless there was a Heartbeat timeout on the receiver side as stated in the previous paragraph) with VC value set to VC0, Resp value set to 0 and cr_ret value set

to 0h. Note that the latter applies even if the MPG takes longer than 8 ms to exit L1/L2 before the MPG sends the associated PM exit message.

If a control parity error is detected on any received MPM, the Management Port Gateway invokes the ‘Heartbeat timeout’ flow.

8.2.5.1.4 Sideband Management Path Power Management Rules

On the sideband interface, it is expected that there is higher-level firmware/software managing the deeper power states of Management Port Gateways on both sides. The sleep and wake req/ack/nak messages (see [Figure 7-12](#)) are provided to negotiate shutdown/wake of the management transport path for deep power states in which the Management Port Gateway logic can be clock gated or powered down (as coordinated by the higher-level firmware). It is especially useful for a low-power chiplet and/or SiP states flows to take advantage of these handshakes and coordinate entry and wake up of the Management Transport Path. These messages and negotiation must occur independently for each RxQ-ID path, and each direction. While not in a PM state, the Management Port Gateway must keep the `mp_wake_req` signal asserted and this informs the Physical Layer adapter to keep the logic up and running.

8.2.5.1.4.1 Sideband PM Entry Rules

- Management Port Gateway Transmitter that initiates the PM entry ensures that no other packets will be transmitted (other than credit returns and PM messages) on any of the enabled RxQ-ID paths.
- Following the above, the Transmitter sends a “Sleep req” message on each of the RxQ-ID paths. After a “Sleep req” message is sent on an RxQ-ID path, only credit return messages can be transmitted on the path until a “Sleep ack” message is received on the path or a Sleep ack timeout occurs (see last bullet in this section below). If the former scenario, additional message transmissions are not permitted until the subsequent PM exit. In the latter scenario, message transmission can resume soon after the timeout.
- After receiving a “Sleep req” message, the receiving Management Port Gateway must ensure that the corresponding Rx buffer is empty, and that all pending credit returns have been sent to the remote Link partner. After these conditions are met, a “Sleep ack” message is scheduled.
- If a chiplet responded with a “Sleep ack” message, the chiplet must send a “Sleep req” message (if not already sent) within 16 ms of sending the “Sleep ack” and receive a response to complete the flow; otherwise, sleep entry is aborted.
- After a Management Port Gateway has sent and received a “Sleep Ack” message on all paths, the MPG is permitted to clock gate or power down, etc. The Management Port Gateway must de-assert the `mp_wake_req` signal before entering the clock gated or power down state.
- If Sleep Nak was sent or received, the sleep entry is aborted.
- Transmitter of a “Sleep req” message can wait for an implementation-dependent timeout to receive a “Sleep ack” before aborting the flow. In multi-module implementations, the “Sleep ack” message must be received across all negotiated RxQ-ID paths before the timeout.

8.2.5.1.4.2 Sideband PM Exit Rules

- Management Port Gateway Transmitter that initiates the exit performs the `mp_wake_req/ack` handshake with its Physical Layer and schedules the “Wake req” message on each RxQ-ID path.
- Partner chiplet’s Physical Layer that receives the “Wake req” message over the sideband wakes up its Management Port Gateway by performing the `pm_clk_req/ack` handshake before transmitting the “Wake req” message in response.

- After the partner chiplet's Management Port Gateway receives the "Wake Req" message, that Management Port Gateway must respond with a "Wake ack" message when the MPG is ready to receive credited packets into its Rx buffer. Moreover, the Management Port Gateway must initiate its own "Wake req" message to the remote Link partner if the MPG has not already done so.
- After a "Wake ack" message is sent and received on all negotiated RxQ-IDs, the PM exit flow is complete and regular packet transfer can begin as soon as the last "Wake ack" message is transmitted.

8.2.5.1.5 Management Port Gateway Mux Arbitration

There is no prescribed arbitration mechanism for the Management Port Gateway mux on the Sideband. Additionally, the size of Management Port Gateway Flow control buffers over RDI (see [Section 8.2.5.1.1](#)) is not specified for Management Port Gateway-initiated traffic. Implementations should take care to ensure that the PHY arbitration rules specified in [Section 4.8](#) are not violated.

8.2.5.2 Mainband

8.2.5.2.1 NOP Message

The Management Port Gateway inserts NOP messages whose format is shown in [Figure 8-44](#), in all QWORD locations in a Management flit in which there is no MPM to send. NOP messages can start only at MPM boundaries within a flit.

Figure 8-44. Management Flit NOP Message on Mainband

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0000_0000h																															
0000_0000h																															

8.2.5.2.2 Credit Return DWORD Format

Figure 8-45. Management Transport Credit Return DWORD (CRD) Format on Mainband

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cr_ret_resp_a	cr_ret_vc_a	cr_ret_a										rsvd	cr_ret_resp_b	cr_ret_vc_b	cr_ret_b										rsvd	rxqid					

See [Section 3.3.3](#) and [Section 3.3.4](#) for details on where this DWORD is sent in a Management Flit for various Flit formats.

The following rules apply:

- rxqid field in the DWORD applies to both credit returns a and b shown in [Figure 8-45](#)
- During VC initialization, on the Management Flit that carries the Management Port Gateway Capability message, all credit return fields must be set to 0
- If there is no credit to return in credit return slots a or b (as shown in [Figure 8-45](#)), a value of 0 is used for all associated credit return fields
- If credit returns a and b carry the same vc:resp fields, then the total credit returned for that rxqid:vc:resp credit type is the sum of cr_ret_a and cr_ret_b

8.2.5.2.3 Management Flit Formats

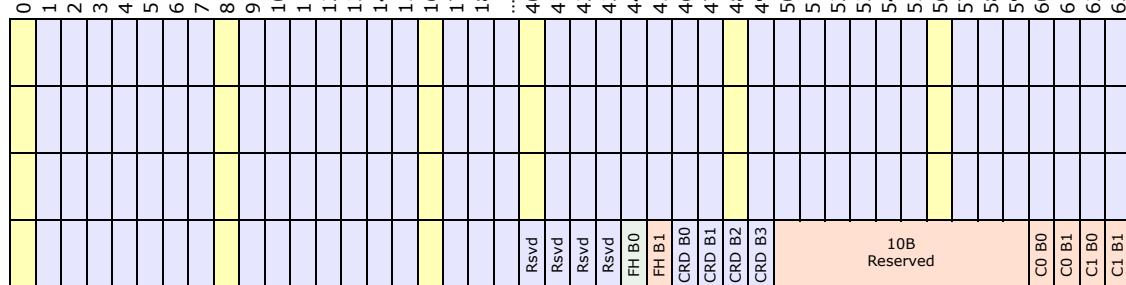
On the mainband, MPMs are supported only over Flit *Format 3* through *Format 6*.

See [Section 3.3.3](#) and [Section 3.3.4](#) for a D2D view of the Management Protocol mapping over Flit *Format 3* through *Format 6*. If Flit *Format 1* and *Format 2* are negotiated, the Management Protocol on that stack is disabled (if supported). Management flits have bits [7:6] of Byte 1 set to 10b. See [Section 8.2.2.2](#) for packet format of MPMs over the mainband. Mapping of these MPMs over Flit *Format 3* through *Format 6* is as follows:

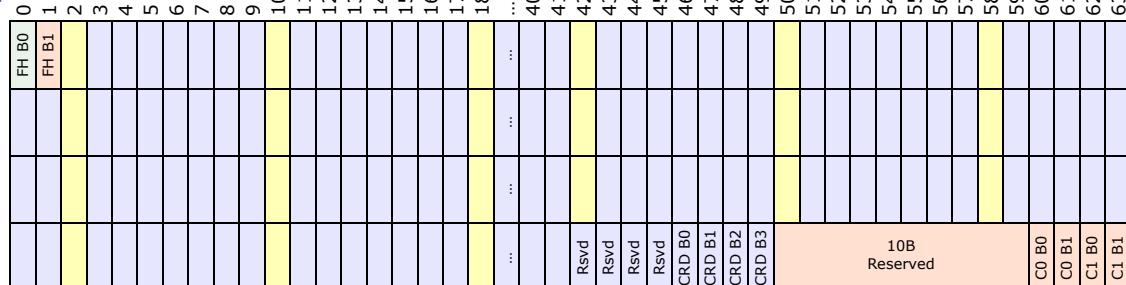
- MPM header and each QWORD of MPM payload (when applicable) can be placed only at specified byte locations in the Management flit, and can start at the 1st byte in the Management flit in which “all bits are populated by protocol layer” (see [Figure 2-1](#) for reference), and at subsequent 8B increments within the flit. While incrementing, only bytes in which “all bits are populated by the Protocol Layer” are considered, excluding CRD byte locations and bytes marked as rsvd for Protocol Layer (e.g., Flit *Format 3*, Bytes 40 through 43). This is pictorially shown in [Figure 8-46](#).

Figure 8-46. Valid MPM Header Start Locations for Various Flit Formats^{a b}

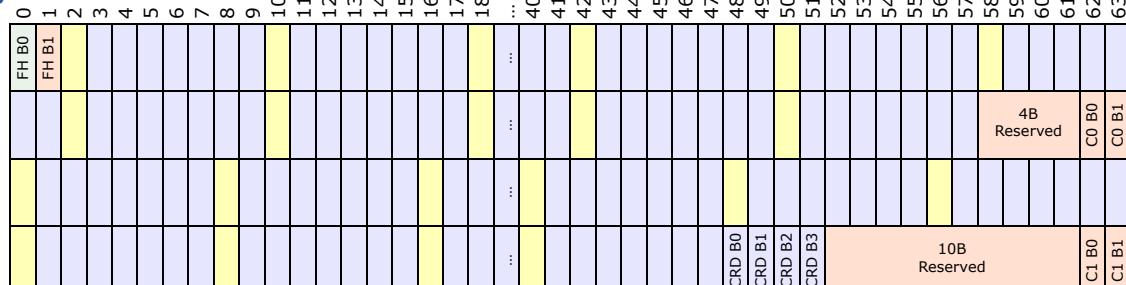
Format 3



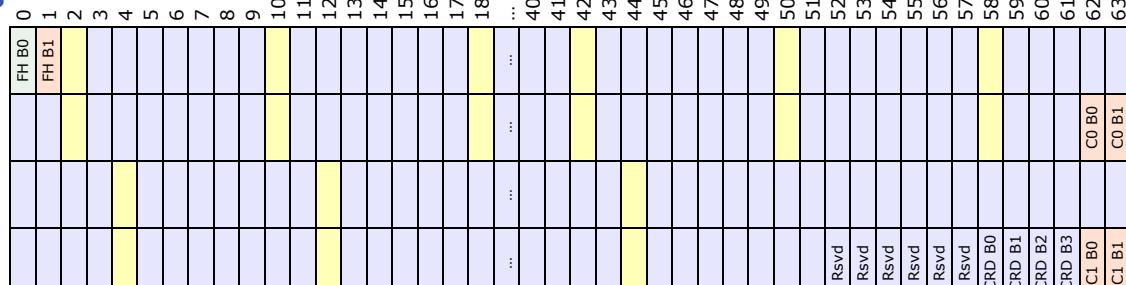
Format 4



Format 5



Format 6



- a. Yellow cells indicate a Valid Management Port Message (MPM) header or Payload QWORD start location. For the other colors, see [Figure 2-1](#) for color mapping.
 - b. B = Byte, C = CRC, CRD = Credit Return DWORD, FH = Flit Header, Rsvd = Reserved. All are numbered, as appropriate, except for Rsvd.

Starting at a valid MPM header byte location (as discussed above), Byte 0 of the first DWORD of the MPM header is sent at that byte, followed by Byte 1 of the first DWORD of the header at starting byte location+1 until Byte 3 of the 2nd DWORD of the header. This is followed by Byte 0 of the 1st DWORD of the MPM payload (if one exists), followed by Byte 1, Byte 2, Byte 3, etc., placed at incrementing byte locations. Non-CRD bytes, bytes that are not marked as reserved and those that are driven by the protocol layer are contiguously packed with MPM bytes after an MPM transmission starts and until the transmission ends. If an MPM cannot be fully transmitted within a Management Flit, the MPM continues in the subsequent Management Flit of the same stack. NOP message(s) (see [Section 8.2.5.2.1](#)) can be inserted between MPMs within a Management Flit. It is also valid to send a Management Flit with all NOP messages in the protocol layer-driven non-CRD bytes and non-reserved byte locations. CRD bytes in a Management Flit always carry the credit return information per the rules stated in [Section 8.2.5.2.2](#).

[Figure 8-47](#) and [Figure 8-48](#) show example mappings of three MPMs inside Flits of *Format 3* and *Format 5*, respectively. The 1st MPM is of an MPM with Data type with a payload size of 15 QWORDs. The 2nd MPM is also of an MPM with Data type with a payload size of 6 QWORDs. The 3rd MPM is an MPM with a payload of size of 1 QWORD. NOPs are inserted after the end of the 3rd MPM until the end of the flit.

Figure 8-47. Example Mapping of MPMs and NOPs in Flit of Format 3^{a b}

MH D0B0	MP D28B0	MP D13B2	FH B0	0	MP D0B0	MH D0B0	MP D14B0	MH D0B0	MH D0B1	MP D14B1	MH D0B1	MH D0B1	1	
MH D0B1	MP D28B1	MP D13B3	FH B1	1	MP D0B1	MH D0B1	MP D14B1	MH D0B1	MH D0B2	MP D14B2	MH D0B2	MH D0B2	2	
MH D0B2	MP D28B2	MP D28B2	MH D0B0	2	MP D0B2	MH D0B2	MP D14B2	MH D0B2	MH D0B3	MP D14B3	MH D0B3	MH D0B3	3	
MH D0B3	MP D28B3	MP D28B3	MH D0B1	3	MP D0B3	MH D0B3	MP D14B3	MH D0B3	MH D1B0	MP D1B0	MH D1B0	MH D1B0	4	
MH D1B0	MP D29B0	MP D29B0	MH D0B2	4	MP D1B0	MH D1B0	MP D1B0	MH D1B0	MH D1B1	MP D1B1	MH D1B1	MH D1B1	5	
MH D1B1	MP D29B1	MP D29B1	MH D0B3	5	MP D1B1	MH D1B1	MP D1B1	MH D1B1	MH D1B2	MP D1B2	MH D1B2	MH D1B2	6	
MH D1B2	MP D29B2	MP D29B2	MH D1B0	6	MP D1B2	MH D1B2	MP D1B2	MH D1B2	MH D1B3	MP D1B3	MH D1B3	MH D1B3	7	
MH D1B3	MP D29B3	MP D29B3	MH D1B1	7	MP D1B3	MH D1B3	MP D1B3	MH D1B3	NOP	MP D0B0	MP D0B0	MP D0B0	8	
MH D1B4	MP D0B0	MP D0B0	MH D1B2	8	NOP	MP D0B1	NOP	MP D0B1	9					
MH D1B5	MP D0B1	MP D0B1	MH D1B3	9	NOP	MP D0B2	NOP	MP D0B2	10					
MH D1B6	MP D0B2	MP D0B2	MH D1B0	10	MP D0B2	MP D0B1	MP D0B1	MP D0B1	MP D0B3	11				
MH D1B7	MP D0B3	MP D0B3	MH D1B1	11	MP D0B3	MP D0B2	MP D0B2	MP D0B2	MP D1B0	12				
MH D1B8	MP D1B0	MP D1B0	MH D1B2	12	MP D1B0	MP D1B1	13							
MH D1B9	MP D1B1	MP D1B1	MH D1B3	13	MP D1B1	MP D1B2	MP D1B2	MP D1B2	MP D1B3	14				
MH D1B10	MP D1B2	MP D1B2	MH D1B0	14	MP D1B2	MP D1B1	MP D1B1	MP D1B1	MP D1B3	15				
MH D1B11	MP D1B3	MP D1B3	MH D1B1	15	MP D1B3	MP D1B2	MP D1B2	MP D1B2	MP D1B3	16				
MH D1B12	MP D1B4	MP D1B4	MH D1B2	16	MP D1B4	MP D1B3	MP D1B3	MP D1B3	MP D1B4	17				
MH D1B13	MP D1B5	MP D1B5	MH D1B3	17	MP D1B5	MP D1B4	MP D1B4	MP D1B4	MP D1B5	18				
MH D1B14	MP D1B6	MP D1B6	MH D1B0	19	MP D1B6	MP D1B5	MP D1B5	MP D1B5	MP D1B6	19				
MH D1B15	MP D1B7	MP D1B7	MH D1B1	20	MP D1B7	MP D1B6	MP D1B6	MP D1B6	MP D1B7	20				
MH D1B16	MP D1B8	MP D1B8	MH D1B2	21	MP D1B8	MP D1B7	MP D1B7	MP D1B7	MP D1B8	21				
MH D1B17	MP D1B9	MP D1B9	MH D1B3	22	MP D1B9	MP D1B8	MP D1B8	MP D1B8	MP D1B9	22				
MH D1B18	MP D1B10	MP D1B10	MH D1B0	23	MP D1B10	MP D1B9	MP D1B9	MP D1B9	MP D1B10	23				
MH D1B19	MP D1B11	MP D1B11	MH D1B1	24	MP D1B11	MP D1B10	MP D1B10	MP D1B10	MP D1B11	24				
MH D1B20	MP D1B12	MP D1B12	MH D1B2	25	MP D1B12	MP D1B11	MP D1B11	MP D1B11	MP D1B12	25				
MH D1B21	MP D1B13	MP D1B13	MH D1B3	26	MP D1B13	MP D1B12	MP D1B12	MP D1B12	MP D1B13	26				
MH D1B22	MP D1B14	MP D1B14	MH D1B0	27	MP D1B14	MP D1B13	MP D1B13	MP D1B13	MP D1B14	27				
MH D1B23	MP D1B15	MP D1B15	MH D1B1	28	MP D1B15	MP D1B14	MP D1B14	MP D1B14	MP D1B15	28				
MH D1B24	MP D1B16	MP D1B16	MH D1B2	29	MP D1B16	MP D1B15	MP D1B15	MP D1B15	MP D1B16	29				
MH D1B25	MP D1B17	MP D1B17	MH D1B3	30	MP D1B17	MP D1B16	MP D1B16	MP D1B16	MP D1B17	30				
MH D1B26	MP D1B18	MP D1B18	MH D1B0	31	MP D1B18	MP D1B17	MP D1B17	MP D1B17	MP D1B18	31				
MH D1B27	MP D1B19	MP D1B19	MH D1B1	32	MP D1B19	MP D1B18	MP D1B18	MP D1B18	MP D1B19	32				
MH D1B28	MP D1B20	MP D1B20	MH D1B2	33	MP D1B20	MP D1B19	MP D1B19	MP D1B19	MP D1B20	33				
MH D1B29	MP D1B21	MP D1B21	MH D1B3	34	MP D1B21	MP D1B20	MP D1B20	MP D1B20	MP D1B21	34				
MH D1B30	MP D1B22	MP D1B22	MH D1B0	35	MP D1B22	MP D1B21	MP D1B21	MP D1B21	MP D1B22	35				
MH D1B31	MP D1B23	MP D1B23	MH D1B1	36	MP D1B23	MP D1B22	MP D1B22	MP D1B22	MP D1B23	36				
MH D1B32	MP D1B24	MP D1B24	MH D1B2	37	MP D1B24	MP D1B23	MP D1B23	MP D1B23	MP D1B24	37				
MH D1B33	MP D1B25	MP D1B25	MH D1B3	38	MP D1B25	MP D1B24	MP D1B24	MP D1B24	MP D1B25	38				
MH D1B34	MP D1B26	MP D1B26	MH D1B0	39	MP D1B26	MP D1B25	MP D1B25	MP D1B25	MP D1B26	39				
MH D1B35	MP D1B27	MP D1B27	MH D1B1	40	MP D1B27	MP D1B26	MP D1B26	MP D1B26	MP D1B27	40				
MH D1B36	MP D1B28	MP D1B28	MH D1B2	41	MP D1B28	MP D1B27	MP D1B27	MP D1B27	MP D1B28	41				
MH D1B37	MP D1B29	MP D1B29	MH D1B3	42	MP D1B29	MP D1B28	MP D1B28	MP D1B28	MP D1B29	42				
MH D1B38	MP D1B30	MP D1B30	MH D1B0	43	MP D1B30	MP D1B29	MP D1B29	MP D1B29	MP D1B30	43				
MH D1B39	MP D1B31	MP D1B31	MH D1B1	44	MP D1B31	MP D1B30	MP D1B30	MP D1B30	MP D1B31	44				
MH D1B40	MP D1B32	MP D1B32	MH D1B2	45	MP D1B32	MP D1B31	MP D1B31	MP D1B31	MP D1B32	45				
MH D1B41	MP D1B33	MP D1B33	MH D1B3	46	MP D1B33	MP D1B32	MP D1B32	MP D1B32	MP D1B33	46				
MH D1B42	MP D1B34	MP D1B34	MH D1B0	47	MP D1B34	MP D1B33	MP D1B33	MP D1B33	MP D1B34	47				
MH D1B43	MP D1B35	MP D1B35	MH D1B1	48	MP D1B35	MP D1B34	MP D1B34	MP D1B34	MP D1B35	48				
MH D1B44	MP D1B36	MP D1B36	MH D1B2	49	MP D1B36	MP D1B35	MP D1B35	MP D1B35	MP D1B36	49				
MH D1B45	MP D1B37	MP D1B37	MH D1B3	50	MP D1B37	MP D1B36	MP D1B36	MP D1B36	MP D1B37	50				
MH D1B46	MP D1B38	MP D1B38	MH D1B0	51	MP D1B38	MP D1B37	MP D1B37	MP D1B37	MP D1B38	51				
MH D1B47	MP D1B39	MP D1B39	MH D1B1	52	MP D1B39	MP D1B38	MP D1B38	MP D1B38	MP D1B39	52				
MH D1B48	MP D1B40	MP D1B40	MH D1B2	53	MP D1B40	MP D1B39	MP D1B39	MP D1B39	MP D1B40	53				
MH D1B49	MP D1B41	MP D1B41	MH D1B3	54	MP D1B41	MP D1B40	MP D1B40	MP D1B40	MP D1B41	54				
MH D1B50	MP D1B42	MP D1B42	MH D1B0	55	MP D1B42	MP D1B41	MP D1B41	MP D1B41	MP D1B42	55				
MH D1B51	MP D1B43	MP D1B43	MH D1B1	56	MP D1B43	MP D1B42	MP D1B42	MP D1B42	MP D1B43	56				
MH D1B52	MP D1B44	MP D1B44	MH D1B2	57	MP D1B44	MP D1B43	MP D1B43	MP D1B43	MP D1B44	57				
MH D1B53	MP D1B45	MP D1B45	MH D1B3	58	MP D1B45	MP D1B44	MP D1B44	MP D1B44	MP D1B45	58				
MH D1B54	MP D1B46	MP D1B46	MH D1B0	59	MP D1B46	MP D1B45	MP D1B45	MP D1B45	MP D1B46	59				
MH D1B55	MP D1B47	MP D1B47	MH D1B1	60	MP D1B47	MP D1B46	MP D1B46	MP D1B46	MP D1B47	60				
MH D1B56	MP D1B48	MP D1B48	MH D1B2	61	MP D1B48	MP D1B47	MP D1B47	MP D1B47	MP D1B48	61				
MH D1B57	MP D1B49	MP D1B49	MH D1B3	62	MP D1B49	MP D1B48	MP D1B48	MP D1B48	MP D1B49	62				
MH D1B58	MP D1B50	MP D1B50	MH D1B0	63	MP D1B50	MP D1B49	MP D1B49	MP D1B49	MP D1B50	63				
MH D1B59	MP D1B51	MP D1B51	MH D1B1	64	MP D1B51	MP D1B50	MP D1B50	MP D1B50	MP D1B51	64				

- a. See [Figure 2-1](#) for color mapping.
- b. B = Byte, C = CRC, CRD = Credit Return DWORD, D = DWORD, FH = Flit Header, MH = MPM Header, MP = MPM Payload, NOP = No Operation.

[Figure 8-49](#) shows an example mapping of four MPMs inside a Format 3 flit. The 3rd MPM rolls over into the 2nd flit. The 1st MPM in this example is of an MPM with Data type with a payload size of 15 QWORDs. The 2nd MPM is also of an MPM with Data type with a payload size of 6 QWORDs. The 3rd MPM is an MPM with payload size of 6 QWORDs, where the 6th QWORD is sent in the 2nd Flit. The 4th MPM in this example is a 1-QWORD Vendor-defined Management Port Gateway message without data. The remainder of the 2nd flit is all NOPs.

Figure 8-49. Example MPM Mapping to Management Flit for Format 3 with MPM Rollover to Next Flit^a ^b

Flit 0		Flit 1																
NOP	NOP	MP D10B0		MH D0B0		MP D14B0		MH D0B0		MP D14B1		MH D0B1		0		1		
		MP D10B1	0	MP D0B1	0	MP D14B1	0	MP D14B1	0	MP D14B1	0	MP D14B1	0	MH D0B2	2	MH D0B2	2	
		MP D10B1	1	MP D0B1	1	MP D14B1	1	MP D14B1	1	MP D14B1	1	MP D14B1	1	MH D0B3	3	MH D0B3	3	
		MP D10B2	2	MP D0B2	2	MP D14B2	2	MP D14B2	2	MP D14B2	2	MP D14B2	2	MH D1B0	4	MH D1B0	4	
		MP D10B3	3	MP D0B3	3	MP D14B3	3	MP D14B3	3	MP D14B3	3	MP D14B3	3	MH D1B1	5	MH D1B1	5	
		MP D11B0	4	MP D1B0	4	MP D1B0	4	MP D1B0	4	MP D1B0	4	MP D1B0	4	MH D1B2	6	MH D1B2	6	
		MP D11B1	5	MP D1B1	5	MP D1B1	5	MP D1B1	5	MP D1B1	5	MP D1B1	5	MH D1B3	7	MH D1B3	7	
		MP D11B2	6	MP D1B2	6	MP D1B2	6	MP D1B2	6	MP D1B2	6	MP D1B2	6	NP D0B0	8	NP D0B0	8	
		MP D11B3	7	MP D1B3	7	MP D1B3	7	MP D1B3	7	MP D1B3	7	MP D1B3	7	NP D0B1	9	NP D0B1	9	
		MP D10B0	8	MP D0B0	8	MP D1B0	8	MP D1B0	8	MP D1B0	8	MP D1B0	8	MH D0B2	10	MH D0B2	10	
		MP D0B1	9	MP D0B1	9	MP D0B1	9	MP D0B1	9	MP D0B1	9	MP D0B1	9	MH D0B3	11	MH D0B3	11	
		MP D0B2	10	MP D0B2	10	MP D0B2	10	MP D0B2	10	MP D0B2	10	MP D0B2	10	NP D1B0	12	NP D1B0	12	
		MP D0B3	11	MP D0B3	11	MP D0B3	11	MP D0B3	11	MP D0B3	11	MP D0B3	11	MH D1B1	13	MH D1B1	13	
		MP D1B0	12	MP D1B0	12	MP D1B0	12	MP D1B0	12	MP D1B0	12	MP D1B0	12	MP D1B2	14	MP D1B2	14	
		MP D1B1	13	MP D1B1	13	MP D1B1	13	MP D1B1	13	MP D1B1	13	MP D1B1	13	MP D1B3	15	MP D1B3	15	
		MP D1B2	14	MP D1B2	14	MP D1B2	14	MP D1B2	14	MP D1B2	14	MP D1B2	14	MP D1B3	15	MP D1B3	15	
		MP D1B3	15	MP D1B3	15	MP D1B3	15	MP D1B3	15	MP D1B3	15	MP D1B3	15	MP D1B3	15	MP D1B3	15	
		NOP	16	...	17	...	18	...	MP D0B3	...	MP D0B3	...	MP D0B3	
									Rsvd	40	Rsvd	41	Rsvd	42	Rsvd	43	Rsvd	44
									Rsvd	41	Rsvd	42	Rsvd	43	Rsvd	44	Rsvd	45
									Rsvd	42	Rsvd	43	Rsvd	44	Rsvd	45	Rsvd	46
									Rsvd	43	Rsvd	44	Rsvd	45	Rsvd	46	Rsvd	47
									FH B0	44	FH B0	45	FH B1	46	CRD B0	47	CRD B1	48
									CRD B0	46	CRD B0	47	CRD B1	48	CRD B2	49	CRD B3	50
									CRD B1	47	CRD B1	48	CRD B2	49	CRD B3	50	CRD B3	51
									CRD B2	48	CRD B2	49	CRD B3	51	CRD B3	52	CRD B3	53
									CRD B3	49	CRD B3	50	CRD B3	51	CRD B3	52	CRD B3	53
									
									10B Reserved	54	10B Reserved	55	10B Reserved	56	10B Reserved	57	10B Reserved	58
									MH D1B0	55	MH D1B0	56	MH D1B0	57	MH D1B0	58	MH D1B0	59
									MH D1B1	55	MH D1B1	56	MH D1B1	57	MH D1B1	58	MH D1B1	59
									MH D1B2	55	MH D1B2	56	MH D1B2	57	MH D1B2	58	MH D1B2	59
									MH D1B3	55	MH D1B3	56	MH D1B3	57	MH D1B3	58	MH D1B3	59
									MP D30B1	55	MP D30B1	56	MP D30B1	57	MP D30B1	58	MP D30B1	59
									MP D11B3	55	MP D11B3	56	MP D11B3	57	MP D11B3	58	MP D11B3	59
									MH D0B0	55	MH D0B0	56	MH D0B0	57	MH D0B0	58	MH D0B0	59
									C0 E0	55	C0 E0	56	C0 E0	57	C0 E0	58	C0 E0	59
									C0 E1	55	C0 E1	56	C0 E1	57	C0 E1	58	C0 E1	59
									C1 E0	55	C1 E0	56	C1 E0	57	C1 E0	58	C1 E0	59
									C1 B1	55	C1 B1	56	C1 B1	57	C1 B1	58	C1 B1	59
									NOP	55	NOP	56	NOP	57	NOP	58	NOP	59

- a. See Figure 2-1 for color mapping.
 - b. B = Byte, C = CRC, CRD = Credit Return DWORD, D = DWORD, FH = Flit Header, MH = MPM Header, MP = MPM Payload, NOP = No Operation, Rsvd = Reserved.

8.2.5.2.4 L1/L2 Link States and Management Transport

See Section 3.6 for details.

8.2.5.2.5 Link Reset/Link Disable and Management Transport

For Management Transport on the mainband that has a Management Port Gateway mux, it should be noted that if the associated protocol stack resets or disables the link, the Management Transport path is also reset/disabled. If this is not desired, it is recommended that protocol stacks in such configurations have a way to disable sending link reset and link disable requests on the FDI so that the Management Transport path is not affected.

8.2.6 Retimers and Management Transport

On the sideband, retimers can support management transport if the retimers need to be part of the UCIe management network. This support is optional. If supporting management transport, the retimers must abide by all the requirements stated earlier in this chapter for a generic chiplet implementation. When retimers are part of the management network, the retimers can be fully managed and be able to forward management packets between their UCIe interface and the retimed interface.

On the mainband, retimers can optionally support management transport.

8.3 UCIe Debug and Test Architecture (UDA)

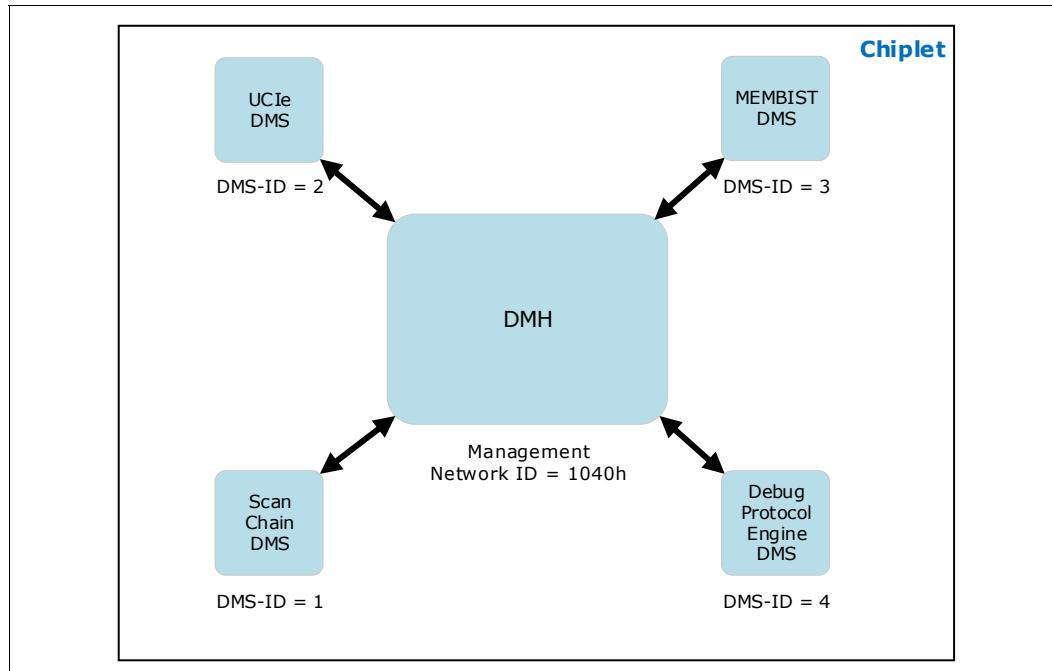
8.3.1 Overview

UCIe Debug and Test (DFx) Architecture (UDA) provides a standardized test and debug infrastructure for UCIe-based chiplets and SiPs to enable standard test and debug methods in a UCIe-based open chiplet ecosystem.

UDA is architected on top of UCIe Manageability Infrastructure and uses the architectural elements of that infrastructure for Chiplet-level and SiP-level testing and debug (see [Section 8.1](#) for details of UCIe Manageability Architecture). UDA requires functional UCIe links (Sideband and/or Mainband) and a functional management network for test and debug purposes. Debug and bring-up of UCIe links and elements that comprise the UDA (see [Section 8.3.1.1](#) through [Section 8.3.1.4](#)) can be performed by any sideband interface of choice (e.g., JTAG, GPIO, Sideband-only UCIe, etc.) of the chiplet vendor, and are beyond the scope of this specification.

Within each chiplet, UDA is architected in a Hub-Spoke model. In this model, DFx Management Hub (DMH) is the Management Element that implements the Debug and Test Protocol(s). UDA allows for SW/FW to discover debug capabilities present in a chiplet, and provides for global security control/status for test/debug functionality present in the chiplet. Chiplet test/debug functionality is implemented in DFx Management Spoke (DMS). Some examples of test/debug functionality are Scan controller, Memory BIST, SoC fabric debug, Core debug, trace protocol engine, etc.

In [Figure 8-50](#), there is one DMH with a Management Network ID of 1040h and 4 DMSs connected to it with DMS-IDs (also referred to as Spoke-IDs) from 1 to 4. Management Network ID is used to route DFx and other relevant manageability packets to DMH in the manageability fabric. See [Section 8.1.3.2](#) for how to interpret this ID. DMS-ID is used to route ID-routed Test and Debug packets to the correct Spoke within DMH.

Figure 8-50. UDA Overview in Each Chiplet – Illustration

8.3.1.1 DFx Management Hub (DMH)

Key points about DMH:

- DMH implements a set of registers that allow Management Firmware to:
 - Enumerate test/debug capabilities within the chiplet
 - Globally access control to/from debug functionality of DMS
 - Reliably report status of test/debug functionality usage within the chiplet
- DMH provides appropriate routing of Manageability packets that target various Spokes under it
- There can be multiple DMHs within a chiplet
- DMH can coexist with other protocol entities under the same Management Network ID
- DMH registers are accessed by the UMAP protocol (see [Section 8.1.4](#) for details)

8.3.1.2 DFx Management Spoke (DMS)

Key points about DMS:

- Spokes provide the required test/debug functionality in a chiplet.
 - Some examples of test/debug functionality that can be implemented in a Spoke are MEMBIST, Scan controller, Core debug, SoC internal debug/test, PCIe link debug, UCIE link debug, Trace protocol, etc.
 - Spoke definition is left to the chiplet vendor to decide based on the chiplet's test/debug requirements.
- Spokes support the UMAP protocol and are discovered by SW as discussed in [Section 8.3.5](#).

- Spokes can optionally support Vendor-defined Test and Debug messages, and these messages are routed to the destination Spoke within a DMH using a DMS-ID.
- Valid DMS-IDs for Spokes are from 1 to 254. A value of 0h is assigned for DMH. A value of FFh is reserved.
- DMS-ID is unique within a given DMH.
- DMH provides a pointer to the first DMS in a linked list of DMSs present within the DMH.
- Each Spoke identifies itself as one of these types (see [Section 8.3.5.3.2.8](#) for more details):
 - UCIE.Physical_Layer
 - UCIE.Adapter
 - UCIE.Adapter_Physical_Layer
 - Vendor-defined
- Each Spoke implements a simple standard register set that helps to uniquely enumerate each Spoke and to allow custom SW to be loaded to interact with the Spoke.
 - All Spokes minimally support DWORD Register Rd/Wr accesses. Support for sizes beyond that are optional.
- Vendor-defined sections of the register space can be used for a vendor to implement any Spoke functionality such as triggering BIST, reading internal debug registers, array dump, etc.

8.3.1.3 Supported Protocols

8.3.1.3.1 UCIE Memory Access Protocol (UMAP)

Used to access registers in DMH/DMS. See [Section 8.1.4](#) for details of this protocol.

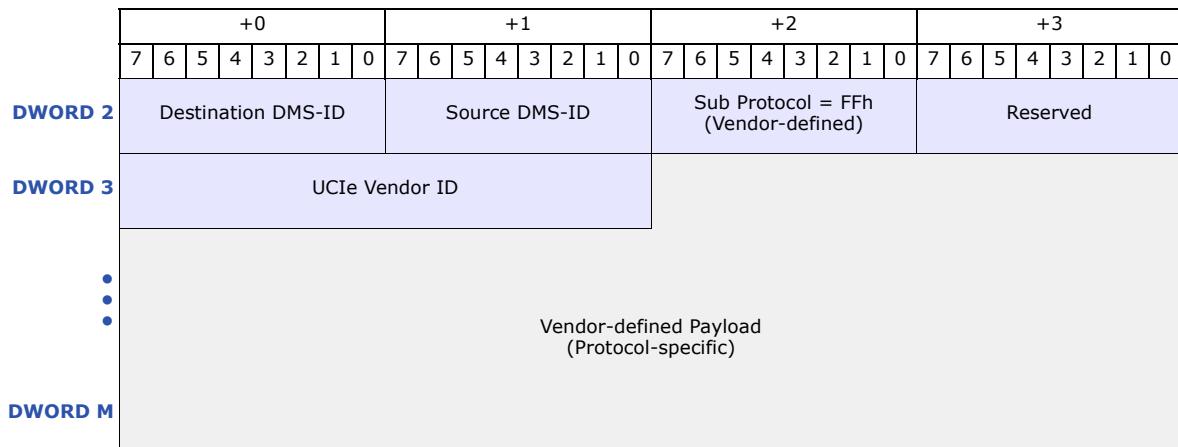
8.3.1.3.2 Vendor-defined Test and Debug Protocol

Used for test as discussed in [Section 8.3.2](#) and [Section 8.3.3](#) and for any other vendor-defined functionality. Format of DWORDS 2 to M of Vendor-defined Test and Debug UCIE DFx Messages (or Vendor-defined UDM, for short) is shown below. DWORDs 0 to 1 of these messages follow the standard format of Management Transport packet described in [Section 8.1.3](#), with the Management Protocol field set to 'Test and Debug Protocol'. Packet Integrity Protection DWORDs (that appear after DWORD M) are as defined in [Section 8.1](#).

- These messages are routed to the correct Spoke within a DMH, using the Destination DMS-ID field in Byte 8 of the message.
- UCIE Vendor-ID field is the UCIE Consortium-assigned Vendor ID for the Spoke's IP Vendor

In [Figure 8-51](#), UCIe Vendor ID[15:8] is sent on Byte 0[7:0] of DWORD 3, and UCIe Vendor ID[7:0] is sent on Byte 1[7:0] of DWORD 3.

Figure 8-51. Vendor-defined Test and Debug UDM



IMPLEMENTATION NOTE

A Spoke's support for Vendor-defined UDM is negotiated/discovered using vendor-defined mechanisms. The Spoke Vendor ID and Spoke Device ID can be used to determine a specific Spoke implementation from a specific Vendor. Vendor-defined registers in the Spoke can be used to negotiate/discover the Vendor-defined Payload format of Vendor-defined UDM.

8.3.1.4 UDM and UCIe Memory Access Protocol Message Encapsulation over UCIe

See [Section 8.2](#) for details of how manageability messages (of which UCIe Memory Access Protocol and UDM are two subtypes) are negotiated, encapsulated, and transported over the UCIe sideband and Main band.

8.3.1.5 UCIe Test Port Options and Other Considerations

See [Chapter 5.0](#) for different port options for testing.

8.3.1.5.1 Determinism Considerations

Testing with low-cost ATE typically requires cycle-accurate determinism. When using UCIe as a test port, how the determinism is achieved end-to-end is implementation-specific and beyond the scope of this specification.

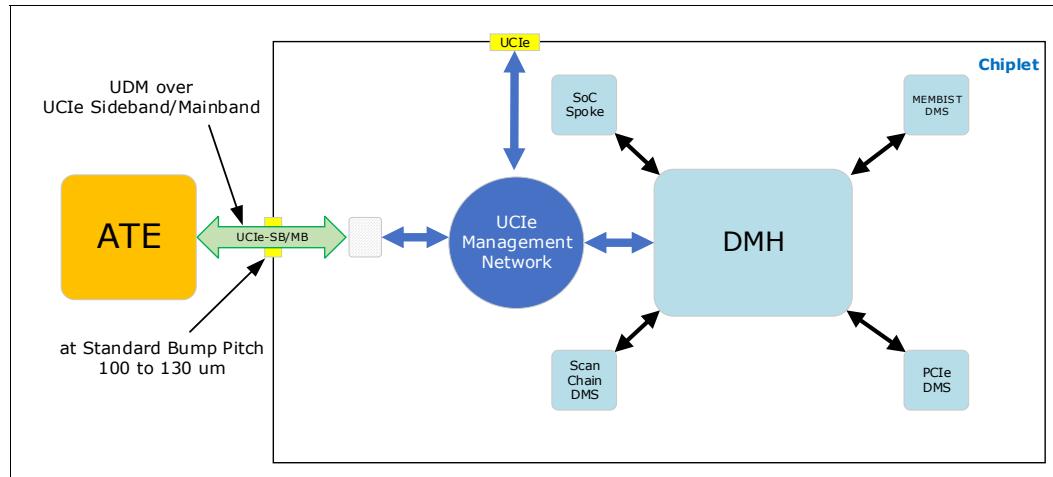
8.3.1.6 DFx Security

See [Section 8.1.3.5](#) for details.

8.3.2 Sort/Pre-bond Chiplet Testing with UDA

This section covers an overview of chiplet-level testing at Sort/Pre-bond using UCIe as the test port. Support for this testing scheme is optional. [Figure 8-52](#) captures this scenario.

Figure 8-52. UCIe-based Chiplet Testing/Debugging at Sort



- UCIe sideband and/or mainband can be used for this testing if they have a bump pitch of 100 um to 130 um.
- For sending/receiving scan test patterns, Vendor-defined UDMs (see [Figure 8-51](#)) are used over UCIe Sideband or mainband. These messages can target the appropriate Spoke (using the DMS-ID field) in the design that implements the scan functionality.
- For general-purpose testing/debugging using register reads and writes, UCIe UMAP messages can be used (e.g., for triggering in-build self-test mechanisms in a chiplet, a UCIe register read/write mechanism can be used to trigger a test and then read the test results).

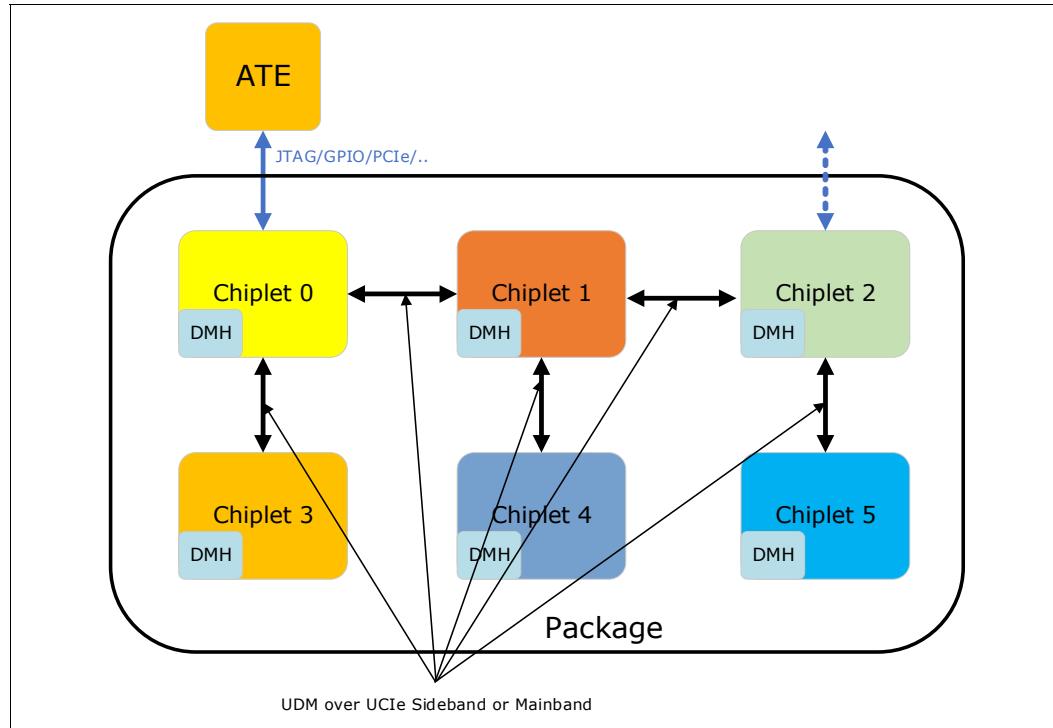
In [Figure 8-52](#), a UCIe Management port embedded in the UCIe controller provides access from the tester to the chiplet's manageability/test/debug fabric. The access control mechanism for ATE to acquire access to the UCIe Management network is implementation-specific.

While the ATE interfaces covered above are UCIe sideband and mainband, other interfaces such as JTAG, GPIO, and PCIe are also possible. Vendors can implement a bridge from these interfaces, with appropriate security control, to the UCIe Management network.

8.3.3 SiP-level Chiplet Testing with UDA

This section covers chiplet-level testing in a package that uses UCIe. Figure 8-53 provides an overview of this. Support for this testing scheme is optional.

Figure 8-53. UCIe-based Testing of Chiplets in an SiP

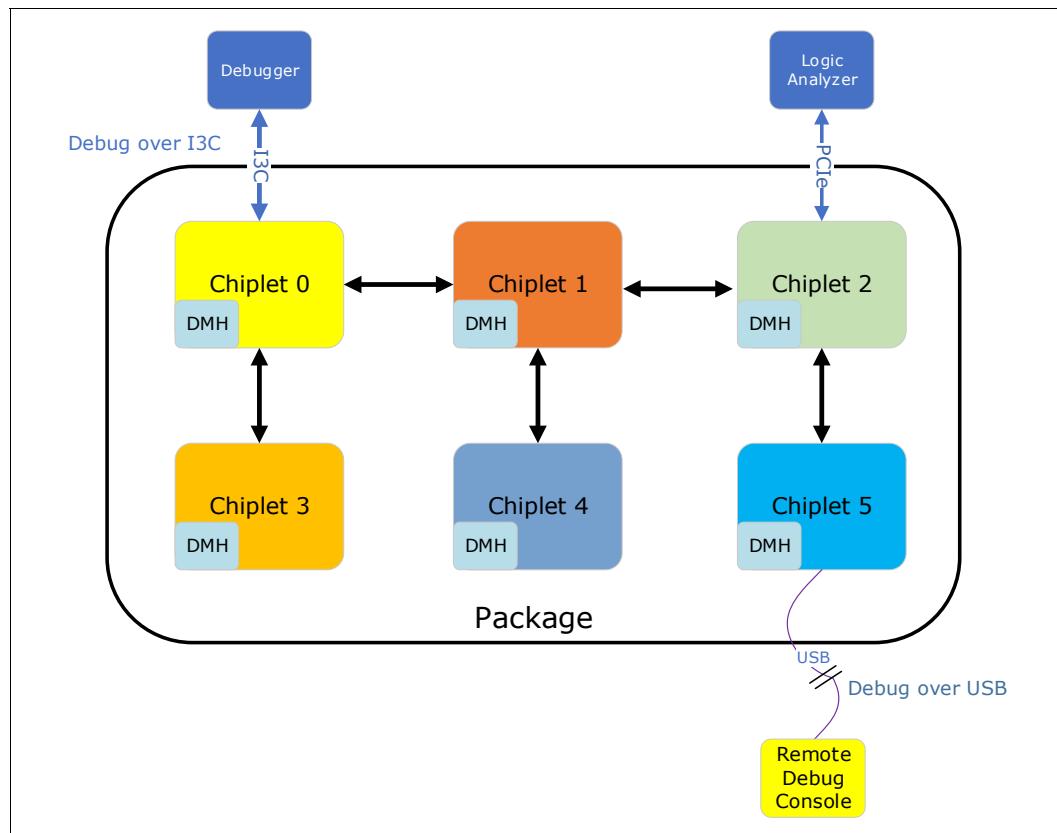


- There is at least one test/debug port pinned out in the package for SiP-level testing/debugging.
 - The port could be any of JTAG, GPIO, PCIe, USB, SMBus, and/or I₂(3)C.
- More than one package port can be used for speeding up package-level test/debug.
- Vendors can implement bridges, with appropriate security control, from these interfaces to the UCIe Management network.
 - On the UCIe Management network, bridged packets follow the UCIe Management Transport Packet format.
- Accesses from package ports are forwarded over UCIe sideband or mainband if they target other chiplets. See [Section 8.1.3.2](#) for details of how the target chiplet of a Manageability packet is determined.
- See [Section 8.2.1](#) for details of how UDMs are encapsulated on the UCIe sideband and mainband.
- Similar to sort testing,
 - For sending/receiving scan test patterns, Vendor-defined UCIe DFx Messages (UDM) are used over UCIe. These messages can target the appropriate Spoke in the design that implements the scan control functionality.
 - For general-purpose testing/debugging using register reads and writes, UMAP messages can be used, as defined in [Section 8.1.4](#).

8.3.4 System Debug with UDA

For system-level debug, various interfaces can be used for Controllability/Observability. In Figure 8-54, an I3C interface running a debug protocol is the connection between the debugger and the DFx hooks on each chiplet. A x16 PCIe interface is used to send debug data to a logic analyzer. PCIe debug VDM packets can be used to carry debug information on this interface. Similarly, a remote debugger could access debug data over USB using an appropriate protocol. Note that per the UCIe Management Network Architecture requirements, a management bridge is required at the USB interface in Chiplet 5, or I3C interface in Chiplet 0, or x16 PCIe interface in Chiplet 2.

Figure 8-54. UCIe-based System Testing/Debug



8.3.5 DMH/DMS Registers

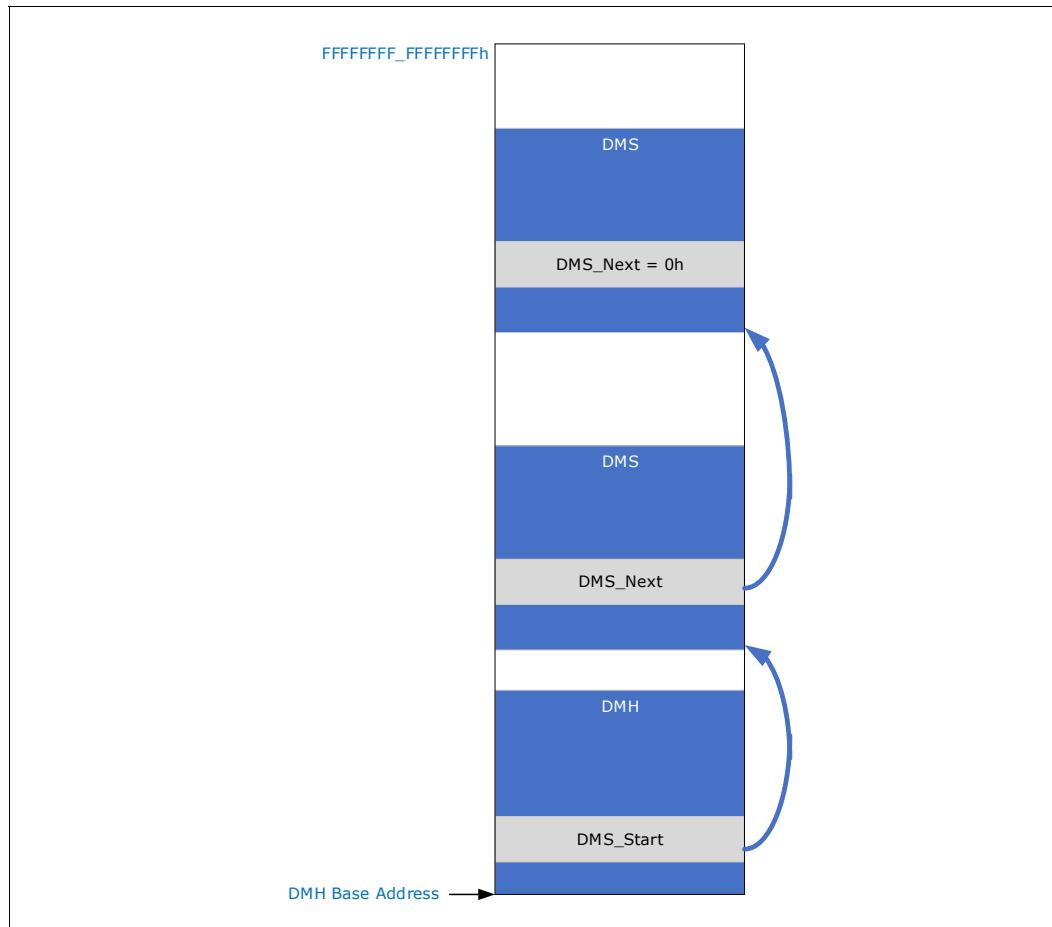
8.3.5.1 DMH/DMS Register Address Space and Access Mechanism

DMH and DMS registers are located within the memory space of the Management Element in which they reside. UMAP is used to access these registers. DMH and DMSs all share the same memory address space of the associated management element, and they each occupy specific address ranges within the address space. DMH and DMSs are discovered using a linked list with pointers in DMH and DMS register space, respectively. In Figure 8-55, DMH has two Spokes connected to it. The pointer in DMH points to the first DMS which then points to the next DMS. The pointer in the second DMS indicates the end of Spokes in the DMH with a value of all 0s in its Next pointer.

All spec-defined registers in DMH and DMS are accessed in DWORD size only.

The DMH base address in [Figure 8-55](#) is from the Capability Directory of Management Element that hosts a DMH (see [Section 8.1.3.6.1](#) for details).

Figure 8-55. DMH/DMS Address Mapping



8.3.5.2 DMH Registers

DMH registers are defined in the DMH Capability shown in [Figure 8-56](#). DBG_STS falls within the “Chiplet Status” Asset Class. All other spec-defined registers fall within the “Chiplet Configuration” asset class.

Figure 8-56. DMH Capability Register Map

31	29	16	15	8	7	0	
Rsvd	Capability ID		Reserved		Ver	0B	
	DBG_CAP						
	DBG_CTL						
	DBG_STS						
	DMH_Length_Low						
	DMH_Length_High						
	DMH_Ext_Cap_Low						
	DMH_Ext_Cap_High						
	DMS_Start_Low						
	DMS_Start_High						
	Reserved					40B	
						•	
						•	
						•	
						128B	
						•	
						•	
						•	
						N	
	Vendor-defined						

8.3.5.2.1 Ver (Offset 00h)**Table 8-32. Version**

Bit	Attribute	Description
7:0	RO	Version Set to 00h.

8.3.5.2.2 Capability ID (Offset 02h)**Table 8-33. Capability ID, Ver**

Bit	Attribute	Description
13:0	RO	Capability ID Set to 2h to indicate DMH.

8.3.5.2.3 DBG_CAP — Debug Capabilities (Offset 04h)

Table 8-34. Debug Capability

Bit	Attribute	Description
3:0	RO	Version Set to 0h.
31:4	RsvdP	Reserved

8.3.5.2.4 DBG_CTL — Debug Control (Offset 08h)

Table 8-35. Debug Control

Bit	Attribute	Description
0	RWL	Disable Accesses to/from DMS 1: Disables UMAP and Test/Debug Vendor-defined UDM accesses to/from DMSs connected to the DMH from/to the PCIe Management network. 0: Enables accesses to DMSs connected to the DMH. Default is 1b. This bit is locked for writes if bit 1 in this register is set to 1.
1	RO	Lock 'Disable Accesses to DMS' Default value is 0. After SW writes 1 to this bit, this bit cannot be modified further until the next Management Reset.
31:2	RsvdP	Reserved

8.3.5.2.5 DBG_STS — Debug Status (Offset Ah)

Table 8-36. Debug Status

Bit	Attribute	Description
0	RO	DMS Accessed At least one DMS was accessed from the management network since the last Management Reset. This bit is cleared on each Management Reset.
31:1	RsvdP	Reserved

8.3.5.2.6 DMH_Length_Low — DMH Register Space Length Low (Offset 10h)

Table 8-37. DMH_Length_Low

Bit	Attribute	Description
31:12	RO	Lower 20 bits of the length of the DMH register space in multiples of 4K. Bits [11:0] in this register are reserved to ensure 4k multiples of length. A value of 1000h for {DMH_Length_High :: DMH_Length_Low} indicates a length of 4K. A value 2000h indicates a length of 8K, etc.
11:0	RsvdP	Reserved

8.3.5.2.7 DMH_Length_High — DMH Register Space Length High (Offset 14h)

Table 8-38. DMH_Length_High

Bit	Attribute	Description
31:0	RO	Upper 32 bits of the length of the DMH register space in multiples of 4K. A value of 1000h for {DMH_Length_High :: DMH_Length_Low} indicates a length of 4K. A value 2000h indicates a length of 8K, etc.

8.3.5.2.8 DMH_Ext_Cap_Low — DMH Extended Capability Pointer Low (Offset 18h)

Table 8-39. DMH Extended Capability Pointer Low

Bit	Attribute	Description
31:2	RO	Lower 30 bits of the DWORD-aligned offset from the DMH starting address, where any extended capabilities start, when present in the DMH. Set to all 0s for this revision of the spec.
1:0	RsvdP	Reserved

8.3.5.2.9 DMH_Ext_Cap_High — DMH Extended Capability Pointer High (Offset 1Ch)

Table 8-40. DMH Extended Capability Pointer High

Bit	Attribute	Description
31:0	RO	Upper 32 bits of the DWORD-aligned offset from the DMH starting address, where any extended capabilities start, when present in the DMH. Set to all 0s for this revision of the spec.

8.3.5.2.10 DMS_Start_Low — DMS Starting Address Low (Offset 20h)

Table 8-41. DMS_Start_Low

Bit	Attribute	Description
31:12	RO	Lower 20 bits of the 4K-aligned starting address (in the UMAP address space of the management element that hosts the DMH) of the first DMS connected to the DMH.
11:0	RsvdP	Reserved

8.3.5.2.11 DMS_Start_High — DMS Starting Address High (Offset 24h)

Table 8-42. DMS_Start_High

Bit	Attribute	Description
31:0	RO	Upper 32 bits of the 4K-aligned starting address (in the UMAP address space of the management element that hosts the DMH) of the first DMS connected to the DMH.

8.3.5.3 DMS Registers

Architected DMS registers are detailed in this section. The UCIe Consortium-assigned Vendor ID at Offset 00h of the DMS register map uniquely identifies each UCIe Vendor. A value of 0h for the Vendor ID indicates that the Spoke is an “empty” Spoke and the register map for such a Spoke is shown in [Figure 8-57](#). For Spokes with a nonzero Vendor ID (also referred to as nonempty Spokes), a ‘Spoke Type’ value is defined that indicates whether the Spoke is associated with a UCIe link or if the Spoke is vendor-defined (see [Section 8.3.5.3.2.8](#) for ‘Spoke Type’ definition):

- If a Spoke is associated with a UCIe link, its ‘Spoke Type’ value is either 0, 1, or 2 (see [Figure 8-59](#) for the register map)
- If the Spoke is a vendor-defined Spoke, the ‘Spoke type’ value is assigned by the vendor within the range of 128 to 255 and some of the architected registers are not applicable (see [Figure 8-60](#) for the register map)

[Figure 8-58](#) shows registers that are common for all Spoke types. For security, DMS registers are classified as follows. See [Section 8.1.3.5.1](#) for the details of each class.

- Spoke STS register falls within the ‘Chiplet Status’ asset class.

- When applicable, UCIe Link Status in UCIe Link DVSEC, UCIe link-related status/log registers in the Adapter_Physical_Layer register block (e.g., Correctable/Uncorrectable Error Status), Compliance and Test-related status registers in the Compliance_Test register block (e.g., Physical Layer Compliance 1 and 2 Status registers), fall within the ‘SiP Status’ asset class.
 - All standard UCIe link registers other than the ones noted above fall within the ‘SiP Configuration’ asset class.
- All other spec-defined registers in DMS fall within the ‘Chiplet Configuration’ asset class.

8.3.5.3.1 “Empty” Spoke Registers

Figure 8-57 shows the register map for “empty” Spokes. Designs can use this Spoke register structure to indicate that the Spoke does not have any Spoke functionality.

Figure 8-57. Empty Spoke Register Map

31	16 15	0
Reserved	Spoke VID = 0h ^a	0B
DMS_Next_Low		4B
DMS_Next_High		8B

a. See Section 8.3.5.3.2.1.

8.3.5.3.1.1 DMS_Next_Low — DMS Next Low Address (Offset 04h)

Table 8-43. DMS_Next_Low Address

Bit	Attribute	Description
31:12	RO	Lower 20 bits of the 4K-aligned starting address (in the UMAP address space of the management element that hosts the DMH) of the next DMS connected to the DMH. If this is the last Spoke in the Spoke chain, this field needs to be set to all 0s.
11:0	RsvdP	Reserved

8.3.5.3.1.2 DMS_Next_High — DMS Next High Address (Offset 08h)

Table 8-44. DMS_Next_High Address

Bit	Attribute	Description
31:0	RO	Upper 32 bits of the 4K-aligned starting address (in the UMAP address space of the management element that hosts the DMH) of the next DMS connected to the DMH. If this is the last Spoke in the Spoke chain, this field needs to be set to all 0s.

8.3.5.3.2 Common DMS Registers for All Non-empty Spokes

Figure 8-58 shows the registers that are present in all non-empty Spokes. Locations marked as “Type-Specific” in Figure 8-58 carry registers that are specific to the ‘Spoke Type’ and are discussed in Section 8.3.5.3.3 and Section 8.3.5.3.4.

Figure 8-58. Common DMS Registers for All Non-empty Spokes Register Map (Sheet 1 of 2)

31	24 23	16 15	8 7	0
Spoke DevID		Spoke VID		0B
DMS_Next_Low				
DMS_Next_High				

Figure 8-58. Common DMS Registers for All Non-empty Spokes Register Map (Sheet 2 of 2)

31	24 23	16 15	8 7	0
Type-Specific		Reserved		Spoke RID
Associated DMS-ID2	Associated DMS-ID1	Associated DMS-ID0		DMS-ID
		Spoke CAP		
		Spoke CTL		
		Spoke STS		
		Spoke CTL		
		DMS_Length_Low		
		DMS_Length_High		
		DMS_Ext_Cap_Low		
		DMS_Ext_Cap_High		
		Type-Specific		48B
				•
				•
				•
		Reserved		80B
				•
				•
				•
		Type-Specific		128B
				•
				•
				N

8.3.5.3.2.1 Spoke VID — Spoke Vendor ID (Offset 00h)

Table 8-45. Spoke Vendor ID

Bit	Attribute	Description
15:0	RO	Spoke Vendor ID Uniquely identifies a Spoke Vendor to Software. This ID is assigned by UCIe Consortium.

8.3.5.3.2.2 Spoke DevID — Spoke Device ID (Offset 02h)

Table 8-46. Spoke Device ID

Bit	Attribute	Description
15:0	RO	Spoke Device ID Uniquely identifies a device from the Vendor identified by the Vendor ID. This ID is assigned by the vendor.

8.3.5.3.2.3 DMS_Next_Low — DMS Next Low Address (Offset 04h)

Table 8-47. DMS_Next_Low Address

Bit	Attribute	Description
31:12	RO	Lower 20 bits of the 4K-aligned starting address (in the UMAP address space of the management element that hosts the DMH) of the next DMS connected to the DMH. If this is the last Spoke in the Spoke chain, this field needs to be set to all 0s.
11:0	RsvdP	Reserved

8.3.5.3.2.4 DMS_Next_High — DMS Next High Address (Offset 08h)

Table 8-48. DMS_Next_High Address

Bit	Attribute	Description
31:0	RO	Upper 32 bits of the 4K-aligned starting address (in the UMAP address space of the management element that hosts the DMH) of the next DMS connected to the DMH. If this is the last Spoke in the Spoke chain, this field needs to be set to all 0s.

8.3.5.3.2.5 Spoke RID — Spoke Revision ID (Offset 0Ch)

Table 8-49. Spoke Revision ID

Bit	Attribute	Description
7:0	RO	Spoke Revision ID Uniquely identifies a Spoke Vendor to Software. This ID is assigned by UCIe Consortium.

8.3.5.3.2.6 DMS-ID — Spoke DMS-ID (Offset 10h)

Table 8-50. DMS-ID

Bit	Attribute	Description
7:0	RO	DFx Management Spoke-ID Statically assigned Spoke-ID value for this Spoke. Spoke-ID is used for ID-routed UDMs.

8.3.5.3.2.7 Associated DMS-ID[0, 1, 2] (Offsets 11h, 12h, 13h)

Table 8-51. Associated DMS-ID[0, 1, 2]

Bit	Attribute	Description
7:0	RO	Associated DMS-ID Spoke-ID associated with other Spokes that constitute the same UCIe link. For example, if there are separate Spokes for some or all of the IPs that constitute a full UCIe stack – Adapter, Physical Layer, Protocol Stack0, Protocol Stack1 – these registers within each Spoke provide the DMS-IDs of the related partner Spokes. If there are no related Spokes, this register reads as FFH. If there are multiple protocol stacks, the lower value DMS-ID belongs to Stack 0 and higher value belongs to Stack 1. These registers are used by SW to identify all the Spokes that constitute a single UCIe link.

8.3.5.3.2.8 Spoke CAP — Spoke Capability (Offset 14h)

Table 8-52. Spoke Capability

Bit	Attribute	Description
3:0	RO	Version Set to 0h for this version of the capability.
7:4	RsvdP	Reserved
15:8	RO	Spoke Type 0: UCIe.Adapter. Indicates a Spoke associated with UCIe Adapter. 1: UCIe.Physical_Layer. Indicates a Spoke associated with UCIe Physical Layer. 2: UCIe.Adapter_Physical_Layer. Indicates a common Spoke across both UCIe Adapter and Physical Layer. 3 to 127: Reserved. 128 to 255: Vendor-defined.
31:16	RsvdP	Reserved

8.3.5.3.2.9 Spoke CTL — Spoke Control (Offset 18h)

Table 8-53. Spoke Control

Bit	Attribute	Description
0	RW	Enable Test and Debug Vendor-defined UDM as Initiator 0: Spoke cannot initiate Vendor-defined UDM. 1: Spoke can initiate Vendor-defined UDM. Spokes that do not implement Vendor-defined UDM as initiator can hardwire this bit to 0.
31:1	RsvdP	Reserved

8.3.5.3.2.10 Spoke STS — Spoke Status (Offset 1Ch)

Table 8-54. Spoke Status

Bit	Attribute	Description
0	RO	Spoke Used Indicates that the Spoke has been accessed at least once since the last Management Reset. Access implies sending or receiving UMAP packets or UDMs. Bit is cleared on the next Management Reset.
31:1	RsvdP	Reserved

8.3.5.3.2.11 DMS_Length_Low — DMS Register Space Length Low (Offset 20h)

Table 8-55. DMS Register Space Length Low

Bit	Attribute	Description
31:12	RO	Lower 20 bits of length of the DMS register space from Offset 0h of DMS, in multiples of 4K. Value of 1000h for {DMS_Length_High :: DMS_Length_Low} indicates 4K length, 2000h indicates 8K length, etc. Bits [11:0] in this register are reserved to ensure 4k multiples of length. UCIE Spoke Types 0, 1, and 2 implemented to this revision of the spec must have a value in this register such that the DMS register space is not larger than 4 MB.
11:0	RsvdP	Reserved

8.3.5.3.2.12 DMS_Length_High — DMS Register Space Length High (Offset 24h)

Table 8-56. DMS Register Space Length High

Bit	Attribute	Description
31:0	RO	Upper 32 bits of length of the DMS register space from Offset 0h of DMS, in multiples of 4K. Value of 1000h for {DMS_Length_High :: DMS_Length_Low} indicates 4K length, 2000h indicates 8K length, etc. UCIE Spoke Types 0, 1, and 2 implemented to this revision of the spec must set this value to all 0s.

8.3.5.3.2.13 DMS_Ext_Cap_Low — DMS Extended Capability Pointer Low (Offset 28h)

Table 8-57. DMS Extended Capability Pointer Low

Bit	Attribute	Description
31:2	RO	Lower 30 bits of the DWORD-aligned offset from the DMS starting address, where any extended capabilities start, when present in the DMS. Value of all 0s indicates that there are no extended capabilities (default).
1:0	RsvdP	Reserved

8.3.5.3.2.14 DMS_Ext_Cap_High — DMS Extended Capability Pointer High (Offset 2Ch)

Table 8-58. DMS Extended Capability Pointer High

Bit	Attribute	Description
31:0	RO	Upper 32 bits of the DWORD-aligned offset from the DMS starting address, where any extended capabilities start, when present in the DMS. Value of all 0s indicates that there are no extended capabilities (default).

8.3.5.3.3 DMS Registers of UCIe Spoke Types ('Spoke Type' = 0 through 2)

Figure 8-59 shows the DMS register map for the UCIe Spoke types. Figure 8-58 and Section 8.3.5.3.2 detail registers that are common to all Spoke types. This section details the remaining registers, which are unique to the UCIe Spoke types.

Figure 8-59. DMS Register Map for UCIe Spoke Types

31	24 23	16 15	8 7	0
	Spoke DevID		Spoke VID	0B
	DMS_Next_Low			
	DMS_Next_High			
	Port ID	Reserved	Spoke RID	
Associated DMS-ID2	Associated DMS-ID1	Associated DMS-ID0	DMS-ID	
	Spoke CAP			
	Spoke CTL			
	Spoke STS			
	Spoke CTL			
	DMS_Length_Low			
	DMS_Length_High			
	DMS_Ext_Cap_Low			
	DMS_Ext_Cap_High			
	Adapter_Physical_Layer_Ptr_Low			48B
	Adapter_Physical_Layer_Ptr_High			
	Compliance_Test_Ptr_Low			
	Compliance_Test_Ptr_High			
	Impl_Spec_Adapter_Ptr_Low			
	Impl_Spec_Adapter_Ptr_High			
	Impl_Spec_Physical_Layer_Ptr_Low			
	Impl_Spec_Physical_Layer_Ptr_High			
	Reserved			80B
	UCIe Link DVSEC			128B
	Vendor-defined			•
	UCIe Link Register Blocks			•
	Vendor-defined			•
				N

8.3.5.3.3.1 Port ID — Management Port ID (Offset 1Eh)

Table 8-59. Port ID

Bit	Attribute	Description
15:0	RO	Port ID For Spoke Types 0, 1, and 2, this register indicates the Port ID of the UCIe link that is associated with the Spoke, if a Port ID exists for the link. A UCIe link has a Port ID assigned to it if the link is a Management Port. If the link does not have an assigned Port ID, this register reads as FFFFh.

8.3.5.3.3.2 Adapter_Physical_Layer_Ptr_Low — Adapter/Physical Layer Register Block Pointer Low (Offset 30h)

Table 8-60. Adapter_Physical_Layer_Ptr_Low

Bit	Attribute	Description
31:12	RO	Lower 20 bits of the 4K-aligned offset (from the starting address of the Spoke) of the UCIe Adapter/Physical Layer register block that is associated with the UCIe link. Accesses to registers that are referenced by Adapter_Physical_Layer_Ptr_Low/High pointers in a UCIe.Adapter Spoke are limited to the 4k block(s) that contain the Adapter registers and the register block header itself, and the 4k block(s) that contain Physical Layer registers are treated as reserved. Accesses to registers that are referenced by Adapter_Physical_Layer_Ptr_Low/High pointers in a UCIe.Physical_Layer Spoke are limited to the 4k block(s) that contain the PHY registers and the register block header itself, and Adapter registers are treated as reserved.
11:0	RsvdP	Reserved

8.3.5.3.3.3 Adapter_Physical_Layer_Ptr_High — Adapter/PHY Register Block Pointer High (Offset 34h)

Table 8-61. Adapter_Physical_Layer_Ptr_High

Bit	Attribute	Description
31:0	RO	Upper 32 bits of the 4K-aligned offset (from the starting address of the Spoke) of the UCIe Adapter/PHY register block that is associated with the UCIe link. Accesses to registers that are referenced by Adapter_Physical_Layer_Ptr_Low/High pointers in a UCIe.Adapter Spoke are limited to the 4k block(s) that contain the Adapter registers and the register block header itself, and the 4k block(s) that contain PHY registers are treated as reserved. Accesses to registers that are referenced by Adapter_Physical_Layer_Ptr_Low/High pointers in a UCIe.Physical_Layer Spoke are limited to the 4k block(s) that contain the PHY registers and the register block header itself, and Adapter registers are treated as reserved.

8.3.5.3.3.4 Compliance_Test_Ptr_Low — Compliance and Test Register Block Pointer Low (Offset 38h)

Table 8-62. **Compliance_Test_Ptr_Low**

Bit	Attribute	Description
31:12	RO	<p>Lower 20 bits of the 4K-aligned offset (from the starting address of the Spoke) of the UCIe Test/Compliance register block that is associated with the UCIe link.</p> <p>Accesses to registers that are referenced by Compliance_Test_Ptr_Low/High pointers in a UCIe.Adapter Spoke are limited to the 4k block(s) that contain the Adapter registers and the register block header itself, and the 4k block(s) that contain PHY registers are treated as reserved.</p> <p>Accesses to registers that are referenced by Compliance_Test_Ptr_Low/High pointers in a UCIe.Physical_Layer Spoke are limited to the 4k block(s) that contain the PHY registers and the register block header itself, and the Adapter registers are treated as reserved.</p> <p>Accesses to registers that are referenced by Compliance_Test_Ptr_Low/High pointers in a UCIe.Adapter_Physical_Layer Spoke have no access restrictions.</p> <p>Set to all 0s if this register block is not implemented.</p>
11:0	RsvdP	Reserved

8.3.5.3.3.5 Compliance_Test_Ptr_High — Compliance and Test Register Block Pointer High (Offset 3Ch)

Table 8-63. **Compliance_Test_Ptr_High**

Bit	Attribute	Description
31:0	RO	<p>Upper 32 bits of the 4K-aligned offset (from the starting address of the Spoke) of the UCIe Test/Compliance register block that is associated with the UCIe link.</p> <p>Accesses to registers that are referenced by Compliance_Test_Ptr_Low/High pointers in a UCIe.Adapter Spoke are limited to the 4k block(s) that contain the Adapter registers and the register block header itself, and the 4k block(s) that contain PHY registers are treated as reserved.</p> <p>Accesses to registers that are referenced by Compliance_Test_Ptr_Low/High pointers in a UCIe.Physical_Layer Spoke are limited to the 4k block(s) that contain the PHY registers and the register block header itself, and the Adapter registers are treated as reserved.</p> <p>Accesses to registers that are referenced by Compliance_Test_Ptr_Low/High pointers in a UCIe.Adapter_Physical_Layer Spoke have no access restrictions.</p> <p>Set to all 0s if this register block is not implemented.</p>

8.3.5.3.3.6 **Impl_Spec_Adapter_Ptr_Low — Implementation-specific Adapter Register Block Pointer Low (Offset 40h)**

Table 8-64. **Impl_Spec_Adapter_Ptr_Low**

Bit	Attribute	Description
31:12	RO	Lower 20 bits of the 4K-aligned offset (from the starting address of the Spoke) of the Adapter Implementation-specific register block. In a UCIE.Physical_Layer Spoke type, this pointer must be set to all 0s. Also set to all 0s if the register block is not implemented in the design.
11:0	RsvdP	Reserved

8.3.5.3.3.7 **Impl_Spec_Adapter_Ptr_High — Implementation-specific Adapter Register Block Pointer High (Offset 44h)**

Table 8-65. **Impl_Spec_Adapter_Ptr_High**

Bit	Attribute	Description
31:0	RO	Upper 32 bits of the 4K-aligned offset (from the starting address of the Spoke) of the Adapter Implementation-specific register block. In a UCIE.Physical_Layer Spoke type, this pointer must be set to all 0s. Also set to all 0s if the register block is not implemented in the design.

8.3.5.3.3.8 **Impl_Spec_Physical_Layer_Ptr_Low — Implementation-specific Physical Layer Register Block Low (Offset 48h)**

Table 8-66. **Impl_Spec_Physical_Layer_Ptr_Low**

Bit	Attribute	Description
31:12	RO	Lower 20 bits of the 4K-aligned offset (from the starting address of the Spoke) of the Physical Layer Implementation-specific register block. In a UCIE.Adapter Spoke type, this pointer must be set to all 0s. Also set to all 0s if the register block is not implemented in the design.
11:0	RsvdP	Reserved

8.3.5.3.3.9 **Impl_Spec_Physical_Layer_Ptr_High — Implementation-specific Physical Layer Register Block High (Offset 4Ch)**

Table 8-67. **Impl_Spec_Physical_Layer_Ptr_High**

Bit	Attribute	Description
31:0	RO	Upper 32 bits of the 4K-aligned offset (from the starting address of the Spoke) of the Physical Layer Implementation-specific register block. In a UCIE.Adapter Spoke type, this pointer must be set to all 0s. Also set to all 0s if the register block is not implemented in the design.

8.3.5.3.3.10 UCIe Link DVSEC — UCIe Link DVSEC (Offset 80h)

UCIe Link DVSEC (see [Section 9.5.1](#)) is mirrored starting at this location. Accesses to the DVSEC by the UCIe.Physical_Layer Spoke type are treated as reserved.

IMPLEMENTATION NOTE

Spokes can restrict access to UCIe link registers based on access control considerations (see [Section 8.1.3.5](#) for details).

8.3.5.3.3.11 UCIe Link Register Blocks (Offset Is Implementation-dependent)

UCIe link memory register blocks — Adapter_Physical_Layer, Compliance_Test (if supported), Impl_Spec_Adapter (if supported), and Impl_Spec_Physical_Layer (if supported) — are mirrored at vendor-defined offsets in the Spoke's memory space.

8.3.5.3.4 DMS Registers of Vendor-defined Spoke Types ('Spoke Type' = 128 through 255)

[Figure 8-60](#) shows the DMS register map for the Vendor-defined Spoke types. [Figure 8-58](#) and [Section 8.3.5.3.2](#) detail registers that are common to all Spoke types. [Section 8.3.5.3.3.1](#) details the Port ID register. Vendor-defined Spokes do not have any additional architected registers.

Figure 8-60. DMS Register Map for Vendor-defined Spoke Types

31	24 23	16 15	8 7	0
	Spoke DevID		Spoke VID	0B
	DMS_Next_Low			
	DMS_Next_High			
	Port ID	Reserved	Spoke RID	
Associated DMS-ID2	Associated DMS-ID1	Associated DMS-ID0	DMS-ID	
	Spoke CAP			
	Spoke CTL			
	Spoke STS			
	Spoke CTL			
	DMS_Length_Low			
	DMS_Length_High			
	DMS_Ext_Cap_Low			
	DMS_Ext_Cap_High			
	Reserved			48B
				•
				•
				•
	Vendor-defined			128B
				•
				•
				•
				N

8.3.5.3.5 DMS Register Implementation in UCIe Adapter and in UCIe Physical Layer

IMPLEMENTATION NOTE

For Spoke Type 0, the DMS registers are implemented in the Adapter. For Spoke Type 1, the DMS registers are implemented in the Physical Layer. For Spoke Type 2, all but the register blocks associated with the Physical Layer are implemented in the Adapter. These registers are accessed over the FDI config bus (`lp_cfg*/p1_cfg*`) using DMS Register read/write opcodes (see [Table 7-1, "Opcode Encodings Mapped to Packet Types"](#)). SoC logic that interfaces with on-die management fabric (which is implementation-specific) is required to perform the conversion from Management Transport protocol UMAP packets to FDI config bus packets. The FDI config bus is defined in [Section 10.2](#).

§ §

9.0 Configuration and Parameters

9.1 High-level Software View of UCIe

A key goal of UCIe is to leverage all the software investments made for PCIe and CXL while still defining the interface in an extensible way for future innovative solutions. To that end, UCIe SW view of the protocol layer is consistent with the associated protocol. For example, the host Downstream Port for UCIe that is capable of supporting CXL protocols will appear to software as a Root Port with CXL DVSEC capability and relevant PCIe capabilities. Similarly, a host downstream port for UCIe that is capable of supporting PCIe protocol only, will appear to software as a Root Port with relevant PCIe capabilities only. Host side or device side view of software for Streaming protocol is implementation-specific since the protocol itself is implementation-specific. It is though strongly recommended that ecosystem implementations define streaming solutions leveraging the SW hooks already in place for supporting CXL and PCIe. The Upstream Ports that connect to a UCIe Root Port can be a PCIe endpoint, PCIe Switch, a CXL endpoint-device, or a CXL Switch. This allows for UCIe solution to be fully backward compatible to pre-UCIe software. The remainder of this chapter talks about SW view of UCIe when paired with PCIe or CXL protocol layers.

UCIe specification allows for a single UCIe Link to be shared by multiple protocol stacks. In this version of the spec, this sharing is limited to at most 2 protocol stacks. Shared Link layer is a new concept from Software perspective and requires new discovery/control mechanisms. The mechanism by which UCIe-aware SW discovers UCIe capability is described in the next section.

Table 9-1 shows the legal/illegal combinations of Upstream and Downstream devices/ports at a given UCIe interface, from a SW viewpoint.

Table 9-1. Software view of Upstream and Downstream Device at UCIe interface

Downstream Component: SW View	Upstream Component: SW View			
	PCIe RP, PCIe Switch DSP ^a	CXL-RP, CXL Switch DSP ^b	CXL Downstream Port RCRB ^c	Streaming Device
PCIe EP, PCIe Switch USP	Valid	Valid	Illegal	
CXL Upstream Port RCRB ^d	Illegal	Illegal	Illegal	Vendor defined
CXL EP	Valid	Valid	Illegal	
Streaming Device	Vendor defined			

a. PCIe RP = As defined in *PCIe Base Specification*

b. CXL RP/Switch DSP = Standard PCIe RP/Switch-DSP with additional CXL Flexbus Port DVSEC capability

c. CXL Downstream Port RCRB = CXL Link at host or at Switch DSP that is enumerated via CXL defined Downstream Port RCRB (instead of via a Root Port)

d. CXL Upstream Port RCRB = CXL upstream port that is enumerated via CXL defined RCRB with CXL Upstream Port RCRB and that has a RCIEP below it.

All the CXL/PCIe legacy/advanced capabilities/registers defined in the respective specifications apply to UCIe host and devices as well. Some Link and PHY layer specific registers in *PCIe Base Specification* do not apply in UCIe context and these are listed in the appendix. In addition, two new

DVSEC capabilities and four other MMIO mapped register blocks are defined to deal with UCIe-specific Adapter and Physical Layer capabilities.

9.2 SW Discovery of UCIe Links

UCIe-aware Firmware/Software may discover the presence and capabilities of UCIe Links in the system per Table 9-2.

Table 9-2. SW discovery of UCIe Links

UCIe Links	How discovered?	Salient Points
In Host	Host specific Register Block called UiRB, containing UCIe Link DVSEC Capability	<ul style="list-style-type: none"> UiRB is at a host defined static location. Each UCIe Link has a separate UiRB Base address and these are enumerated to OS via UCIe Early discovery table (UEDT)^a Association of a UCIe Link to 1 or more Root ports is described in UEDT, allowing for UCIe-aware SW to understand the potential shared nature of the UCIe Link.
In Endpoints	Dev0/Fn0 of the device carries a UCIe Link DVSEC Capability.	<ul style="list-style-type: none"> In multi-stack implementations, Dev0/Fn0 of the endpoint in only one of the stacks carries the UCIe Link DVSEC Capability.
In Switch USP	Dev0/Fn0 of the USP carrying a UCIe Link DVSEC Capability	<ul style="list-style-type: none"> In multi-stack implementations, Dev0/Fn0 of the USP in only one of the stacks carries the UCIe Link DVSEC Capability.
In Switch DSP	Dev0/Fn0 of the Switch USP carrying one or more UiSRB DVSEC Capability	<ul style="list-style-type: none"> UCIe Links below the switch are described in UiSRB whose base address is provided in the UiSRB DVSEC Capability A UCIe Link DVSEC capability per downstream UCIe Link is present in the UiSRB Association of a UCIe Link to 1 or more Switch DSPs is described as part of the UCIe Link DVSEC Capability, allowing for UCIe-aware SW to understand the potential shared nature of the UCIe interface <p>Note: It is legal for a Switch USP to carry the UiSRB DVSEC capability but not a UCIe Link DVSEC Capability</p>

a. UEDT structure is standardized as part of the ACPI specification.

9.3 Register Location Details and Access Mechanism

- 2 UCIe DVSEC capabilities (UCIe Link DVSEC, UiSRB DVSEC) and four other MMIO-mapped register blocks are defined in this version of the Specification.
- UCIe Link DVSEC capability is located in UiRB for host root ports and in UiSRB for Switch downstream ports.
- UiRB region is defined at a static location on the host side and its size is enumerated in the UEDT structure. Only UCIe Link related registers are permitted in this region and designs must not implement non-UCIe related functionality in this region.
- There is a unique UiRB base address for each UCIe Link, in the host
- UiSRB region base address is provided in the UiSRB DVSEC capability. This region is part of a BAR region of Switch Dev0/Fn0 USP.
- For scalability/flexibility reasons, multiple UiSRB DVSEC capabilities can exist in a Switch USP function. In case of multiple UiSRB DVSEC capabilities in the USP, a given DSP UCIe Link can only be described in one of the UiSRB structures.
- Configuration space registers are accessed using configuration reads and configuration writes. Register Blocks are in memory mapped regions and are accessed using standard memory reads and memory writes.
- UCIe Retimer registers are not directly accessible from host SW. They can be accessed only by way of a Mailbox mechanism over the sideband interface (hence the terms *SB-MMIO* and *SB-*

Config in [Table 9-3](#)). The Mailbox mechanism is available via RP/DSP UCIE Link DVSEC Capability to access the UCIE Retimer registers on the Retimer closest to the host. For accessing UCIE Retimer registers on the far end Retimer, the same Mailbox mechanism is also available in the UCIE Link DVSEC capability of EP/USP. See [Section 9.5.1.11](#) and [Section 9.5.1.12](#) for details of the Mailbox mechanism.

- For debug and runtime Link health monitoring reasons, host SW can also access the UCIE related registers in any partner die on the sideband interface, using the same Mailbox mechanism. For brevity purposes, that is not shown in [Table 9-3](#). Note that register accesses over sideband are limited to only the UCIE-related Capability registers (the two DVSECs currently defined in the spec) and the four defined UCIE Register Blocks. Nothing else on the remote die are accessible via the sideband mechanism.

[Table 9-3](#) summarizes the location of various register blocks in each native UCIE port/device. Henceforth a “UCIE port/device/EP/Switch” is used to refer to a standard PCIe or CXL port/device/EP/Switch with UCIE Link DVSEC Capability.

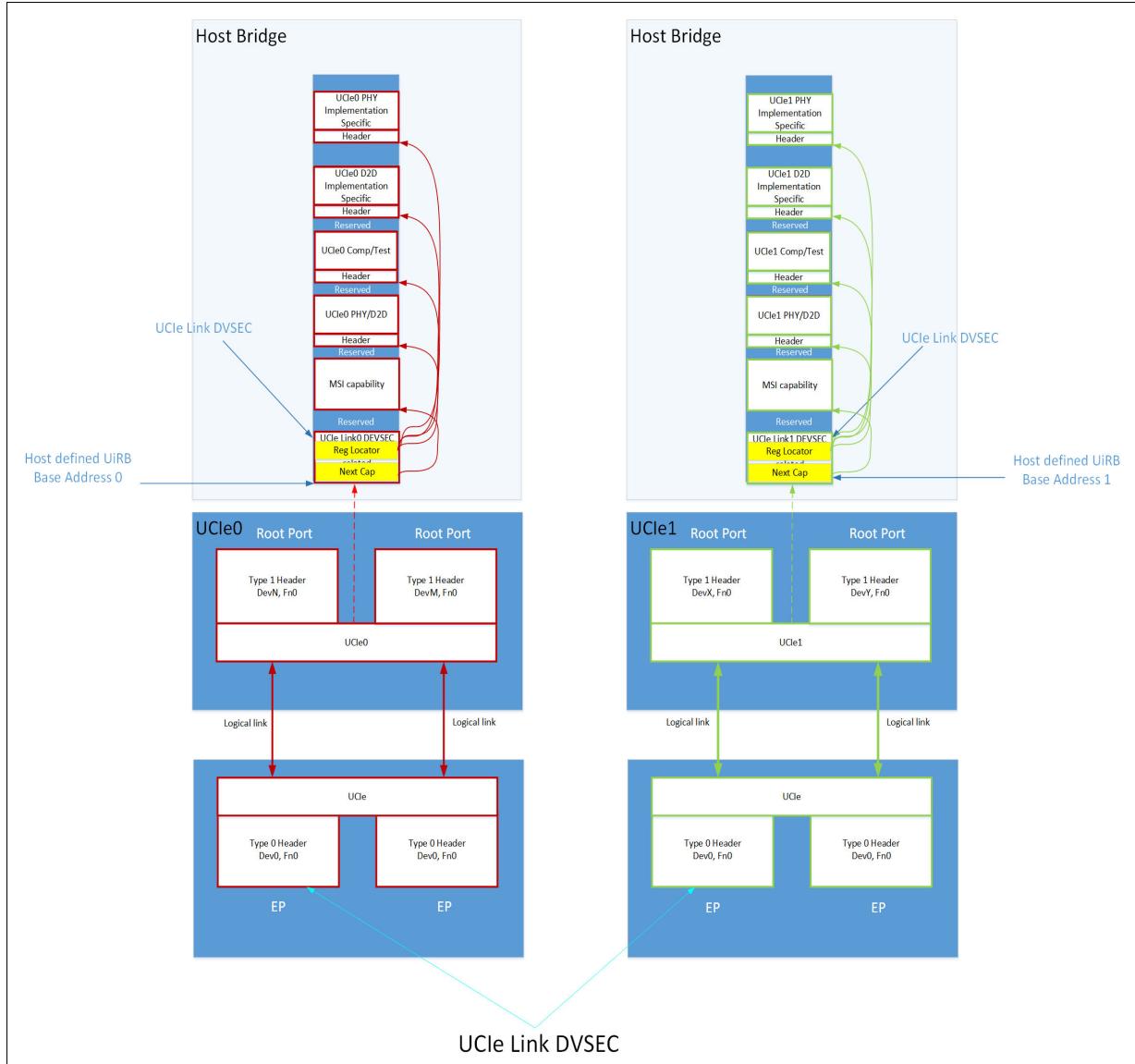
Table 9-3. Summary of location of various UCIE Link related registers

Register	Where the Register Resides					Comments
	RP	Switch USP	Switch DSP	EP	UCIE Retimer	
UCIE Link DVSEC	UiRB	Config space	UiSRB	Config Space	Sideband Config Space	Registers that define the basic UCIE interface related details
UCIE D2D/PHY Register Block	UiRB	Switch USP-BAR Region	UiSRB	EP-BAR Region	SB-MMIO Space	Registers that define lower-level functionality for the D2D/PHY interface of a typical UCIE implementation
UCIE Test/Compliance Register Block	UiRB	Switch USP-BAR Region	UiSRB	EP-BAR Region	SB-MMIO Space	Registers for Test/Compliance of UCIE interface
UCIE Implementation Specific Register Block	UiRB	Switch USP-BAR Region	UiSRB	EP-BAR Region	SB-MMIO Space	Registers for vendor specific implementation

9.4 Software view Examples

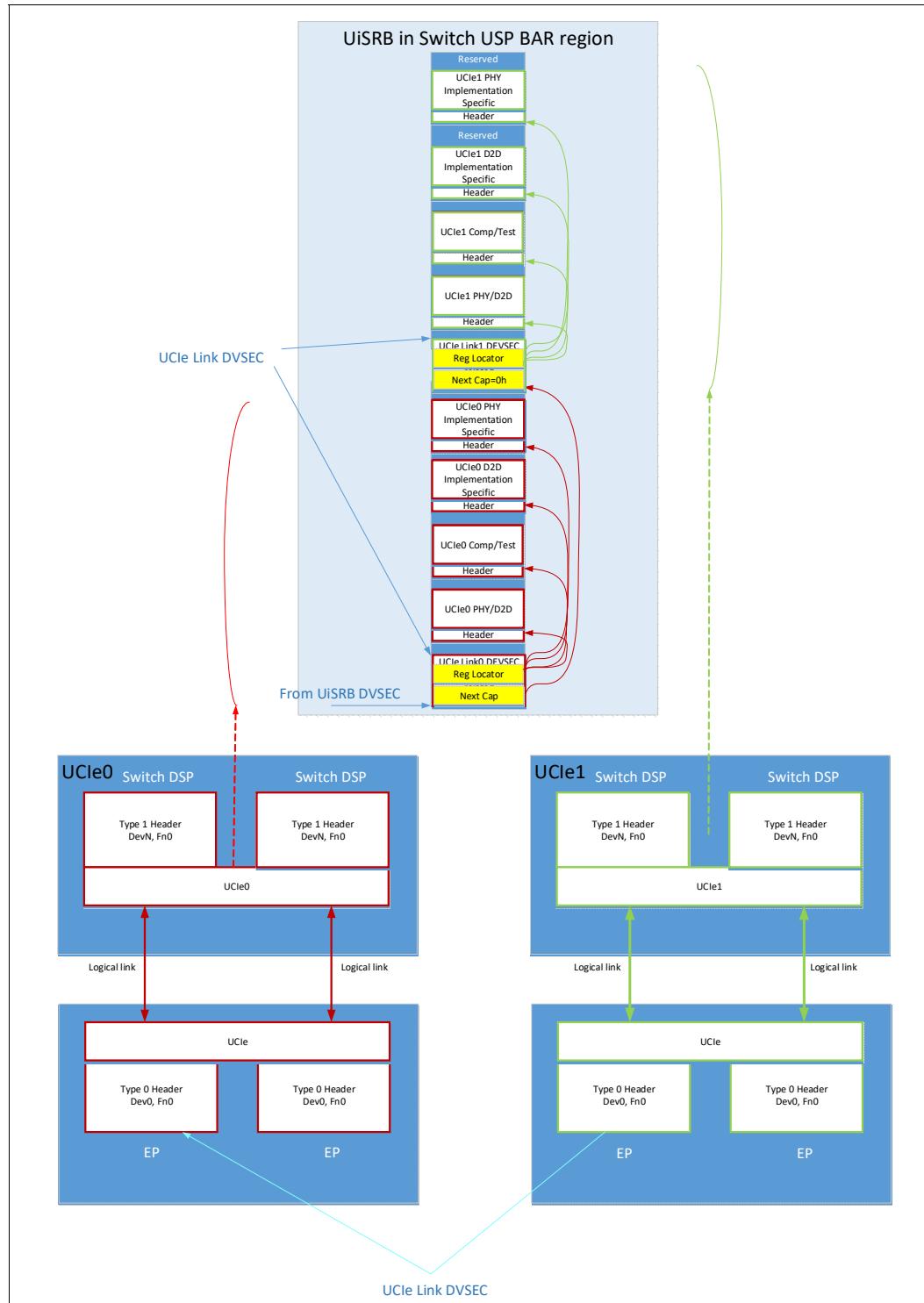
Figure 9-1 summarizes all the details of UCIe related DVSEC Capabilities and SW discovery, for an implementation consisting of Root Ports and Endpoints. This example has a host with 2 UCIe downstream Links that each carry traffic from 2 Root Ports.

Figure 9-1. Software view Example with Root Ports and Endpoints



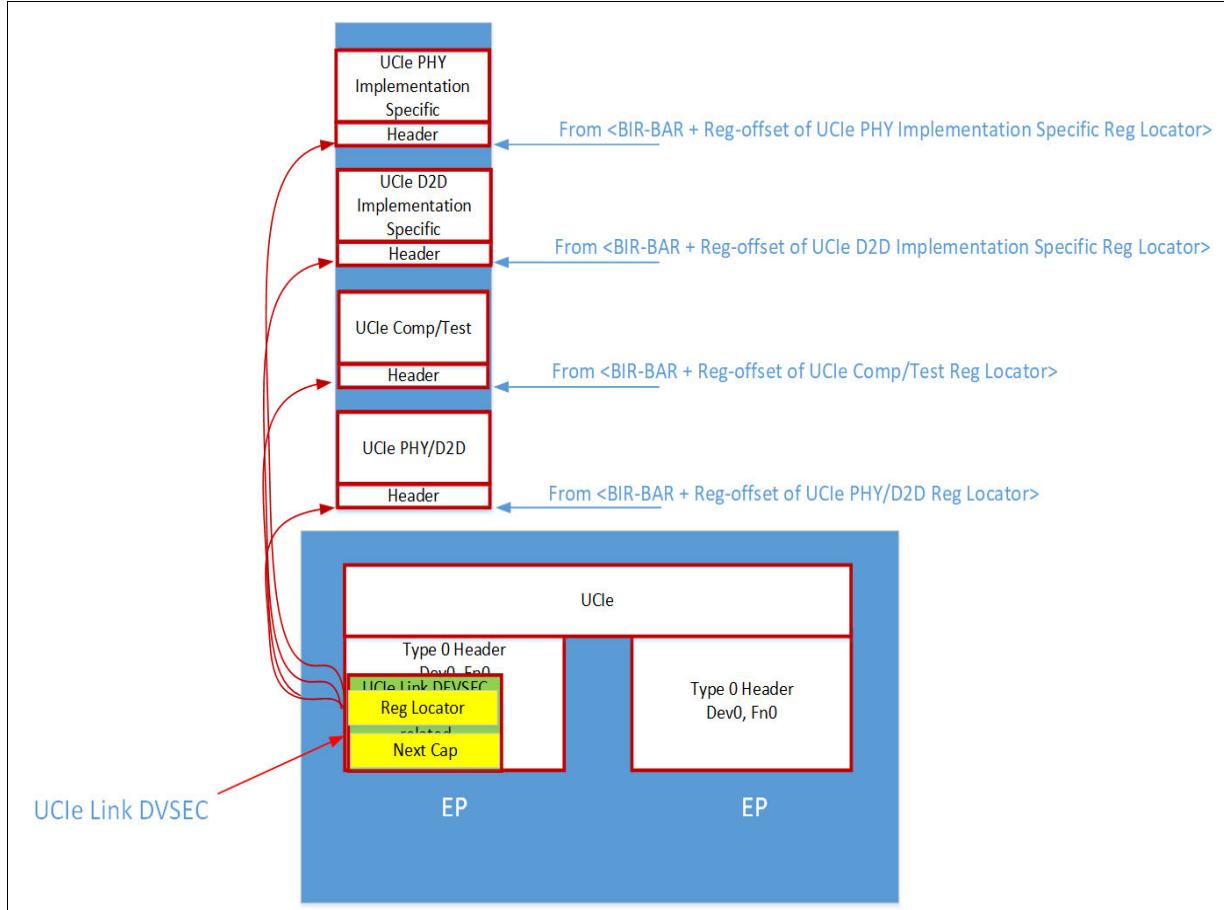
Example in [Figure 9-2](#) has a Switch with 2 UCIe Links on its downstream side and each UCIe Link carries traffic from 2 Switch DSPs.

Figure 9-2. Software view Example with Switch and Endpoints



Example in [Figure 9-3](#) shows details UCIE registers in an implementation where two EPs are sharing a common UCIE Link.

Figure 9-3. Software view Example of UCIE Endpoint



9.5 UCIE Registers

Table 9-4 summarizes the attributes for the register bits defined in this chapter. Unless otherwise specified, the definition of these attributes is consistent with *PCIe Base Specification* and *CXL Specification*.

Table 9-4. Register Attributes

Attribute	Description
RO	Read Only
ROS	Read Only Sticky ^a
RW	Read-Write
RWL	Read Write Lock Follow RW behavior until locked. When locked, the bit value cannot be altered by software. The locking condition associated with each RWL field is specified as part of the field definition.
RWO	Read-Write-One-To-Lock Field becomes RO after writing 1 to it. Cleared by management reset.
RWS	Read Write Sticky ^a
RW1C	Read-Write-One-To-Clear
RW1CS	Read-Write-One-To-Clear-Sticky ^a
HWInit	Hardware Initialized ^b
RsvdP	Reserved and Preserved
RsvdZ	Reserved and Zero

- a. Definition of 'sticky' follows the underlying protocol definition if any of the Protocol stacks are PCIe or CXL. For Streaming, the sticky registers are recommended to preserve their value even if the Link is down. In all scenarios, Domain Reset must initialize these to their default values.
- b. Typically, this register attribute is used for functionality/capability that can vary with package integration. For example, a chiplet that is capable of 32 GT/s maximum speed might be routed to achieve a maximum speed of 16 GT/s in a given package implementation. To account for such scenarios, the Max link speed field in the UCIE Link Capability register has the HWInit attribute and its value could be configured by a package-level strap or device/system firmware to reflect the maximum speed of that implementation.

All numeric values in various data structures, individual registers and register fields defined in this chapter are always encoded in little endian format, unless stated otherwise.

9.5.1 UCIE Link DVSEC

This is the basic capability register set that is required to operate a UCIE Link. And this is one of two DVSEC capabilities defined for UCIE in the first generation. Not all the registers in the capability are applicable to all device/port types. The applicable registers for each device/port type are indicated in the right side of [Figure 9-4](#). Software may use the presence of this DVSEC to differentiate between a UCIE device vs. a standard PCIe or CXL device. Software may use this DVSEC to differentiate between a UCIE Root Port and a standard PCIe or CXL Root Port.

Figure 9-4. UCIe Link DVSEC

The diagram illustrates the UCIe Link DVSEC structure, which is organized into several horizontal sections. A vertical bracket on the right side groups these sections into four categories labeled 'a', 'b', 'c', and 'd'. Category 'a' covers the first three sections: PCI Express Extended Capability Header, Designated Vendor Specific Header 1, and Capability Descriptor. Category 'b' covers the next five sections: Designated Vendor Specific Header 2, UCIe Link Capability, UCIe Link Control^e, UCIe Link Status, and Error Notification Control. Category 'c' covers the Link Event Notification Control, Register Locator 0 Low, Register Locator 0 High, and three reserved fields (indicated by three dots). Category 'd' covers the Sideband Mailbox Index Low, Sideband Mailbox Index High, Sideband Mailbox Data Low, Sideband Mailbox Data High, and the final group of registers: Reserved, Sideband Mailbox Status, Sideband Mailbox Control, Requester ID/Reserved, Reserved, Associated Port Numbers (1-N), and three more reserved fields.

PCI Express Extended Capability Header		
Designated Vendor Specific Header 1		
Capability Descriptor		Designated Vendor Specific Header 2
UCIe Link Capability		
UCIe Link Control ^e		
UCIe Link Status		
Error Notification Control		Link Event Notification Control
Register Locator 0 Low		
Register Locator 0 High		
...		
...		
Reserved		
Sideband Mailbox Index Low		
Sideband Mailbox Index High		
Sideband Mailbox Data Low		
Sideband Mailbox Data High		
Reserved	Sideband Mailbox Status	Sideband Mailbox Control
Requester ID/Reserved		
Reserved		
Associated Port Numbers (1-N)		
...		

- a. Applies to UCIe-EP, UCIe-USP, UCIe-Retimer.
- b. Applies to UCIe-EP, UCIe-USP when paired with a retimer.
- c. Applies to UCIe-RP.
- d. Applies to UCIe-DSP.
- e. Software writes to this register need to be broadcast to both D2D Adapter and PHY blocks because some registers could be implemented in either block or both blocks.

9.5.1.1 PCI Express Extended Capability Header (Offset 0h)

Set as follows for UCIe Link DVSEC. All bits in this register are RO.

Table 9-5. UCIe Link DVSEC - PCI Express Extended Capability Header

Field	Bit Location	Value	Comments
Capability ID	15:0	0023h	Value for PCI Express DVSEC capability
Revision ID	19:16	1h	Latest revision of the DVSEC capability
Next Capability Offset	31:20	Design Dependent	<p>For UCIe Link DVSEC in UiRB: Set to point to the next capability associated with this UCIe Link. In this revision of the spec, this field points to the MSI capability.</p> <p>The offset is in granularity of Bytes from the base address of UiRB. For example, if this is set to 100h, the next capability is located at offset of 100h from the base of UiRB.</p> <p>UCIe Link DVSEC in UiSRB: Set to point to the UCIe Link DVSEC capability of the next UCIe Link associated with a downstream port of the switch. The last UCIe Link DVSEC capability will set this offset to 0h indicating there are no more UCIe Links on downstream ports.</p> <p>The offset is in granularity of Bytes from the base address of UiSRB. For example, if this is set to 100h, the next DVSEC capability for the next Link is located at offset of 100h from the base of UiSRB.</p> <p>Retimer: Set to 0h</p> <p>Others: design dependent</p>

9.5.1.2 Designated Vendor Specific Header 1, 2 (Offsets 4h and 8h)

A few things to note on the various fields described in Table 9-6. DVSEC Revision ID field represents the version of the DVSEC structure. The DVSEC Revision ID is incremented whenever the structure is extended to add more functionality. Backward compatibility shall be maintained during this process. For all values of n, DVSEC Revision ID n+1 structure may extend Revision ID n by replacing fields that are marked as reserved in Revision ID n, but must not redefine the meaning of existing fields. Software that was written for a lower Revision ID may continue to operate on UCIe DVSEC structures with a higher Revision ID, but will not be able to take advantage of new functionality.

All bits in this register are RO.

Table 9-6. UCIe Link DVSEC - Designated Vendor Specific Header 1, 2

Register	Field	Bit Location	Value
Designated Vendor-Specific Header 1 (offset 04h)	DVSEC Vendor ID	15:0	D2DEh
	DVSEC Revision	19:16	0h
	Length	31:20	Device dependent. See Section 9.5.1.19 for some examples.
Designated Vendor-Specific Header 2 (offset 08h)	DVSEC ID	15:0	0h

9.5.1.3 Capability Descriptor (Offset Ah)

Provides a way for SW to discover which optional capabilities are implemented by the UCIe Port/Device.

Table 9-7. UCIe Link DVSEC - Capability Descriptor

Bit	Attribute	Description
2:0	RO	Number of Register locators 0h: 2 Register Locators 1h: 3 Register Locators 2h: 4 Register Locators ... 6h: 8 Register locators 7h: 1 Register Locator For this revision of UCIe, only values 0h, 1h, 2h and 7h are valid.
3	RO(RP/DSP), HWInit(EP/USP), RsvdP(Retimer)	Sideband mailbox Registers Present 0h: No sideband mailbox register set present in this capability 1h: Sideband mailbox register set present in this capability For RP/DSP, default value of this is 1. EP/USP must set this bit when they are paired with a retimer and must clear this bit in all other scenarios.
7:4	RO(DSP), RsvdP (Others)	Number of Switch DSPs associated with this UCIe Link Applies only to UCIe Link DVSEC in UI SRB. The specific 'port number' values of each Switch downstream port associated with this UCIe Link is called out in the Associated Port Number register(s) in this capability. 0h: 1 Port 1h: 2 ports ... Fh: 16 ports 'Port Number' is bits 31:24 of the PCIe Link capabilities register of the downstream port. For first generation of UCIe, only values 0h and 1h are legal.
15:8	RsvdP	Reserved

9.5.1.4 UCIe Link DVSEC - UCIe Link Capability (Offset Ch)

Basic characteristics of the UCIe Link are discovered by SW using this register.

Table 9-8. UCIe Link DVSEC - UCIe Link Capability (Sheet 1 of 2)

Bit	Attribute	Description
0	RO	Raw Format If set, indicates the Link can support Raw Format.
3:1	HWInit	Max Link Width 0h: x16 1h: x32 2h: x64 3h: x128 4h: x256 7h: x8 Others - Reserved
7:4	HWInit	Max Link Speeds 0h: 4 GT/s 1h: 8 GT/s 2h: 12 GT/s 3h: 16 GT/s 4h: 24 GT/s 5h: 32 GT/s Others: Reserved
8	RO (Retimer), RsvdP (others)	Retimer - Set by retimer to indicate it to SW
9	RsvdP (Retimer), RO (others)	Multi-protocol capable^a 0 - single stack capable 1 - multi-protocol capable Only 2 stacks max is possible
10	RO	Advanced Packaging 0 = Standard package mode for UCIe Link 1 = Advanced package mode for UCIe Link
11	RO	68B Flit Format for Streaming Protocol If set, indicates 68B Flit Format is supported for Streaming protocol. This is only set if at least one of the Protocol Layers is Streaming protocol.
12	RO	Standard 256B End Header Flit Format for Streaming Protocol If set, indicates Standard 256B End Header Flit Format is supported for Streaming protocol. This is only set if at least one of the Protocol Layers is Streaming protocol.
13	RO	Standard 256B Start Header Flit Format for Streaming Protocol If set, indicates Standard 256B Start Header Flit Format is supported for Streaming protocol. This is only set if at least one of the Protocol Layers is Streaming protocol.
14	RO	Latency-Optimized 256B Flit Format without Optional Bytes for Streaming Protocol If set, indicates Latency-Optimized 256B without Optional Bytes Flit Format is supported for Streaming protocol. This is only set if at least one of the Protocol Layers is Streaming protocol.
15	RO	Latency-Optimized 256B Flit Format with Optional Bytes for Streaming Protocol If set, indicates Latency-Optimized 256B with Optional Bytes Flit Format is supported for Streaming protocol. This is only set if at least one of the Protocol Layers is Streaming protocol.
16	RO	Enhanced Multi-protocol Capable 0 = Not capable of multi-protocol with different protocols 1 = Capable of multi-protocol with different protocols
17	RO	Standard Start Header Flit for PCIe Protocol If set, indicates Standard Start Header 256B Flit Format is supported for PCIe protocol. This is only set if at least one of the Protocol Layers is PCIe protocol.

Table 9-8. UCIe Link DVSEC - UCIe Link Capability (Sheet 2 of 2)

Bit	Attribute	Description
18	RO	Latency-Optimized Flit with Optional Bytes for PCIe Protocol If set, indicates that the Latency-Optimized Flit Format with Optional Bytes is supported for PCIe. This is only set if at least one of the Protocol Layers is PCIe protocol.
19	RO	'Runtime Link Testing Parity' Feature Error Signaling If set, design supports signaling errors detected during Runtime link testing with parity as Correctable errors. If cleared, this error signaling mechanism is not supported.
20	HWInit	APMW (Advanced Package Module Width) If set, indicates the Advanced Package Module size is x32 or a x64 module operating in x32 mode (decided at integration time). If reset, indicates x64 Advanced Package Module.
21	RO/RsvdP	x32 Width Support in x64 Module If set, indicates that a x64 Advanced Package Module can operate in x32 mode; otherwise, it cannot operate in x32 mode. For x32 Advanced Package Module, this bit is reserved.
22	HWInit	SPMW (Standard Package Module Width) If 1, indicates the Standard Package Module size is a x8 module, or a x16 module operating in x8 mode (decided at integration time). If 0, indicates x16 Standard Package Module.
23	RO	Sideband Performant Mode Operation (PMO) When set, indicates that the sideband supports performant mode operation. When cleared, performant mode operation is not supported.
31:24	RsvdP	Reserved

a. This bit was named and referred to as "Multi-stack" in r1.1 and prior revisions of the spec.

9.5.1.5 UCIe Link DVSEC - UCIe Link Control (Offset 10h)

Basic UCIe Link control bits are in this register.

Table 9-9. UCIe Link DVSEC - UCIe Link Control (Sheet 1 of 3)

Bit	Attribute	Description
0	RW (RP/DSP), HWInit (Others)	Raw Format Enable: If set, enables the Link to negotiate Raw Format during Link training. Default value of this is 0b for RP and firmware/SW sets this bit based on system usage scenario. Switch DSP can set the default via implementation-specific mechanisms such as straps/FW/etc., to account of system usage scenario (like UCIe retimer). This allows for the DSP Link to train up without Software intervention and be UCIe-unaware-OS compatible.
1	RW (RP/DSP), RO (EP/DSP), RsvdP (Retimer)	Multi-protocol enable^a: When set, multi-protocol training is enabled else not. Default is same as 'Multi-protocol Capable' bit in UCIe Link Capability register.
5:2	RW (RP/DSP), RsvdP (Others)	Target Link Width 0h: Reserved 1h: x8 2h: x16 3h: x32 4h: x64 5h: x128 6h: x256 Others are Reserved. Default is same as 'Max Link Width' field in UCIe Link Capability Register.

Table 9-9. UCIe Link DVSEC - UCIe Link Control (Sheet 2 of 3)

Bit	Attribute	Description
9:6	RW (RP/DSP), RsvdP (Others)	<p>Target Link Speed 0h: 4 GT/s 1h: 8 GT/s 2h: 12 GT/s 3h: 16 GT/s 4h: 24 GT/s 5h: 32 GT/s Others: Reserved Default is same as 'Max Link speed' field in UCIe Link Capability Register.</p>
10	RW, with auto clear (RP/DSP), RsvdP (Others)	<p>Start UCIe Link training - When set to 1, Link training starts with Link Control bits programmed in this register and with the protocol layer capabilities. Bit is automatically cleared when Link training completes with either success or error. The status register captures the final status of the Link training. Note that if the Link is up when this bit is set to 1 from 0, the Link will go through full training through Link Down state thus resetting everything beneath the Link. If Link Status (in UCIe Link Status register) is 0b and the link is already in training (i.e., the link training state machine is in between RESET and ACTIVE states), when this bit transitions from 0 to 1, link does not restart the training and this bit's transition from 0 to 1 is ignored. Primary usage intended for this bit is for initial Link training out of reset on the host side. Note: For downstream ports of a switch with UCIe, local HW/FW has to autonomously initiate Link training after a conventional reset, without waiting for higher level SW to start the training via this bit, to ensure backward compatibility. Default is 0.</p>
11	RW with auto clear (RP/DSP), RsvdP (Others)	<p>Retrain UCIe Link - When set to 1, Link that is already up (Link_status=up) will be retrained without going through Link Down state. SW can use this bit to potentially recover from Link errors. If the Link is down (Link_status=down) when this bit is set, there is no effect from this bit being set. SW should use the 'Start UCIe Link training' bit in case the Link is down. The Link_status bit in the status register can be read by software to determine whether to use this bit or not. Note that when retrain happens, the Link speed or width can change because of reliability reasons, and it will be captured through the appropriate status bit in the Link Status register. Bit is automatically cleared when Link retraining completes with either success or error (as reported via the appropriate status bits in the Link Status register) or if the Link retrain did not happen at all for the reason stated earlier. Default is 0.</p>
12	RW/RO	<p>Unused - Implementations are encouraged to implement this as an RO bit with a default value of 0. However, for backward compatibility, implementations are permitted to implement this as an RW bit with a default value of 1. Writes to this bit have no effect on link functionality.</p>
13	RW	<p>68B Flit Format for Streaming Protocol If set, enables 68B Flit Format advertisement if the corresponding capability is supported. Default is same as the '68B Flit Format for Streaming Protocol' bit in the UCIe Link Capability register.</p>
14	RW	<p>Standard 256B End Header Flit Format for Streaming Protocol If set, enables Standard 256B End Header Flit Format advertisement if the corresponding capability is supported. Default is same as the 'Standard 256B End Header Flit Format for Streaming Protocol' bit in the UCIe Link Capability register.</p>
15	RW	<p>Standard 256B Start Header Flit Format for Streaming Protocol If set, enables Standard 256B Start Header Flit Format advertisement if the corresponding capability is supported. Default is same as the 'Standard 256B Start Header Flit Format for Streaming Protocol' bit in the UCIe Link Capability register.</p>

Table 9-9. UCIe Link DVSEC - UCIe Link Control (Sheet 3 of 3)

Bit	Attribute	Description
16	RW	Latency -Optimized 256B Flit Format without Optional Bytes for Streaming Protocol If set, enables Latency-Optimized 256B Flit Format without Optional bytes advertisement if the corresponding capability is supported. Default is same as the 'Latency-Optimized 256B Flit Format without for Streaming Protocol' bit in the UCIe Link Capability register.
17	RW	Latency-Optimized 256B Flit Format with Optional Bytes for Streaming Protocol If set, enables Latency-Optimized 256B Flit Format with Optional bytes advertisement if the corresponding capability is supported. Default is same as the 'Latency-Optimized 256B Flit Format for Streaming Protocol' bit in the UCIe Link Capability register.
18	RW (RP/DSP), RO (EP/USP), RsvdP (Retimer)	Enhanced Multi-Protocol Enable When set, enhanced multi-protocol training is enabled else not. Enhanced Multi-Protocol permits 2 stacks with the same or different protocols. Default is same as 'Enhanced Multi-Protocol Capable' bit in UCIe Link Capability register.
19	RW	Standard Start Header Flit for PCIe Protocol If set, enables Standard Start Header 256B Flit Format for PCIe protocol. Default is same as 'Standard Start Header Flit for PCIe Protocol' bit in UCIe Link Capability register.
20	RW	Latency-Optimized Flit with Optional Bytes for PCIe Protocol If set, enables the Latency-Optimized Flit Format with Optional Byte for PCIe. Default is same as 'Latency-Optimized Flit with Optional Bytes for PCIe Protocol' bit in UCIe Link Capability register.
21	RW	Sideband Performant Mode Operation (PMO) When set, Sideband Performant Mode Operation is enabled for negotiation; otherwise, it is not. Default is the same as the Capability bit.
31:22	RsvdP	Reserved

a. This bit was named and referred to as "Multi-stack" in r1.1 and prior revisions of the spec.

9.5.1.6 UCIe Link DVSEC - UCIe Link Status (Offset 14h)

Basic UCIe Link status bits are in this register.

Table 9-10. UCIe Link DVSEC - UCIe Link Status (Sheet 1 of 3)

Bit	Attribute	Description
0	RO	Raw Format Enabled: If set, indicates the Adapter negotiated Raw Format operation with remote Link partner. This bit is only valid when Link Status bit in this register indicates 'Link Up'.
1	RsvdZ (Retimer), RO (Others)	Multi-protocol enabled^a: When set, multi-protocol training has been enabled with remote training partner. This bit is only valid when Link Status bit in this register indicates 'Link Up'.
2	RsvdZ (Retimer), RO (Others)	Enhanced Multi-protocol Enabled When set, multi-protocol training has been enabled with remote training partner. This bit is only valid when Link Status bit in this register indicates 'Link Up'.
3	RO	x32 Advanced Package Module Enabled When set, indicates that the Advanced Package operating module size is x32.
6:4	RsvdZ	Reserved

Table 9-10. UCIe Link DVSEC - UCIe Link Status (Sheet 2 of 3)

Bit	Attribute	Description
10:7	RO	<p>Link Width enabled 0h: x4 1h: x8 2h : x16 3h : x32 4h : x64 5h : x128 6h : x256</p> <p>This has meaning only when Link status bit shows Link is up.</p>
14:11	RO	<p>Link Speed enabled 0h: 4GT/s 1h: 8GT/s 2h: 12GT/s 3h: 16GT/s 4h: 24GT/s 5h: 32GT/s Others: Reserved</p> <p>This field has meaning only when Link status field shows Link is up</p>
15	RO	<p>Link Status 0 - Link is down. 1 - Link is up</p> <p>This bit indicates the status of the mainband.</p> <p>Transitioning a Link from down to up requires a full Link training, which can be achieved using one of these methods:</p> <ul style="list-style-type: none"> Start Link training via the bits in the UCIe Link Control register of the upstream device Using the protocol layer reset bit associated with the Link, like the SBR bit in the BCTL register of the RP P2P space Using the protocol layer Link Disable bit associated with the Link, like the Link Disable bit in the Link CTL register of the PCIe capability register in the RP P2P space, and then releasing the disable. <p>Notes: If the Link is actively retraining, this bit reflects a value of 1.</p> <p>This bit is a consolidated status of the RDI and FDI (i.e., if both the RDI and FDI are up, then this bit is set to 1; otherwise, this bit is cleared to 0).</p> <p>In multi-stack implementations, this bit is a consolidated status of the RDI and any of the FDIs (i.e., if RDI is up and any of the FDIs is up, then this bit is set to 1; otherwise, this bit is cleared to 0).</p>
16	RO	<p>Link Training/Retraining 1b - Currently Link is training or retraining 0b - Link is not training or retraining</p>
17	RW1C (RP/DSP), RsvdZ (Others)	<p>Link Status changed 1b - Link either transitioned from up to down or down to up. 0b - No Link status change since the last time SW cleared this bit</p>
18	RW1C (RP/DSP), RsvdZ (Others)	<p>HW autonomous BW changed UCIe autonomously changed the Link width or speed to correct Link reliability related issues.</p>
19	RW1CS	<p>Detected UCIe Link correctable error Further details of specific type of correctable error is found in Table 9-30 register.</p>
20	RW1CS	<p>Detected UCIe Link Uncorrectable Non-fatal error Further details of specific type of Uncorrectable error is found in Table 9-27 register.</p>

Table 9-10. UCIe Link DVSEC - UCIe Link Status (Sheet 3 of 3)

Bit	Attribute	Description
21	RW1CS	Detected UCIe Link Uncorrectable Fatal error Further details of specific type of Uncorrectable error is found in Table 9-27 register.
25:22	RO	Flit Format Status This field and the Flit Format field in the Header Log 2 register in the D2D/PHY register block (see Section 9.5.3.8) are mirror copies. This field indicates the negotiated Flit Format. This field is only valid when Link Status bit in this register indicates 'Link Up'.
26	RO	Sideband Performant Mode Operation (PMO) When set, Sideband Performant Mode Operation was successfully negotiated and is operational. When cleared, legacy mode sideband operation is active. Sideband Performant Mode is not operational. This bit has meaning only when either Link status indicates link is up (in UCIe Link Status register of UCIe Link DVSEC capability) or management port capability indicates Port Status as 'Link Not Up' (see Table 8-12).
31:27	RsvdZ	Reserved

a. This bit was named and referred to as "Multi-stack" in r1.1 and prior revisions of the spec.

9.5.1.7 UCIe Link DVSEC - Link Event Notification Control (Offset 18h)

Link event notification related controls are in this register.

Table 9-11. UCIe Link DVSEC - Link Event Notification Control

Bit	Attribute	Description
0	RW(RP/DSP), RsvdP (Others)	'Link Status changed' UCIe Link Event Interrupt enable 0: Reporting of this event via interrupt is not enabled 1: Reporting of this event via interrupt is enabled. Default is 0
1	RW(RP/DSP), RsvdP (Others)	'HW autonomous BW changed' UCIe Link Event Interrupt enable 0: Reporting of this event via interrupt is not enabled 1: Reporting of this event via interrupt is enabled Default is 0
10:2	RsvdP	Reserved
15:11	RO(RP/DSP), RsvdP(Others)	Link Event Notification Interrupt number This field indicates which MSI vector (for host UCIe Links), or MSI/MSI-X vector (for switch DSP UCIe Links) is used for the interrupt message generated in association with the events that are controlled via this register. For MSI, the value in this field indicates the offset between the base Message Data and the interrupt message that is generated. Hardware is required to update this field so that it is correct if the number of MSI Messages assigned to the Function changes when software writes to the Multiple Message Enable field in the Message Control Register for MSI. For first generation of UCIe, maximum 2 interrupt vectors could be requested for UCIe related functionality and the 'Link event' is one of them. For MSI-X (applicable only for interrupts from Switch DSPs with UCIe Links), the value in this field indicates which MSI-X Table entry is used to generate the interrupt message. The entry must be one of the first 32 entries even if the Function implements more than 32 entries. For a given MSI-X implementation, the entry must remain constant. For UCIe related interrupts, a switch should request its interrupt requirements from either MSI or MSI-X capability but not both.

9.5.1.8 UCIe Link DVSEC - Error Notification Control (Offset 1Ah)

Link error notification related controls are in this register.

Note: This register only controls the propagation of the error condition and it has no impact on the setting of the appropriate status bits in the Link Status register, when the relevant error happens.

Table 9-12. UCIe Link DVSEC - Error Notification Control (Sheet 1 of 3)

Bit	Attribute	Description
0	RW(RP/DSP), RsvdP (Others)	<p>'Correctable error detected' protocol layer based reporting enable</p> <p>0: Reporting of this error via protocol layer mechanism is not enabled 1: Reporting of this error via protocol layer mechanism is enabled</p> <p>Default is 0</p> <p>When enabled, the reported PCIe/CXL protocol layer correctable error type is 'Correctable internal error'.</p> <p>This bit is applicable for only RP/DSP.</p>
1	RW	<p>'Correctable error detected' UCIe Link Error Interrupt enable RP/DSP</p> <p>0: Reporting of this error via UCIe Link Error interrupt is not enabled 1: Reporting of this error via UCIe Link Error interrupt is enabled</p> <p>EP/USP</p> <p>0: Reporting of this error via sideband error message is not enabled 1: Reporting of this error via sideband error message is enabled</p> <p>Note that in the case of EP/USP connected to a retimer, their sideband error message targets the retimer and how the retimer sends it across to the partner retimer is vendor specific.</p> <p>Retimer connected to RP/DSP</p> <p>0: Reporting of this error via sideband error message to RP/DSP is not enabled 1: Reporting of this error via sideband error message to RP/DSP is enabled</p> <p>Retimer connected to EP/USP</p> <p>0: Reporting of this error to the partner retimer is disabled. 1: Reporting of this error to the partner retimer is enabled. The specific mechanism for reporting the error to the partner retimer is vendor-specific.</p> <p>Default is 0</p>
2	RW(RP/DSP), RsvdP (Others)	<p>'Uncorrectable non-fatal error detected' protocol layer based reporting enable</p> <p>0: Reporting of this error via protocol layer mechanism is not enabled 1: Reporting of this error via protocol layer mechanism is enabled</p> <p>Default is 0</p> <p>This bit is applicable for only RP/DSP.</p>

Table 9-12. UCIe Link DVSEC - Error Notification Control (Sheet 2 of 3)

Bit	Attribute	Description
3	RW	<p>'Uncorrectable non-fatal error detected' UCIe Link Error Interrupt enable</p> <p>RP/DSP 0: Reporting of this error via UCIe Link Error interrupt is not enabled 1: Reporting of this error via UCIe Link Error interrupt is enabled</p> <p>EP/USP 0: Reporting of this error via sideband error message is not enabled 1: Reporting of this error via sideband error message is enabled</p> <p>Note that in the case of EP/USP connected to a retimer, their sideband error message targets the retimer and how the retimer sends it across to the partner retimer is vendor specific.</p> <p>Retimer connected to RP/DSP 0: Reporting of this error via sideband error message to RP/DSP is not enabled 1: Reporting of this error via sideband error message to RP/DSP is enabled</p> <p>Retimer connected to EP/USP 0: Reporting of this error to the partner retimer is disabled. 1: Reporting of this error to the partner retimer is enabled. The specific mechanism for reporting the error to the partner retimer is vendor specific.</p> <p>Default is 0</p>
4	RW (RP/DSP), RsvdP (Others)	<p>'Uncorrectable fatal error detected' protocol layer based reporting enable</p> <p>0: Reporting of this error via protocol layer mechanism is not enabled 1: Reporting of this error via protocol layer mechanism is enabled Default is 0 When enabled, the reported PCIe/CXL protocol layer uncorrectable error type is 'Uncorrectable internal error' This bit is applicable for only RP/DSP.</p>

Table 9-12. UCIe Link DVSEC - Error Notification Control (Sheet 3 of 3)

Bit	Attribute	Description
5	RW	<p>'Uncorrectable fatal error detected' UCIe Link Error Interrupt enable RP/DSP 0: Reporting of this error via UCIe Link Error interrupt is not enabled 1: Reporting of this error via UCIe Link Error interrupt is enabled</p> <p>EP/USP 0: Reporting of this error via sideband error message is not enabled 1: Reporting of this error via sideband error message is enabled</p> <p>Note that in the case of EP/USP connected to a retimer, their sideband error message targets the retimer and how the retimer sends it across to the partner retimer is vendor specific.</p> <p>Retimer connected to RP/DSP 0: Reporting of this error via sideband error message to RP/DSP is not enabled 1: Reporting of this error via sideband error message to RP/DSP is enabled</p> <p>Retimer connected to EP/USP 0: Reporting of this error to the partner retimer is disabled. 1: Reporting of this error to the partner retimer is enabled. The specific mechanism for reporting the error to the partner retimer is vendor specific.</p> <p>Default is 0</p>
10:6	RsvdP	Reserved
15:11	RW/RO	<p>Link Error Notification Interrupt number</p> <p>This field indicates which MSI vector (for host UCIe Links), or MSI/MSI-X vector (for switch DSP UCIe Links) is used for the interrupt message generated in association with the events that are controlled via this register.</p> <p>For MSI, the value in this field indicates the offset between the base Message Data and the interrupt message that is generated. Hardware is required to update this field so that it is correct if the number of MSI Messages assigned to the Function changes when software writes to the Multiple Message Enable field in the Message Control Register for MSI. For first generation of UCIe, maximum 2 interrupt vectors could be requested for UCIe related functionality and the 'Error' is one of them.</p> <p>For MSI-X (applicable only for interrupts from Switch DSPs with UCIe Links), the value in this field indicates which MSI-X Table entry is used to generate the interrupt message. The entry must be one of the first 32 entries even if the Function implements more than 32 entries. For a given MSI-X implementation, the entry must remain constant.</p> <p>For UCIe related interrupts, a switch should request its interrupt requirements from either MSI or MSI-X capability but not both.</p> <p>It is strongly recommended that this field be implemented as RO but for backward compatibility reasons, it is also permitted to be implemented as RW. This field has no meaning for Switch USP and EP.</p>

9.5.1.9 UCIe Link DVSEC - Register Locator 0, 1, 2, 3 Low (Offset 1Ch and when Register Locators 1, 2, 3 are present Offsets 24h, 2Ch, and 34h respectively)

The starting address of the MMIO-mapped register blocks for D2D/PHY, Compliance/Test and Implementation-specifics are located by SW via these registers.

Note: All register blocks start with a header section that indicates the size of the block in multiples of 4 KB.

Table 9-13. UCIe Link DVSEC - Register Locator 0, 1, 2, 3 Low

Bit	Attribute	Description
2:0	RO	<p>Register BIR For UCIe DVSEC capability in host UiRB, Switch UiSRB and in UCIe Retimer, this field is reserved. For others, its defined as follows: Indicates which one of a Dev0/Fn0 Base Address Registers, located beginning at 10h in Configuration Space, or entry in the Enhanced Allocation capability with a matching BAR Equivalent Indicator (BEI), is used to map the UCIe Register blocks into Memory Space. Defined encodings are:</p> <ul style="list-style-type: none"> • 0 Base Address Register 10h • 1 Base Address Register 14h • 2 Base Address Register 18h • 3 Base Address Register 1Ch • 4 Base Address Register 20h • 5 Base Address Register 24h <p>All other Reserved. The Registers block must be wholly contained within the specified BAR. For a 64-bit Base Address Register, the Register BIR indicates the lower DWORD.</p>
6:3	RO	<p>Register Block Identifier</p> <ul style="list-style-type: none"> • Identifies the type of UCIe register blocks. Defined encodings are: 0h UCIe D2D/PHY Register Block • 1h UCIe Test/Compliance Register Block • 2h D2D Adapter Implementation specific register block • 3h PHY Implementation specific register block • All other encodings are reserved <p>The same register block identifier value cannot be repeated in multiple Register Locator entries.</p>
11:7	RsvdP	Reserved
31:12	RO	<p>Register Block Offset Addr[31:12] of the 4-KB aligned offset from the starting address of the Dev0/Fn0 BAR pointed to by the Register BIR field (for EP, Switch USP) or from the start of UiRB/UiSRB region (for hosts/Switch). This field is reserved for retimers.</p>

9.5.1.10 UCIe Link DVSEC - Register Locator 0, 1, 2, 3 High (Offset 20h and when Register Locators 1, 2, 3 Are Present Offsets 28h, 30h, and 38h respectively)

Addr[63:32] of the starting address of the MMIO-mapped register blocks for D2D/PHY, Compliance/Test and Implementation-specifics are located by SW via these registers.

Note: All register blocks start with a header section that indicates the size of the block in multiples of 4 KB.

Table 9-14. UCIe Link DVSEC - Register Locator 0, 1, 2, 3 High

Bit	Attribute	Description
63:32	RO	Register Block Offset Addr[63:32] of the 4-KB aligned offset from the starting address of the Dev0/Fn0 BAR pointed to by the Register BIR field (for EP, Switch USP) or from the start of UiRB/UiSRB region (for hosts/Switch). This field is reserved for retimers.

9.5.1.11 UCIe Link DVSEC - Sideband Mailbox Index Low (Offset is design dependent)

Mailbox registers are to be implemented by all hosts with UCIe Links. Switches with downstream UCIe Links and EP/USP, when paired with UCIe Retimer, should also implement this register. Note that accesses to mailbox are inherently non-atomic in nature and hence it is up to higher-level software to coordinate access to any mailbox-related register so that one agent does not step on another agent using the mailbox mechanism. Those mechanisms for software coordination are beyond the scope of this specification.

Table 9-15. UCIe Link DVSEC - Sideband Mailbox Index Low

Bit	Attribute	Description
4:0	RW	Opcode 00000b 32b Memory Read 00001b 32b Memory Write 00100b 32b Configuration Read 00101b 32b Configuration Write 01000b 64b Memory Read 01001b 64b Memory Write 01100b 64b Configuration Read 01101b 64b Configuration Write OthersReserved Default 00100
12:5	RW	BE[7:0] Default Fh
31:13	RW	Addr[18:0] of Sideband Accesses Format for this field is as defined in the sideband interface definition in Chapter 7.0 . Note: The address offset defined as part of this address field is DWORD aligned for 32bit accesses and QWORD aligned for 64bit accesses. Default is 0.

9.5.1.12 UCIe Link DVSEC - Sideband Mailbox Index High (Offset is design dependent)

Mailbox registers are to be implemented by all hosts with UCIe Links. Switches with downstream UCIe Links and EP/USP, when paired with UCIe Retimer, should also implement this register. Note that accesses to mailbox are inherently non-atomic in nature and hence it is up to higher-level software to coordinate access to any mailbox-related register so that one agent does not step on another agent using the mailbox mechanism. Those mechanisms for software coordination are beyond the scope of this specification.

Table 9-16. UCIe Link DVSEC - Sideband Mailbox Index High

Bit	Attribute	Description
4:0	RW	Addr[23:19] of Sideband Accesses Format for this field is as defined in the sideband interface definition in Chapter 7.0. Default is 0.
31:5	RsvdP	Reserved

9.5.1.13 UCIe Link DVSEC - Sideband Mailbox Data Low (Offset is design dependent)

Table 9-17. UCIe Link DVSEC - Sideband Mailbox Data Low

Bit	Attribute	Description
31:0	RW	For sideband write opcodes, this carries the write data [31:0] to the destination. For sideband read opcodes, this carries the data read from the destination when the Write/Read Trigger bit in the Mailbox Control register is cleared, after it was initially set. This field's value is undefined until the Write/Read trigger bit is cleared on reads.

9.5.1.14 UCIe Link DVSEC - Sideband Mailbox Data High (Offset is design dependent)

Table 9-18. UCIe Link DVSEC - Sideband Mailbox Data High

Bit	Attribute	Description
31:0	RW	For sideband write opcodes, this carries the write data [63:32] to the destination. For sideband read opcodes, this carries the data read from the destination when the Write/Read Trigger bit in the Mailbox Control register is cleared, after it was initially set. This field's value is undefined until the Write/Read trigger bit is cleared on reads. For 32b Writes/Reads, this register does not carry valid data.

9.5.1.15 UCIe Link DVSEC - Sideband Mailbox Control (Offset is design dependent)

Table 9-19. UCIe Link DVSEC - Sideband Mailbox Control

Bit	Attribute	Description
0	RW, with auto clear	Write/Read trigger: When this bit is written to a 1 from a value of 0, the mailbox generates traffic on the sideband interface, using the contents of the Mailbox Header and Data registers. This bit automatically clears when the write or read access triggered by this bit being set, is complete on the sideband bus. SW can poll this bit to know when the write/read has actually completed at the destination. It can then go read the Mailbox data register for the read data.
7:1	RsvdP	Reserved

9.5.1.16 UCIe Link DVSEC - Sideband Mailbox Status (Offset is design dependent)

Table 9-20. UCIe Link DVSEC - Sideband Mailbox Status

Bit	Attribute	Description
1:0	RW1C(RP/DSP), RW1C(EP/USP), when implemented	Write/Read status 00b: CA received 01b: UR received 10b: Reserved 11b: Success This bit has valid value only when the Write/Read Trigger bit is cleared from being a 1 prior to it.
7:2	RsvdZ	Reserved

9.5.1.17 UCIe Link DVSEC - Requester ID (Offset is design dependent)

Table 9-21. UCIe Link DVSEC - Requester ID

Bit	Attribute	Description
23:0	RW(RP)/RsvdP (Others)	Applicable only for host side UCIe Links. Segment No: Bus No: Dev No: Fn No for MSIs triggered on behalf of the associated UCIe Link Note: For MSIs issued on behalf of UCIe Links on downstream ports of switches, the Switch USP BDF is used. UCIe Link DVSEC capabilities in UISRB implement this as RO 0.
31:24	RsvdP	Reserved

9.5.1.18 UCIe Link DVSEC - Associated Port Numbers (Offset is design dependent)

These registers apply only to UCIe Link DVSEC capabilities present in UiSRB.

Table 9-22. UCIe Link DVSEC - Associated Port Numbers

Bit	Attribute	Description
7:0	RO	Port Number 1 - 'Port number' of the 1st switch DSP associated with this UCIe. This value is from the Link Capabilities register of that switch DSP.
15:8	RO	Port Number 2 - 'Port number' of the 2nd switch DSP associated with this UCIe, if any. If there is no 2nd switch DSP associated with this UCIe Link, this field is treated as reserved and should not be included as part of the "length" field of the 'Designated Vendor specific Header 1' register and SW should not consider this as part of the DVSEC capability. Note: Only a maximum of two Port numbers can be associated with a UCIe Link in the current revision of the specification.

9.5.1.19 Examples of setting the Length field in DVSEC for various Scenarios

Example#1: UCIe EP supporting 2 Register Locators and not associated with a UCIe-Retimer, would set the length field in DVSEC capability to indicate 48B.

Example#2: Host UiRB supporting 3 register locators would set the length to indicate 84B.

Example#3: Switch UiSRB supporting 3 register locators and associated with just 1 DSP port to a UCIe Link, would set the length to indicate 85B.

9.5.2 UCIe Switch Register Block (UiSRB) DVSEC Capability

This capability can only be present in the config space of the upstream port of a Switch. There can be multiple of these in the same USP config space.

9.5.2.1 PCI Express Extended Capability Header (Offset 0h)

Set as follows for UCIe Switch Register Block DVSEC. All bits in this register are RO.

Table 9-23. UiSRB DVSEC - PCI Express Extended Capability Header

Field	Bit Location	Value	Comments
Capability ID	15:0	0023h	Value for PCI Express DVSEC capability
Revision ID	19:16	1h	Latest revision of the DVSEC capability
Next Capability Offset	31:20	Design Dependent	

9.5.2.2 Designated Vendor Specific Header 1, 2 (Offsets 4h and 8h)

A few things to note on the various fields described in Table 9-6. DVSEC Revision ID field represents the version of the DVSEC structure. The DVSEC Revision ID is incremented whenever the structure is extended to add more functionality. Backward compatibility shall be maintained during this process. For all values of n, DVSEC Revision ID n+1 structure may extend Revision ID n by replacing fields that are marked as reserved in Revision ID n, but must not redefine the meaning of existing fields. Software that was written for a lower Revision ID may continue to operate on UCIe DVSEC structures with a higher Revision ID, but will not be able to take advantage of new functionality.

All bits in this register are RO.

Table 9-24. UiSRB DVSEC - Designated Vendor Specific Header 1, 2

Register	Field	Bit Location	Value
Designated Vendor-Specific Header 1 (offset 04h)	DVSEC Vendor ID	15:0	D2DEh
	DVSEC Revision	19:16	0h
	Length	31:20	14h
Designated Vendor-Specific Header 2 (offset 08h)	DVSEC ID	15:0	1h

9.5.2.3 UCIe Switch Register Block (UiSRB) Base Address (Offset Ch)

All bits in this register are RO.

Table 9-25. UiSRB DVSEC - UiSRB Base Address

Bit	Attributes	Description
0	RO	Register BIR Indicates which one of a Switch USP Function's Base Address Registers, located beginning at 10h in Configuration Space, or entry in the Enhanced Allocation capability with a matching BAR Equivalent Indicator (BEI), is used to locate the UCIe Switch Register Block. Defined encodings are: <ul style="list-style-type: none"> • 0 Base Address Register 10h • 1 Base Address Register 14h • All other Reserved. The Registers block must be wholly contained within the specified BAR. For a 64-bit Base Address Register, the Register BIR indicates the lower DWORD.
11:1	RsvdP	Reserved
63:12	RO	Register Block Offset A 4-KB-aligned offset from the starting address of the Switch USP BAR indicated by the Register BIR field. The BAR value + Offset indicated in this register is where the UCIe Switch Register Block (UiSRB) starts. Ex: If this register is 100, UiSRB starts at the <64-bit BAR value + 100000h>

9.5.3 D2D/PHY Register Block

These registers occupy 8 KB of register space. The first 4 KB are for the D2D Adapter, and the next 4 KB are for the Physical Layer. In the PHY register block, extended capabilities start at Offset 200h. If an implementation does not support any extended capabilities, it must implement a NULL capability at Offset 200h (which implements 0h for the DWORD at that offset). The D2D Adapter registers are enumerated below. The location of these registers in the system MMIO region is as described in Section 9.3.

9.5.3.1 UCIe Register Block Header

Table 9-26. D2D/PHY Register Block - UCIe Register Block Header (Offset 0h)

Bit	Attributes	Description
15:0	RO	Vendor ID Default is set to Vendor ID assigned for UCIe Consortium - D2DEh
31:16	RO	Vendor ID Register Block Set to 0h to indicate D2D/PHY register block
35:32	RO	Vendor Register Block Version Set to 0h
63:36	RsvdP	Reserved
95:64	RO	Vendor Register Block Length - The number of bytes in the register block including the UCIe Register block header. Default is 2000h.
127:96	RsvdP	Reserved

9.5.3.2 Uncorrectable Error Status Register (Offset 10h)

Table 9-27. Uncorrectable Error Status Register (Sheet 1 of 2)

Bit	Attribute	Description
0	RW1CS	Adapter Timeout: Set to 1b by hardware if greater than 8ms has elapsed for Adapter handshakes with its remote Link partner. The Header Log 2 register captures the reason for a timeout. This error will bring the main Link Down. Default Value is 0b.
1	RW1CS	Receiver Overflow: Set to 1b by hardware if Receiver overflow errors are detected. The Header Log 2 register captures the encoding to indicate the type of Receiver overflow. This error will bring the Link Down. Default Value is 0b.
2	RW1CS	Internal Error: Set to 1b by hardware if an internal Data path error is detected or if LinkError state was detected on the RDI. Examples of such errors include (but not limited to) uncorrectable error correcting code (ECC) error in the Retry buffer, sideband parity errors etc. This error will bring the Link Down. It includes fatal error indicated by the Physical Layer that brought the Link Down. Default Value is 0b.
3	RW1CS (RP/DSP/ Retimer), RsvdZ (Others)	Sideband Fatal Error Message received: Set to 1b by hardware if the Adapter received a Fatal {ErrMsg} sideband message. Default Value is 0b.

Table 9-27. Uncorrectable Error Status Register (Sheet 2 of 2)

Bit	Attribute	Description
4	RW1CS(RP/DSP/ Retimer), RsvdZ(Others)	Sideband Non-Fatal Error Message received: Set to 1b by hardware if the Adapter received a Non-Fatal {ErrMsg} sideband message. Default Value is 0b.
5	RW1CS	Invalid Parameter Exchange: Set to 1b if the Adapter was not able to determine a valid protocol or Flit Format for operation.
31:6	RsvdZ	Reserved

9.5.3.3 Uncorrectable Error Mask Register (Offset 14h)

The Uncorrectable Error Mask Register controls reporting of individual errors. When a bit is 1b in this register, the corresponding error status bit in the Uncorrectable Error Status register is not forwarded to the Protocol Layer for escalation/signaling but it does not impact error logging in the "First Fatal Error Indicator" field in the Header Log 2 register.

Table 9-28. Uncorrectable Error Mask Register

Bit	Attribute	Description
0	RWS	Adapter Timeout Mask Default Value is 1b.
1	RWS	Receiver Overflow Mask Default Value is 1b.
2	RWS	Internal Error Mask Default Value is 1b.
3	RWS	Sideband Fatal Error Message received Mask Default Value is 1b.
4	RWS	Sideband Non-Fatal Error Message received Mask Default Value is 1b.
5	RWS	Invalid Parameter Exchange Mask Default Value is 1b.
31:6	RsvdP	Reserved

9.5.3.4 Uncorrectable Error Severity Register (Offset 18h)

The Uncorrectable Error Severity register controls whether an individual error is reported as a Non-fatal or Fatal error. An error is reported as a fatal uncorrectable error when the corresponding bit in the severity register is 1b. If the bit is 0b, the corresponding error is reported as a non-fatal uncorrectable error.

Table 9-29. Uncorrectable Error Severity Register

Bit	Attribute	Description
0	RWS	Adapter Timeout Severity Default Value is 1b.
1	RWS	Receiver Overflow Severity Default Value is 1b.
2	RWS	Internal Error Severity Default Value is 1b.
3	RWS	Sideband Fatal Error Message received Severity Default Value is 1b.
4	RWS	Sideband Non-Fatal Error Message received Severity Default Value is 0b.
5	RWS	Invalid Parameter Exchange Severity Default Value is 1b
31:6	RsVdP	Reserved

9.5.3.5 Correctable Error Status Register (Offset 1Ch)

Table 9-30. Correctable Error Status Register

Bit	Attribute	Description
0	RW1CS	CRC Error Detected: Set to 1b by hardware if the Adapter detected a CRC Error when Adapter Retry was negotiated with remote Link partner. Default Value is 0b.
1	RW1CS	Adapter LSM transition to Retrain: Set to 1b by hardware if the Adapter LSM transitioned to Retrain state. Default Value is 0b.
2	RW1CS	Correctable Internal Error: Set to 1b by hardware if an internal correctable Data path error is detected. Examples of such errors include (but are not limited to) correctable error correcting code (ECC) error in the Retry buffer, Physical Layer indicated correctable error on RDI, etc. Default Value is 0b.
3	RW1CS (RP/DSP/ Retimer), RsVdZ (Others)	Sideband Correctable Error Message received: Set to 1b by hardware if the Adapter received a Correctable {ErrMsg} sideband message with Device origin encoding in the message information. Default Value is 0b.
4	RW1CS	'Runtime Link Testing Parity' Error
31:5	RsVdZ	Reserved

9.5.3.6 Correctable Error Mask Register (Offset 20h)

The Correctable Error Mask Register controls the reporting of individual errors. When a bit is 1b in this register, setting of the corresponding error status bit is not forwarded to the Protocol Layer for escalation/signaling.

Table 9-31. Correctable Error Mask Register

Bit	Attribute	Description
0	RWS	CRC Error Detected Mask Default Value is 1b.
1	RWS	Adapter LSM transition to Retrain Mask Default Value is 1b.
2	RWS	Correctable Internal Error Mask Default Value is 1b.
3	RWS	Device Correctable Error Message received Mask Default Value is 1b.
4	RWS	'Runtime Link Testing Parity' Error Mask Default Value is 1b.
31:5	RsvdP	Reserved

9.5.3.7 Header Log 1 Register (Offset 24h)

This register is used to log the header on sideband register accesses that receive UR/CA error status.

Table 9-32. Header Log 1 Register

Bit	Attribute	Description
63:0	ROS	Header Log 1: This logs the header for the sideband mailbox register access that received a completion with Completer Abort status or received a completion with Unsupported Request status. Note that register accesses that time out are not required to be logged at the requester. If the Write/Read Status field in the 'Sideband Mailbox Status' register indicates 'Success' or the Write/Read trigger bit in the Sideband Mailbox Control register is set to 1, this field's value is undefined. This register is rearmed for logging new errors every time the Write/Read Trigger bit in the Mailbox Control register sees a 0-to-1 transition. Default Value is 0.

9.5.3.8 Header Log 2 Register (Offset 2Ch)

This register is used to log syndrome of various sideband and mainband errors and specific status logging on link training.

Table 9-33. Header Log 2 Register (Sheet 1 of 2)

Bit	Attribute	Description
3:0	ROS	<p>Adapter Timeout encoding: Captures the reason for the first Adapter Timeout that was logged in Uncorrectable Error Status. Default Value is 0000b. The encodings are interpreted as follows:</p> <ul style="list-style-type: none"> 0001b: Parameter Exchange flow timed out 0010b: Adapter LSM request to remote Link partner did not receive a response after 8 ms. Bits [9:7] capture the specific state request that did not receive a response. Bit 10 of this register captures which Adapter LSM timed out. 0011b: Adapter LSM transition to Active timeout. This is recorded in case the Adapter never received Active Request from remote Link partner for 8 ms after sending an Active Request on sideband even though it received an Active Response. Bit 10 of this register captures which Adapter LSM timed out. 0100b: Retry Timeout - no Ack or Nak received after 8 ms, when Retry was enabled. Timeout counter is only incremented while RDI is in Active and Adapter's Retry buffer is not empty. 0101b: Local sideband access timeout 0110b: Retimer credit return timeout - no Retimer credit received for greater than 8 ms if one or more Retimer credits have been consumed by the Adapter. This timer is only counting during Active state. If RDI moves to Retrain, this timer must be Reset since the Retimer credits are also Reset. 0111b: Remote Register Access timeout. This is triggered when if the Adapter has observed N timeouts for Register Accesses where N is \geq register access timeout threshold. other encodings are reserved. <p>If the Adapter Timeout status bit is cleared in the 'Uncorrectable Error Status' register, this field's value is undefined.</p>
6:4	ROS	<p>Receiver Overflow encoding: Captures the encoding for the first Receiver overflow error that occurred. Default value is 000b. The encodings are interpreted as follows:</p> <ul style="list-style-type: none"> 001b: Transmitter Retry Buffer overflow 010b: Retimer Receiver Buffer overflow 011b: FDI sideband buffer overflow 100b: RDI sideband buffer overflow other encodings are reserved. <p>If the Receiver overflow status bit is cleared in the 'Uncorrectable Error Status' register, this field's value is undefined.</p>
9:7	ROS	<p>Adapter LSM response type</p> <ul style="list-style-type: none"> 001b: Active 010b: L1 011b: L2 100b: LinkReset 101b: Disable Other encodings are reserved <p>If the Adapter Timeout status bit is cleared in the 'Uncorrectable Error Status' register, this field's value is undefined.</p>
10	ROS	<p>Adapter LSM id</p> <ul style="list-style-type: none"> 0b : Adapter LSM 0 timed out 1b : Adapter LSM 1 timed out
12:11	RsvdZ	Reserved

Table 9-33. Header Log 2 Register (Sheet 2 of 2)

Bit	Attribute	Description
13	RO	Parameter Exchange Successful : Hardware updates this bit to 1b after successful Parameter exchange with remote Link partner, on every link training.
17:14	ROS	<p>Flit Format: This field logs the negotiated Flit Format, it is the current snapshot of the format the Adapter is informing to the Protocol Layer. See Chapter 3.0 for the definitions of these formats. The encodings are:</p> <ul style="list-style-type: none"> 0001b - <i>Format 1</i> 0010b - <i>Format 2</i> 0011b - <i>Format 3</i> 0100b - <i>Format 4</i> 0101b - <i>Format 5</i> 0110b - <i>Format 6</i> <p>Other encodings are Reserved</p>
22:18	ROS	<p>First Fatal Error Indicator: 5-bit encoding that indicates which bit of Uncorrectable Error Status errors was logged first. The value of this field has no meaning if the corresponding status bit is cleared. The encoding of this field is as follows:</p> <ul style="list-style-type: none"> 00h if the error corresponding to Uncorrectable Error Status register[0] is the first fatal error. 01h if the error corresponding to Uncorrectable Error Status register[1] is the first fatal error. ... <p>Because reserved bits may be repurposed in future versions of the specification, software might observe that this field points to a reserved bit (from its perspective) in the Uncorrectable Error Status register. This can happen when an older version of Software is run on newer hardware. Software must be aware that it still needs to clear the Status register bit if it desires to allow for continued error logging. How SW handles error status bits it does not understand is beyond the scope of the specification.</p> <p>Once set, the value of this field does not change until SW clears the corresponding Uncorrectable Error Status register bit. When SW clears the corresponding status bit, HW is rearmed to capture subsequent first fatal errors.</p> <p>Note that because of an inherent race condition between HW setting a new status bit and SW clearing an older status bit, SW must be aware that this field might not always indicate the first error amongst all the errors logged in the Uncorrectable Error Status register. For example, if the Uncorrectable Error Status bit 0 was set first by HW and in the time SW reads the status and cleared it, bit 1 in the Status register was set. So, after SW clears bit 0 if error corresponding to bit 0 recurs, it will be captured as the next first error even though the error corresponding to bit 1 occurred earlier. If multiple errors are encountered simultaneously, which error is logged as the First Fatal Error is implementation-dependent.</p>
31:23	RsvdZ	Reserved

9.5.3.9 Error and Link Testing Control Register (Offset 30h)

Table 9-34. Error and Link Testing Control Register (Sheet 1 of 2)

Bit	Attribute	Description
3:0	RW	Remote Register Access Threshold: Indicates the number of consecutive timeouts for remote register accesses that must occur before the Register Access timeout is logged and the error escalated to a Link_Status=Down condition. Default Value is 0100b.
4	RW	Runtime Link Testing Tx Enable: Software writes to this bit to enable Parity byte injections in the data stream as described in Section 3.9 . Runtime Link Rx Enable must be set to 1b for remote Link Partner for successful enabling of this mode. Default Value is 0b.
5	RW	Runtime Link Testing Rx Enable: Software writes to this bit to enable Parity byte checking in the data stream as described in Section 3.9 . Runtime Link Tx Enable must be set to 1b for remote Link Partner for successful enabling of this mode. Default Value is 0b.
8:6	RW	Number of 64 Byte Inserts: Software writes to this to indicate the number 64 Byte inserts are done at a time for Runtime Link Testing. The encodings are: 000b: one 64B insert (for debug purposes only) 001b: two 64B inserts (for debug purposes only) 010b: four 64B inserts Other encodings are reserved. Default value is 000b. See Section 3.9 for guidance on how Software should set this field.
9	RW1C	Parity Feature Nak received: Hardware updates this bit if it receives a Nak from remote Link partner when attempting to enable Runtime Link Testing.
12:10	RsvdP	Reserved
14:13	RW	CRC Injection Enable : Software writes to this bit to trigger CRC error injections, The error is injected by inverting 1, 2 or 3 bits in the CRC bytes. The specific bits inverted are implementation specific. The CRC injection must not happen for Flits that are already inverting CRC bits for Viral handling. The encodings are interpreted as : 00b : CRC Injection is Disabled. 01b : 1 bit is inverted 10b : 2 bits are inverted 11b : 3 bits are inverted. Default Value is 00b.

Table 9-34. Error and Link Testing Control Register (Sheet 2 of 2)

Bit	Attribute	Description
16:15	RW	CRC Injection Count : Software writes to this bit to program the number of CRC injections. It only takes effect if CRC injection Enable is not Disabled. 00b : Single Flit is corrupted. CRC Injection Busy is reset to 0b after single Flit corruption. 01b: A CRC error is injected every 8 Flits. Hardware continues to inject a CRC error every 8 Flits until CRC Injection Enable is 00b. CRC Injection Busy is reset to 0b only after CRC Injection Enable is 00b. 10b: A CRC error is injected every 16 Flits. Hardware continues to inject a CRC error every 16 Flits until CRC Injection Enable is 00b. CRC Injection Busy is reset to 0b only after CRC Injection Enable is 00b. 11b: A CRC error is injected every 64 Flits. Hardware continues to inject a CRC error every 64 Flits until CRC Injection Enable is 00b. CRC Injection Busy is reset to 0b only after CRC Injection Enable is 00b.
17	RO	CRC Injection Busy : Hardware loads a 1b to this bit once it has begun CRC Injection. Software is permitted to poll on this bit. See CRC Injection Count description to see how this bit returns to 0b.
31:18	RsvdP	Reserved

9.5.3.10 Runtime Link Testing Parity Log 0 (Offset 34h)

Table 9-35. Runtime Link Testing Parity Log 0 Register

Bit	Attribute	Description
63:0	RW1C	Parity Log for Module 0: Hardware updates the bit corresponding to the parity error byte with error over the period when Runtime Link Testing was enabled at Rx. Default Value is 0.

9.5.3.11 Runtime Link Testing Parity Log 1 (Offset 3Ch)

Table 9-36. Runtime Link Testing Parity Log 1 Register

Bit	Attribute	Description
63:0	RW1C	Parity Log for Module 1: Hardware updates the bit corresponding to the parity error byte with error over the period when Runtime Link Testing was enabled at Rx. Default Value is 0. This register is only applicable if the Adapter is designed for handling two or more Physical Layer modules. It is reserved otherwise.

9.5.3.12 Runtime Link Testing Parity Log 2 (Offset 44h)

Table 9-37. Runtime Link Testing Parity Log 2 Register

Bit	Attribute	Description
63:0	RW1C	Parity Log for Module 2: Hardware updates the bit corresponding to the parity error byte with error over the period when Runtime Link Testing was enabled at Rx. Default Value is 0. This register is only applicable if the Adapter is designed for handling four Physical Layer modules. It is reserved otherwise.

9.5.3.13 Runtime Link Testing Parity Log 3 (Offset 4Ch)

Table 9-38. Runtime Link Testing Parity Log 3 Register

Bit	Attribute	Description
63:0	RW1C	<p>Parity Log for Module 3: Hardware updates the bit corresponding to the parity error byte with error over the period when Runtime Link Testing was enabled at Rx. Default Value is 0.</p> <p>This register is only applicable if the Adapter is designed for handling four Physical Layer modules. It is reserved otherwise.</p>

9.5.3.14 Advertised Adapter Capability Log (Offset 54h)

Table 9-39. Advertised Adapter Capability Log Register

Bit	Attribute	Description
63:0	RW1C	<p>Advertised Adapter Capability: Hardware updates the bits corresponding to the data bits it sent in the {AdvCap.Adapter} sideband message. Default Value is 0.</p>

9.5.3.15 Finalized Adapter Capability Log (Offset 5Ch)

Table 9-40. Finalized Adapter Capability Log Register

Bit	Attribute	Description
63:0	RW1C	<p>Finalized Adapter Capability: Hardware updates the bits corresponding to the data bits it sent (DP) or received (UP) in the {FinCap.Adapter} sideband message. Default Value is 0.</p>

9.5.3.16 Advertised CXL Capability Log (Offset 64h)

Table 9-41. Advertised CXL Capability Log Register

Bit	Attribute	Description
63:0	RW1C	<p>Advertised CXL Capability: Hardware updates the bits corresponding to the data bits it sent in the {AdvCap.CXL} sideband message, when it is sent with MsgInfo=0000h. Default Value is 0.</p>

9.5.3.17 Finalized CXL Capability Log (Offset 6Ch)

Table 9-42. Finalized CXL Capability Log Register

Bit	Attribute	Description
63:0	RW1C	<p>Finalized CXL Capability: Hardware updates the bits corresponding to the data bits it sent (DP) or received (UP) in the {FinCap.CXL} sideband message, when it is sent with MsgInfo=0000h. Default Value is 0.</p>

9.5.3.18 Advertised Multi-Protocol Capability Log Register (Offset 78h)

This register is reserved for designs that do not implement the Enhanced Multi-protocol capability.

Table 9-43. Advertised Multi-Protocol Capability Log Register

Bit	Attribute	Description
63:0	RW1C	Advertised Multi-Protocol Capability: Hardware updates the bits corresponding to the data bits it sent in the {MultiProtAdvCap.Adapter} sideband message. Default value is 0.

9.5.3.19 Finalized Multi-Protocol Capability Log Register (Offset 80h)

This register is reserved for designs that do not implement the Enhanced Multi-protocol capability.

Table 9-44. Finalized Multi-Protocol Capability Log Register

Bit	Attribute	Description
63:0	RW1C	Finalized Multi-Protocol Capability: Hardware updates the bits corresponding to the data bits it sent in the {MultiProtFinCap.Adapter} sideband message. Default value is 0.

9.5.3.20 Advertised CXL Capability Log Register for Stack 1 (Offset 88h)

This register is reserved for designs that do not implement the Enhanced Multi-protocol capability.

Table 9-45. Advertised CXL Capability Log Register for Stack 1

Bit	Attribute	Description
63:0	RW1C	Advertised CXL Capability: Hardware updates the bits corresponding to the data bits it sent in the {AdvCap.CXL} sideband message when it is sent with MsgInfo=0001h. Default value is 0.

9.5.3.21 Finalized CXL Capability Log Register for Stack 1 (Offset 90h)

This register is reserved for designs not implementing the Enhanced multi-protocol capability.

Table 9-46. Finalized CXL Capability Log Register for Stack 1

Bit	Attribute	Description
63:0	RW1C	Finalized CXL Capability: Hardware updates the bits corresponding to the data bits it sent in the {FinCap.CXL} sideband message when it is sent with MsgInfo=0001h. Default value is 0.

9.5.3.22 PHY Capability (Offset 1000h)

This register is global, and not per module.

Table 9-47. Physical Layer Capability Register

Bit	Attribute	Description
2:0	RO	Reserved
3	RO	Terminated Link: If set to 1, the Receiver supports termination. This bit is always cleared to 0 in an Advanced Package.
4	RO	TX Equalization support 0: TXEQ not supported 1: TXEQ supported
9:5	RO	Supported Tx Vswing encodings 01h: 0.4 V 02h: 0.45 V 03h: 0.5 V 04h: 0.55 V 05h: 0.6 V 06h: 0.65 V 07h: 0.7 V 08h: 0.75 V 09h: 0.8 V 0Ah: 0.85 V 0Bh: 0.9 V 0Ch: 0.95 V 0Dh: 1.0 V 0Eh: 1.05 0Fh: 1.1 V 10h: 1.15 V All other encodings are reserved. This field matches the value advertised by the UCIE Module in the 'Voltage swing' field during MBINIT.PARAM.
10	RsvdP	Reserved
12:11	RO	Rx Clock Mode support 00b: Supports both free running and strobe modes 10b: Free running mode only All other encodings are reserved. This corresponds to the local UCIE Module's capability.
14:13	RO	Rx Clock phase support 00b: Differential clock only (all data rates) 01b: Quadrature clock (24/32 GT/s); Differential clock (16 GT/s and lower) 10b: Same as 01b (for backward compatibility) This corresponds to the local UCIE Module's capability.
15	RO	Package type 0b: Advanced Package 1b: Standard Package
16	RO	Tightly coupled mode (TCM) support 0b: TCM not Supported 1b: TCM supported This corresponds to the local UCIE Module's capability.
31:17	RsvdP	Reserved

9.5.3.23 PHY Control (Offset 1004h)

This register is global, and not per module.

Table 9-48. Physical Layer Control Register

Bit	Attribute	Description
2:0	RW/RO	Reserved. Implementations are encouraged to implement this as an RO bit with a default value of 000b. However, for backward compatibility, implementations are permitted to implement this as an RW bit with a default value of 000b.
3	RW	Rx Terminated Control 0b: Rx Termination disabled 1b: Rx Termination enabled Default is same as 'Terminated Link' bit in PHY capability register. Note that this bit is always cleared to 0 for Advanced Packages. This control is provided for debug purposes only.
4	RW	Tx Eq Enable 0b: Eq Disabled 1b: Eq Enabled Default is 0
5	RW	Rx Clock Mode Select 0: Strobe Mode 1: Free running mode Default is 0 if the Rx of the local UCIE Module supports Strobe Mode; otherwise, the bit is set to 1. This control is provided for debug purposes only. This bit is sent as the 'Clock Mode' bit in the {MBINIT.PARAM configuration req} sideband message.
6	RW	Rx Clock phase support select 0: Differential clock only (all data rates) 1: Quadrature clock (24/32 GT/s); Differential clock (16 GT/s and lower) Default is 0. This control is provided for debug purposes only. This bit is sent as the 'Clock Phase' bit in the {MBINIT.PARAM configuration req} sideband message.
7	RW/RsvdP	Force x32 Width Mode in x64 Module This bit is used only for test and debug purposes. In normal operation, this bit should be reset to 0. When set, this bit will force the x64 module to present "UCIE-A x32 bit =1" during the MBINIT.PARAM exchange phase independent of the value of bit 20, APMW, in the UCIE Link Capability register. This bit applies to all modules in a multi-module link. For x32 Advanced Package modules, this bit is reserved.
8	RW/RsvdP	Force x8 Width Mode in a UCIE-S x16 Module This bit is used only for test and debug purposes. In normal operation, this bit should be reset to 0. When set, this bit will force the x16 module to present "UCIE-S x8" bit =1 during the MBINIT.PARAM exchange phase independent of the value of bit 22, SPMW, in the UCIE Link Capability register. This feature can be used only when there is no lane reversal on the UCIE-S x16 link. This bit applies only to Module 0 in a multi-module link. When set in a multi-module link, it trains only Module 0. For a x8 Standard Package Module, this bit is reserved.
31:9	RsvdP	Reserved

9.5.3.24 PHY Status (Offset 1008h)

This register is global and not per module.

Table 9-49. Physical Layer Status Register

Bit	Attribute	Description
2:0	RO	Reserved
3	RO	Rx Termination Status 0: Rx Termination disabled 1: Rx Termination enabled Default is same as 'Terminated Link' bit in PHY capability register. This is the current status of the local UCIe Module. Note that this is always 0 for Advanced Packages. For Standard packages, whether the Rx decides to terminate the Link could depend on several factors (including channel length in the Package, etc.), and that decision is implementation-specific. For Transmitter of a remote Link partner, it needs this information in order to know whether to Hi-Z the Data and Track Lanes during clock gating and when not performing Runtime Recalibration, respectively. It is expected that this information is known a priori at Package integration time, and the Transmitter is informed of this in an implementation-specific manner.
4	RO	Tx Eq Status 0: Eq Disabled 1: Eq Enabled Default is 0
5	RO	Clock Mode Status 0: Strobe Mode 1: Free running mode Default is 0. This is remote partner's advertised value during MBINIT.PARAM.
6	RO	Clock phase Status 0: Differential clock only (all data rates) 1: Quadrature clock (24/32 GT/s); Differential clock (16 GT/s and lower) This is remote partner's advertised value during MBINIT.PARAM.
7	RO	Lane Reversal within Module: Indicates if Lanes within a module are reversed 0: Lanes within module not reversed 1: Lanes within module are reversed
31:8	RsvdP	Reserved

9.5.3.25 PHY Initialization and Debug (Offset 100Ch)

This register is global, and not per module.

Table 9-50. Phy Init and Debug Register

Bit	Attribute	Description
2:0	RW	<p>Initialization control</p> <p>000b: Initialize to Active. This is the regular Link bring up. 001b: Initialize to MBINIT (Debug mode) (i.e., pause training after completing step-2 of MBINIT.PARAM). 010b: Initialize to MBTRAIN (Debug/compliance mode) (i.e., pause training after entering MBTRAIN after completing step-1 of MBTRAIN.VALVREF). 011b = Pause after completing step-1 of MBTRAIN.RXDESKEW; regardless of entering for initial bring up or from Retrain. 100b = Pause after completing step-1 of MBTRAIN.DATATRAINCENTER2; regardless of entering for initial bring up or from Retrain. All other encodings are reserved.</p> <p>When training has paused, the corresponding state timeouts must be disabled, and hardware resumes training on any of the following triggers:</p> <ul style="list-style-type: none"> • A 0b-to-1b transition on 'Resume Training' bit in this register • Sideband message for the corresponding state is received from remote link partner (e.g., if paused in MBINIT, receiving {MBINIT.CAL Done req} from remote link partner is also a trigger to move forward) <p>A device that does not support the UCIe Test and Compliance register block is permitted to only implement encodings 000b through 010b. Default is 000b.</p>
4:3	RsvdP	Reserved
5	RW	<p>Resume Training</p> <p>A 0b-to-1b transition on this bit triggers hardware to resume training from the last link training state, achieved via 'Initialization Control' field in this register until ACTIVE. A device that does not support the UCIe Test and Compliance register block is permitted to hardwire this bit to 0b. Default is 0b.</p>
31:6	RsvdP	Reserved

9.5.3.26 Training Setup 1 (Offset 1010h)

This register is replicated per module. Offsets 1010h to 101Ch are used in 4B increments for multi-module scenarios

Table 9-51. Training Setup 1 Register

Bit	Attribute	Description
2:0	RW	Data pattern used during training 000b: Per-Lane LFSR pattern 001b: Per-Lane ID pattern 010b: If @PHY-Compliance {Per-Lane Clock pattern AA pattern} Else Reserved 011b: If @PHY-Compliance {Per-Lane all 0 pattern} Else Reserved 100b: If @PHY-Compliance {Per-Lane all 1 pattern} Else Reserved 101b: If {@PHY-Compliance Per-Lane inverted Clock pattern} Else Reserved All other encodings are reserved Default is 000b.
5:3	RW	Valid Pattern used during training 000b: Functional valid pattern (1111 0000 (lsb first)) All other encodings are reserved Default is 000b.
9:6	RW	Clock Phase control 0h: Clock PI center found by Transmitter 1h: Left edge found through Data to clock training 2h: Right edge found through Data to clock training All other encodings are reserved Default = 0
10	RW	Training mode 0b: Continuous mode 1b: Burst Mode Default = 0
26:11	RW	Burst Count: Indicates the duration of selected pattern (UI count) Default = 4h
31:27	RsvdP	Reserved

9.5.3.27 Training Setup 2 (Offset 1020h)

This register is replicated per module. Offsets 1020h to 102Ch are used in 4B offset increments for multi-module scenarios.

Table 9-52. Training Setup 2 Register

Bit	Attribute	Description
15:0	RW	Idle count: Indicates the duration of low following the burst (UI count) Default = 4h
31:16	RW	Iterations: Indicates the iteration count of bursts followed by idle (UI count) Default = 4h

9.5.3.28 Training Setup 3 (Offset 1030h)

This register is replicated per module. Offsets 1030h to 1048h are used in 8B offset increments for multi-module scenarios.

Table 9-53. Training Setup 3 Register

Bit	Attribute	Description
63:0	RW	Lane mask: Indicated the Lanes to mask during Rx comparison. Example 1h = Lane 0 is masked during comparison. Default = 0 (no mask).

9.5.3.29 Training Setup 4 (Offset 1050h)

This register is replicated per module. Offsets 1050h to 105Ch are used in 4B offset increments for multi-module scenarios.

Table 9-54. Training Setup 4 Register

Bit	Attribute	Description
3:0	RW	Repair Lane mask: Indicated the Redundant Lanes to mask during Rx comparison. Example 1h = RD0 is masked during comparison 2h: RD1 mask. Default = 0 (no mask).
15:4	RW	Max error Threshold in per Lane comparison: Indicates threshold for error counting to start. For Tx-initiated tests, these values are sent in the corresponding {Start Tx Init D to C point test req} and {Start Tx Init D to C eye sweep req} sideband messages. The remote Link partner must use these values for checking errors against the threshold. For Rx-initiated tests, these values are sent in the corresponding {Start Rx Init D to C point test req} and {Start Rx Init D to C eye sweep req} sideband messages as an inform. The receiver uses these values for checking errors against the threshold. Default = 0 (all errors are counted).
31:16	RW	Max error Threshold in aggregate comparison: Indicates threshold for error counting to start. For Tx-initiated tests, these values are sent in the corresponding {Start Tx Init D to C point test req} and {Start Tx Init D to C eye sweep req} sideband messages. The remote Link partner must use these values for checking errors against the threshold. For Rx-initiated tests, these values are sent in the corresponding {Start Rx Init D to C point test req} and {Start Rx Init D to C eye sweep req} sideband messages as an inform. The receiver uses these values for checking errors against the threshold. Default = 0 (all errors are counted).

9.5.3.30 Current Lane Map Module 0 (Offset 1060h)

Table 9-55. Current Lane Map Module 0 Register

Bit	Attribute	Description
63:0	RW	Current Rx Lane map (CLM) for Module-0: If a bit is 1 it indicates the corresponding physical Lane is operational. For Standard package modules, bits 63:16 of this register are not applicable. For UCIe-A x32 implementations (i.e., APMW bit in UCIe Link Capability register is set), bits 63:32 of this register are not applicable. Default Value is all 0s.

9.5.3.31 Current Lane Map Module 1 (Offset 1068h)

Table 9-56. Current Lane Map Module 1 Register

Bit	Attribute	Description
63:0	RW	Current Rx Lane map (CLM) for Module-1: If a bit is 1 it indicates the corresponding physical Lane is operational. For Standard package modules, bits 63:16 of this register are not applicable. For UCIe-A x32 implementations (i.e., APMW bit in UCIe Link Capability register is set), bits 63:32 of this register are not applicable. Default Value is all 0s. This register is reserved if Module 1 is not present

9.5.3.32 Current Lane Map Module 2 (Offset 1070h)

Table 9-57. Current Lane Map Module 2 Register

Bit	Attribute	Description
63:0	RW/RsvdP	Current Rx Lane map (CLM) for Module-2: If a bit is 1 it indicates the corresponding physical Lane is operational. For Standard package modules, bits 63:16 of this register are not applicable. For UCIe-A x32 implementations (i.e., APMW bit in UCIe Link Capability register is set), bits 63:32 of this register are not applicable. Default Value is all 0s. This register is reserved if Module 2 is not present

9.5.3.33 Current Lane Map Module 3 (Offset 1078h)

Table 9-58. Current Lane Map Module 3 Register

Bit	Attribute	Description
63:0	RW/RsvdP	Current Rx Lane map (CLM) for Module-3: If a bit is 1 it indicates the corresponding physical Lane is operational. For Standard package modules, bits 63:16 of this register are not applicable. For UCIe-A x32 implementations (i.e., APMW bit in UCIe Link Capability register is set), bits 63:32 of this register are not applicable. Default Value is all 0s. This register is reserved if Module 3 is not present

9.5.3.34 Error Log 0 (Offset 1080h)

This register is replicated per module. Offsets 1080h to 108Ch are used in 4B offset increments for multi-module scenarios.

Table 9-59. Error Log 0 Register

Bit	Attribute	Description
7:0	ROS	<p>State N: Captures the current Link training state machine status. State Encodings are given by:</p> <ul style="list-style-type: none"> 00h RESET 01h SBINIT 02h MBINIT.PARAM 03h MBINIT.CAL 04h MBINIT.REPAIRCLK 05h MBINIT.REPAIRVAL 06h MBINIT.REVERSALMB 07h MBINIT.REPAIRMB 08h MBTRAIN.VALVREF 09h MBTRAIN.DATAVREF 0Ah MBTRAIN.SPEEDIDLE 0Bh MBTRAIN.TXSELFCAL 0Ch MBTRAIN.RXSELFCAL 0Dh MBTRAIN.VALTRAINCENTER 0Eh MBTRAIN.VALTRAINVREF 0Fh MBTRAIN.DATATRAINCENTER1 10h MBTRAIN.DATATRAINVREF 11h MBTRAIN.RXDESKEW 12h MBTRAIN.DATATRAINCENTER2 13h MBTRAIN.LINKSPEED 14h MBTRAIN.REPAIR 15h PHYRETRAIN 16h LINKINIT 17h ACTIVE 18h TRAINERROR 19h L1/L2 <p>All other encodings are reserved Default is 0</p>
8	ROS	Lane Reversal: 1b indicates Lane Reversal within the module. Default is 0
9	ROS	Width Degrade: 1b indicates Module width Degrade. Applicable to Standard package only. Default is 0.
15:10	RsvdZ	Reserved
23:16	ROS	State (N-1): Captures the state before State N was entered for Link training state machine. State encodings are the same as State N field. Default is 0
31:24	ROS	State (N-2): Captures the state before State (N-1) was entered for Link training state machine. State encodings are the same as State N field. Default is 0

9.5.3.35 Error Log 1 (Offset 1090h)

This register is replicated per module. Offsets 1090h to 109Ch are used in 4B offset increments for multi-module scenarios.

Table 9-60. Error Log 1 Register

Bit	Attribute	Description
7:0	ROS	State (N-3): Captures the state status before State (N-2) was entered. State encodings are the same as State N field. Default is 0
8	RW1CS	State Timeout Occurred: Hardware sets this to 1b if a Link Training State machine state or sub-state timed out and it was escalated as a fatal error. Default value is 0b.
9	RW1CS	Sideband Timeout Occurred: Hardware sets this to 1b if a sideband handshake timed out, for example, if a RDI request did not get a response for 8ms. Sideband handshakes related to Link Training messages are not included here. Default value is 0b.
10	RW1CS	Remote LinkError received: Hardware sets this to 1b if remote Link partner requested LinkError transition through RDI sideband. Default value is 0b.
11	RW1CS	Internal Error: Hardware sets this to 1b if any implementation specific internal error occurred in the Physical Layer. Default value is 0b.
31:12	RsvdZ	Reserved

9.5.3.36 Runtime Link Test Control (Offset 1100h)

Table 9-61. Runtime Link Test Control (Sheet 1 of 2)

Bit	Attribute	Description
0	RW/RO	Implementations are encouraged to implement this as an RO bit with a default value of 0. However, for backward compatibility, implementations are permitted to implement this as an RW bit with a default value of 0.
1	RW/RO	Implementations are encouraged to implement this as an RO bit with a default value of 0. However, for backward compatibility, implementations are permitted to implement this as an RW bit with a default value of 0.
2	RW	Apply Module 0 Lane Repair: For Advanced Package, software programs this bit to inform Physical Layer hardware to apply Lane repair for this logical module id at the next Retrain cycle, if this Module is operational. For Standard Package, this bit will trigger a width degrade for logical Module 0, if possible and relevant. Default value is 0.
3	RW	Apply Module 1 Lane Repair: For Advanced Package, software programs this bit to inform Physical Layer hardware to apply Lane repair for this logical module id at the next Retrain cycle, if this Module is operational. For Standard Package, this bit will trigger a width degrade for logical Module 1, if possible and relevant. Default value is 0. These bits are reserved if Module 1 is not present.

Table 9-61. Runtime Link Test Control (Sheet 2 of 2)

Bit	Attribute	Description
4	RW	Apply Module 2 Lane Repair: For Advanced Package, software programs this bit to inform Physical Layer hardware to apply Lane repair for this logical module id at the next Retrain cycle, if this Module is operational. For Standard Package, this bit will trigger a width degrade for logical Module 2, if possible and relevant. Default value is 0. These bits are reserved if Module 2 is not present.
5	RW	Apply Module 3 Lane Repair: For Advanced Package, software programs this bit to inform Physical Layer hardware to apply Lane repair for this logical module id at the next Retrain cycle, if this Module is operational. For Standard Package, this bit will trigger a width degrade for logical Module 3, if possible and relevant. Default value is 0. These bits are reserved if Module 3 is not present.
6	RW	Start: Software writes to this bit before setting Link Retrain bit to inform hardware that the contents of this register are valid. HW clears this bit to 0 after the Busy bit in the Runtime Link Test Status register is set to 1.
7	RW	Inject Stuck-at fault: Software writes 1b to this bit to indicate hardware must inject a stuck at fault for the Lane id identified in Lane Repair id (the specific Module's lane(s) in which the fault is injected is indicated by the 'Apply Module x Lane Repair' bits) for the corresponding field. Injecting the fault at Tx or Rx is implementation specific. This bit takes effect during the next link retraining (see Section 4.5.3.7 for further details). Default value is 0b.
14:8	RW	Module 0 Lane repair id: For Advanced Package, software programs this bit to inform Physical Layer hardware to apply Lane repair for this logical transmit Lane id in logical Module 0 at the next Retrain cycle, if this Module is operational. For Standard Package, this bit will trigger a width degrade for logical Module 0, if possible and relevant. Default is 0. These bits are reserved if Module 0 is not present.
21:15	RW	Module 1 Lane repair id: For Advanced Package, software programs this bit to inform Physical Layer hardware to apply Lane repair for this logical transmit Lane id in logical Module 1 at the next Retrain cycle, if this Module is operational. For Standard Package, this bit will trigger a width degrade for logical Module 1, if possible and relevant. Default is 0. These bits are reserved if Module 1 is not present.
28:22	RW	Module 2 Lane repair id: For Advanced Package, software programs this bit to inform Physical Layer hardware to apply Lane repair for this logical transmit Lane id in logical Module 2 at the next Retrain cycle, if this Module is operational. For Standard Package, this bit will trigger a width degrade for logical Module 2, if possible and relevant. Default is 0. These bits are reserved if Module 2 is not present.
35:29	RW	Module 3 Lane repair id: For Advanced Package, software programs this bit to inform Physical Layer hardware to apply Lane repair for this logical transmit Lane id in logical Module 3 at the next Retrain cycle, if this Module is operational. For Standard Package, this bit will trigger a width degrade for logical Module 3, if possible and relevant. Default is 0. These bits are reserved if Module 3 is not present.
63:36	RsvdP	Reserved

9.5.3.37 Runtime Link Test Status (Offset 1108h)

Table 9-62. Runtime Link Test Status Register

Bit	Attribute	Description
0	RO	Busy: Hardware loads 1b to this bit once Start bit is written by software. Hardware loads 0b to this bit once it has attempted to complete the actions requested in Runtime Link Test Control register. Default is 0
31:1	RsvdZ	Reserved

9.5.3.38 Mainband Data Repair (Offset 110Ch)

This register is replicated per advanced module. For Standard package, this register is not applicable. Offsets 110Ch to 1124h are used in 8B offset increments for multi-module scenarios.

Table 9-63. Mainband Data Repair Register (Sheet 1 of 2)

Bit	Attribute	Description
7:0	RO	Repair Address for TRD_P[0]: Indicates the physical Lane repaired when TRD_P[0] is used in remapping scheme 00h: TD_P[0] Repaired 01h: TD_P[1] Repaired 02h: TD_P[2] Repaired 1Eh: TD_P[30] Repaired 1Fh: TD_P[31] Repaired F0h: Repair attempt failed FFh: No Repair
15:8	RO	Repair Address for TRD_P[1]: Indicates the physical Lane repaired when TRD_P[1] is used in remapping scheme 00h: Invalid 01h: TD_P[1] Repaired 02h: TD_P[2] Repaired 1Eh: TD_P[30] Repaired 1Fh: TD_P[31] Repaired F0h: Repair attempt failed FFh: No Repair
23:16	RO	Repair Address for TRD_P[2]: Indicates the physical Lane repaired when TRD_P[2] is used in remapping scheme 20h: TD_P[32] Repaired 21h: TD_P[33] Repaired 22h: TD_P[34] Repaired 3Eh: TD_P[62] Repaired 3Fh: TD_P[63] Repaired F0h: Repair attempt failed FFh: No Repair This field is reserved for UCIe-A x32 module implementations.

Table 9-63. Mainband Data Repair Register (Sheet 2 of 2)

Bit	Attribute	Description
31:24	RO	<p>Repair Address for TRD_P[3]: Indicates the physical Lane repaired when TRD_P[3] is used in remapping scheme</p> <p>20h: Invalid 21h: TD_P[33] Repaired 22h: TD_P[34] Repaired 3Eh: TD_P[62] Repaired 3Fh: TD_P[63] Repaired F0h: Repair attempt failed FFh: No Repair</p> <p>This field is reserved for UCIE-A x32 module implementations.</p>
39:32	RO	<p>Repair Address for RRD_P[0]: Indicates the physical Lane repaired when RRD_P[0] is used in remapping scheme</p> <p>00h: RD_P[0] Repaired 01h: RD_P[1] Repaired 02h: RD_P[2] Repaired 1Eh: RD_P[30] Repaired 1Fh: RD_P[31] Repaired F0h: Repair attempt failed FFh: No Repair</p>
47:40	RO	<p>Repair Address for RRD_P[1]: Indicates the physical Lane repaired when RRD_P[1] is used in remapping scheme</p> <p>00h: RD_P[0] Repaired 01h: RD_P[1] Repaired 02h: RD_P[2] Repaired 1Eh: RD_P[30] Repaired 1Fh: RD_P[31] Repaired F0h: Repair attempt failed FFh: No Repair</p>
55:48	RO	<p>Repair Address for RRD_P[2]: Indicates the physical Lane repaired when RRD_P[2] is used in remapping scheme</p> <p>20h: RD_P[32] Repaired 21h: RD_P[33] Repaired 22h: RD_P[34] Repaired 3Eh: RD_P[62] Repaired 3Fh: RD_P[63] Repaired F0h: Repair attempt failed FFh: No Repair</p> <p>This field is reserved for UCIE-A x32 implementations.</p>
63:56	RO	<p>Repair Address for RRD_P[3]: Indicates the physical Lane repaired when RRD_P[3] is used in remapping scheme</p> <p>20h: RD_P[32] Repaired 21h: RD_P[33] Repaired 22h: RD_P[34] Repaired 3Eh: RD_P[62] Repaired 3Fh: RD_P[63] Repaired F0h: Repair attempt failed FFh: No Repair</p> <p>This field is reserved for UCIE-A x32 module implementations.</p>

9.5.3.39 Clock, Track, Valid and Sideband Repair (Offset 1134h)

This register is replicated per module. Offsets 1134h to 1140h are used in 4B offset increments for multi-module scenarios.

Table 9-64. Clock, Track, Valid and Sideband Repair Register

Bit	Attribute	Description
3:0	RO	Repair Address for TRDCK_P: Indicates the physical Lane repaired when TRDCK_P is used in remapping scheme 0h: TCKP_P Repaired 1h: TCKN_P Repaired 2h: TTRK_P Repaired 7h: Repair attempt failed Fh: No Repair All other encodings are reserved.
7:4	RO	Repair Address for RRDCK_P: Indicates the physical Lane repaired when RRDCK_P is used in remapping scheme 0h: RCKP_P Repaired 1h: RCKN_P Repaired 2h: RTRK_P Repaired 7h: Repair attempt failed Fh: No Repair All other encodings are reserved.
9:8	RO	Repair Address for TRDVLD_P: Indicates the physical Lane repaired when TRDVLD_P is used in remapping scheme 00b: TVLD_P Repaired 01b: Repair attempt failed 10b: Reserved 11b: No Repair
11:10	RO	Repair Address for RRDVLD_P: Indicates the physical Lane repaired when RRDVLD_P is used in remapping scheme 00b: RVLD_P Repaired 01b: Repair attempt failed 10b: Reserved 11b: No Repair
15:12	RsvdP	Reserved
19:16	RO	Repair Address for Sideband Transmitter: Indicates sideband repair result for the Transmitter Result[3:0]
23:20	RO	Repair Address for Sideband Receiver: Indicates sideband repair result for the Transmitter Result[3:0]
31:24	RsvdP	Reserved

9.5.3.40 UCIe Link Health Monitor (UHM) DVSEC

This DVSEC is an extended Capability. It is required for all devices that support Compliance testing (as indicated by the presence of Compliance/Test Register Locator) and optional otherwise. This DVSEC contains the required registers for SW to read eye margin values per lane. SW Flow for Eye Margining is as follows:

- SW ensures that the Eye Margin Valid (EMV) bit in UHM_STS register is cleared
- SW triggers a retrain of the link
 - When the retrain completes (as indicated by bit 16 in the UCIe Link Status register) and the EMV bit is set in the UHM_STS register, SW can read the EM*_Ln*_Mod* registers in UHM DVSEC to know the margins. Receive margins are logged in the Tx UHM registers.

Note that HW may also measure Eye Margins during HW-autonomous retraining and/or initial training and if measured, is permitted to report it in the Eye Margin registers whenever the EMV bit is cleared.

For x32 Advanced Packaging implementations, EML* and EMR* registers for Lanes 63:32 are RsvdP.

Figure 9-5. UCIe Link Health Monitor (UHM) DVSEC

PCI Express Extended Capability Header					
Designated Vendor Specific Header 1					
Reserved	Designated Vendor Specific Header 2				
UHM_STS	Reserved				
Reserved					
Reserved					
EMR_Ln1_Mod0	EML_Ln1_Mod0	EMR_Ln0_Mod0	EML_Ln0_Mod0		
EMR_Ln3_Mod0	EML_Ln3_Mod0	EMR_Ln2_Mod0	EML_Ln2_Mod0		
...					
...					
EMR_Ln1_Mod1	EML_Ln1_Mod1	EMR_Ln0_Mod1	EML_Ln0_Mod1		
EMR_Ln3_Mod1	EML_Ln3_Mod1	EMR_Ln2_Mod1	EML_Ln2_Mod1		
...					
...					

Table 9-65. UHM DVSEC - Designated Vendor Specific Header 1, 2 (Offsets 04h and 08h)

Register	Field	Bit Location	Value
Designated Vendor-Specific Header 1 (offset 04h)	DVSEC Vendor ID	15:0	D2DEh
	DVSEC Revision	19:16	0h
	Length	31:20	Design dependent
Designated Vendor-Specific Header 2 (offset 08h)	DVSEC ID	15:0	1h

9.5.3.40.1 UHM Status (Offset Eh)

Table 9-66. UHM Status

Bit	Attribute	Description
7:0	RO	Step Count Step count used in the reporting of margin information. A value of 0 indicates 256. For example, a value of 32 indicates that the UI is equally divided into 32 steps and Eye Margin registers provide the left and right margins in multiples of UI/32.
8	RW1C	Eye Margin Valid (EMV) This bit, when set, indicates that margin registers carry valid information from the last retrain. SW must clear this bit before initiating link retrain, if it intends to measure eye margins during the retrain. On a SW-initiated link retrain, if after retrain, this bit is cleared, then SW should infer that there was some error in margin measurement. Note that HW logs any new Eye Margin measurements (whether it is measured during SW-initiated retrain, during HW-autonomous retraining, or during initial training) in the Eye Margin registers only when this bit is cleared.
15:9	RsvdP	Reserved

9.5.3.40.2 Eye Margin (Starting Offset 18h)

Table 9-67. EML_Lnx_Mody

Bit	Attribute	Description
7:0	RO	Eye Margin Left for Lane x and Module y Provides the left eye margin relative to the PI center, in units of UI/Step Count.

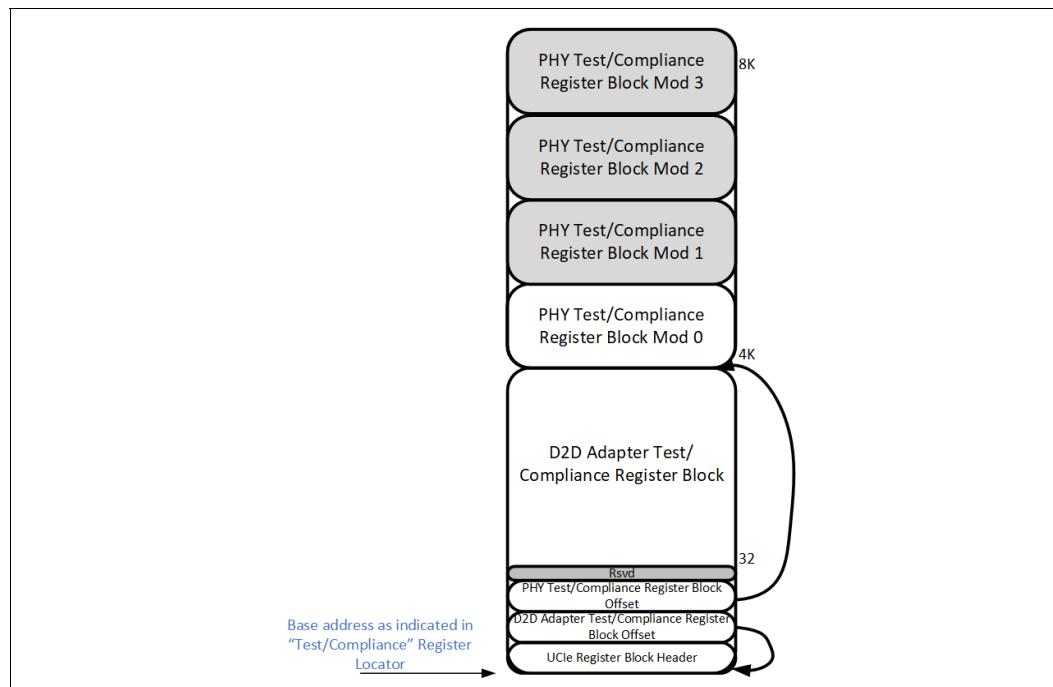
Table 9-68. EMR_Lnx_Mody

Bit	Attribute	Description
7:0	RO	Eye Margin Right for Lane x and Module y Provides the right eye margin relative to the PI center, in units of UI/Step Count.

9.5.4 Test/Compliance Register Block

The Test/Compliance register block is 8 KB in size with first 4 KB from base address (as enumerated via register locator with Register Block Identifier of 1h) used for D2D Adapter-related Test/Compliance registers and the second 4 KB used for PHY-related Test/Compliance registers. For future extensibility, these offsets are enumerable via the associated Register Block Offset registers, as shown in [Figure 9-6](#).

Figure 9-6. UCIE Test/Compliance Register Block



9.5.4.1 UCIe Register Block Header

Table 9-69. UCIe Register Block Header (Offset 0h)

Bit	Attributes	Description
15:0	RO	Vendor ID Default is set to Vendor ID assigned for UCIe Consortium - D2DEh.
31:16	RO	Vendor ID Register Block Set to 1h to indicate Test Compliance register block.
35:32	RO	Vendor Register Block Version Set to 0h.
63:36	RsvdP	Reserved
95:64	RO	Vendor Register Block Length The number of bytes in the register block including the UCIe register block header. Default is 2000h.
127:96	RsvdP	Reserved

9.5.4.2 D2D Adapter Test/Compliance Register Block Offset

Table 9-70. D2D Adapter Test/Compliance Register Block Offset (Offset 10h)

Bit	Attributes	Description
7:0	RO	D2D Adapter Test/Compliance Register Block Offset (D2DOFF) 4-KB granular offset from Test/Compliance Register Block base address for D2D Adapter Test/Compliance registers. This field should be set to 0. However, SW must read this field to know the actual offset, for future compatibility reasons.
15:8	RO	D2D Adapter Test/Compliance Register Block Length 4-KB granular length of the D2D Adapter Test/Compliance registers. This field should be set to 1 to indicate 4-KB length. However, SW must read this field to know the actual length, for future compatibility reasons.
31:16	RsvdP	Reserved

9.5.4.3 PHY Test/Compliance Register Block Offset

Table 9-71. PHY Test/Compliance Register Block Offset (Offset 14h)

Bit	Attributes	Description
7:0	RO	PHY Test/Compliance Register Block Offset (PHYOFF) 4-KB granular offset from Test/Compliance Register Block base address for PHY Adapter Test/Compliance registers. This field should be set to 1, indicating that the registers start at 4 KB from the base address. However, SW must read this field to know the actual offset, for future compatibility reasons.
15:8	RO	PHY Test/Compliance Register Block Length 4-KB granular length of the PHY Test/Compliance registers. This field should be set to 1 to indicate 4-KB length. However, SW must read this field to know the actual length, for future compatibility reasons.
31:16	RsvdP	Reserved

9.5.4.4 D2D Adapter Test/Compliance Register Block

9.5.4.4.1 Adapter Compliance Control

Table 9-72. Adapter Compliance Control (Offset 20h from D2DOFF)

Bit	Attributes	Description
1:0	RW	<p>Compliance Mode Any write to this register takes effect after the next entry of RDI state status to Retrain.</p> <ul style="list-style-type: none"> • 00b = Normal mode of operation • 01b = PHY only Link Training or Retraining <ul style="list-style-type: none"> — Adapter performs the necessary RDI handshakes to bring RDI to Active but does not perform Parameter exchanges or Adapter vLSM handshakes and keeps FDI in Reset to prevent mainband traffic. — Adapter must still trigger RDI to Retrain if software programmed the Retrain bit in Link Control. — Sideband Register Access requests and completions are operational in this mode. • 10b = Adapter Compliance <ul style="list-style-type: none"> — Adapter performs the necessary RDI handshakes to bring RDI to Active but does not perform Parameter exchanges or Adapter vLSM handshakes (unless triggered by software) and keeps FDI in Reset. — Adapter only performs actions based on the triggers and setup according to the registers defined in Section 9.5.4.4.2 to Section 9.5.4.4.6. — Adapter must still trigger RDI to Retrain if software programmed the Retrain bit in Link Control. — Sideband Register Access requests and completions are operational in this mode. • 11b = Reserved <p>Any RDI transition to LINKERROR when this field is either 01b or 10b does not reset any registers. Default is 00b.</p>
2	RW	<p>Force Link Reset If set to 1b, Adapter transitions RDI to LinkError state. This bit is used by Compliance software to re-initialize the DUT anytime during Compliance testing. If SW expectation is that the DUT reinitializes to normal mode at the end of link reset, the Compliance Mode field in this register must be 00b and the Compliance Enable for PHY bit in the PHY Compliance Control Register must be 0b.</p>
31:3	RsvdP	Reserved

9.5.4.4.2 Flit Tx Injection Control

Table 9-73. Flit Tx Injection Control (Offset 28h from D2DOFF) (Sheet 1 of 2)

Bit	Attributes	Description
0	RW	Flit Tx Injection Enable Setting this bit to 1b starts Flit injection from the Adapter to the PHY at the Transmitter. Clearing this bit to 0b stops Flit injection on the Link. Default is 0b.
3:1	RW	Flit Type Type of Flit injected. <ul style="list-style-type: none"> • 000b = Adapter NOP Flits. These bypass TX retry buffer. • 001b = Test Flits. • 010b = Alternate between NOP Flits and Test Flits. • All other encodings are reserved. Default is 000b.
5:4	RW	Injection mode <ul style="list-style-type: none"> • 00b = Continuous injection of Flits as specified by Flit Type field. • 01b = Inject 'Flit Inject Number' of Flits contiguously without any intervening Protocol Flits. • 10b = Inject 'Flit Inject Number' of Flits while interleaving with Protocol Flits. If Protocol Flits are available, alternate between Protocol Flits and Injected Flits. If no Protocol Flits are available then, inject consecutively. • 11b = Reserved. Default is 00b.
13:6	RW	Flit Inject Number If the Injection mode is not 00b, this field indicates the number of Flits injected. Default is 00h.
17:14	RW	Payload Type This field determines the payload type used if Test Flits are injected. Payload includes all bits in the Flit with the exception of Flit Header, CRC, and Reserved bits. <ul style="list-style-type: none"> • 0h = Fixed 4B pattern picked up from 'Payload Fixed Pattern' field of this register, inserted so as to cover all the Payload bytes (with the same pattern replicated in incrementing 4B chunks) • 1h = Random 4B pattern picked up from a 32b LFSR (linear feedback shift register used for pseudo random pattern generation), inserted so as to cover all the Payload bytes (with the same pattern replicated in incrementing 4B chunks) • 2h = Fixed 4 byte pattern picked up from 'Payload Fixed Pattern' field of this register, inserted once at the 'Flit Byte Offset' location within the Flit • 3h = Random 4B pattern picked up from a 32b LFSR, inserted once at the 'Flit Byte Offset' location within the Flit and the rest of the payload is assigned 0b • 4h = Same as 2h, except the 4B pattern is injected every 'Pattern Repetition' bytes starting with 'Flit Byte Offset' • 5h = Same as 3h, except the 4B pattern is injected every 'Pattern Repetition' bytes starting with 'Flit Byte Offset' and the rest of the payload is assigned 0b • All other encodings are reserved Default is 0h. LFSR seed and primitive polynomial choice is implementation specific. Note: While in mission mode, because scrambling is always enabled, changing the Payload Type may have no benefit. This may, however, be useful during compliance testing with scrambling disabled.

Table 9-73. Flit Tx Injection Control (Offset 28h from D2DOFF) (Sheet 2 of 2)

Bit	Attributes	Description
25:18	RW	Flit Byte Offset See 'Payload Type'. Default is 00h.
31:26	RW	Pattern Repetition See 'Payload Type'. A value of 00h or 01h must be interpreted as a single pattern occurrence. Default is 00h.
63:32	RW	Payload Fixed Pattern See 'Payload Type'. Default is 0000 0000h.

9.5.4.4.3 Adapter Test Status (Offset 30h from D2DOFF)**Table 9-74. Adapter Test Status (Sheet 1 of 2)**

Bit	Attributes	Description
0	RO	Compliance Status If Adapter is in 'PHY only Link Training or Retraining' or 'Adapter Compliance' mode, it is set to 1b; otherwise, it is 0b.
2:1	RO	Flit Tx Injection Status <ul style="list-style-type: none"> 00b = No Flits injected. 01b = At least one Flit was injected, but not completed. For Continuous Injection mode, this will be the status until Flit Injection Enable transitions from 1b to 0b. 10b = Completed Flit Injection, for cases in which a finite number of Flit injections was set up. 11b = Flit Injection Enable transitioned from 1b to 0b before Flit injections were complete. <p>This field is cleared to 00b on a 0b-to-1b transition of Flit Injection Enable bit. Default is 00b.</p>
4:3	RW1C	Flit Rx Status <ul style="list-style-type: none"> 00b = No Test Flits received 01b = Received at least one Test Flit without CRC error All other encodings are reserved <p>Default is 00b.</p>
5	RO	Link State Request Injection Status for Stack 0 <ul style="list-style-type: none"> 0b = No request injected 1b = Completed Request Injection <p>This bit is cleared to 0b on a 0b-to-1b transition of 'Link State Request or Response Injection Enable'.</p>
6	RO	Link State Response Injection Status for Stack 0 <ul style="list-style-type: none"> 0b = No response injected 1b = Completed Response Injection <p>This bit is cleared to 0b on a 0b-to-1b transition of 'Link State Request or Response Injection Enable'.</p>
7	RO	Link State Request Injection Status for Stack 1 <ul style="list-style-type: none"> 0b = No request injected 1b = Completed Request Injection <p>This bit is cleared to 0b on a 0b-to-1b transition of 'Link State Request or Response Injection Enable'.</p>

Table 9-74. Adapter Test Status (Sheet 2 of 2)

Bit	Attributes	Description
8	RO	Link State Response Injection Status for Stack 1 <ul style="list-style-type: none"> 0b = No response injected 1b = Completed Response Injection This bit is cleared to 0b on a 0b-to-1b transition of 'Link State Request or Response Injection Enable'.
10:9	RO	Retry Injection Status <ul style="list-style-type: none"> 00b = No errors injected on Transmitted Flits 01b = Injected error on at least one transmitted Flit 10b = Finished error injection sequence on transmitted Flits 11b = Reserved This field is cleared to 00b on a 0b-to-1b transition of 'Retry Injection Enable'.
11	RO	Number of Retries Exceeded Threshold Set to 1b if the number of independent retry events exceed the threshold defined in 'Tx Retry Error Threshold'. This bit is cleared to 0b on a 0b-to-1b transition of 'Retry Injection Enable'.
31:12	RsvdZ	Reserved

9.5.4.4.4 Link State Injection Control Stack 0 (Offset 34h from D2DOFF)

As mentioned in [Section 11.2](#), this register only takes effect when the Adapter is in Adapter Compliance Mode.

Table 9-75. Link State Injection Control Stack 0

Bit	Attributes	Description
0	RW	Link State Request or Response Injection Enable at Tx <ul style="list-style-type: none"> 0b = Link State Request or Response Injection not enabled at Tx 1b = Link State Request or Response Injection enabled at Tx
1	RW	Injection Type <ul style="list-style-type: none"> 0b = Inject a request packet with the request matching "Link Request" field 1b = Inject a response packet with the response matching "Link Response" field when a request matching "Link Request" field is received
5:2	RW	Link Request The encodings match the State request encodings of FDI.
9:6	RW	Link Response The encodings match the State response encodings of FDI.
31:10	RsvdP	Reserved

9.5.4.4.5 Link State Injection Control Stack 1 (Offset 38h from D2DOFF)

As mentioned in Section 11.2, this register only takes effect when the Adapter is in Adapter Compliance Mode.

Table 9-76. Link State Injection Control Stack 1

Bit	Attributes	Description
0	RW	Link State Request or Response Injection Enable at Tx <ul style="list-style-type: none"> 0b = Link State Request or Response Injection not enabled at Tx 1b = Link State Request or Response Injection enabled at Tx
1	RW	Injection Type <ul style="list-style-type: none"> 0b = Inject a request packet with the request matching "Link Request" field 1b = Inject a response packet with the response matching "Link Response" field when a request matching "Link Request" field is received
5:2	RW	Link Request The encodings match the State request encodings of FDI.
9:6	RW	Link Response The encodings match the State response encodings of FDI.
31:10	RsVdP	Reserved

9.5.4.4.6 Retry Injection Control (Offset 40h from D2DOFF)

Table 9-77. Retry Injection Control (Sheet 1 of 2)

Bit	Attributes	Description
0	RW	Retry Injection Enable Setting this bit to 1b enables and starts error injections at Tx to force Retry on the UCIE Link. Clearing this bit to 0b stops Flit injection on the Link. Default is 0b.
3:1	RW	Error Injection Type on Transmitted Flits <ul style="list-style-type: none"> 000b = No errors injected on Transmitted Flits 001b = 1-bit error injected in 'Byte Offset' of the Flit, it is permitted to invert any bit in the corresponding byte position 010b = 2-bit error injected in 'Byte Offset' of the Flit, it is permitted to invert any two bits in the corresponding byte position 011b = 3-bit error injected in 'Byte Offset' of the Flit, it is permitted to invert any three bits in the corresponding byte position All other encodings are reserved Default is 000b.
11:4	RW	Byte Offset See 'Error Injection Type on Transmitted Flits'. 00h means error is injected on Byte 0, 01h means error is injected in Byte 1, and so on. Default is 00h.
19:12	RW	Number of Flits between Injected Errors A nonzero value indicates the exact number of Flits after which a subsequent error is injected. A value of 0 will inject errors after a pseudo-random number of Flits between 1 and 31, chosen from a 32b LFSR output. Default is 00h.

Table 9-77. Retry Injection Control (Sheet 2 of 2)

Bit	Attributes	Description
27:20	RW	<p>Number of Errors Injected Represents the number of errors injected on the Transmitted Flits. A value of 0 indicates that the error injection continues until the Retry Injection Enable is disabled. Default is 00h.</p>
30:28	RW	<p>Flit Type for Error Injection</p> <ul style="list-style-type: none"> • 000b = Inject errors on any Flit type. • 001b = Only inject errors on NOP Flits. • 010b = Only inject errors on Payload Flits (Protocol Flits or Test Flits). • 011b = Only inject errors on Test Flits. • 100b = Only inject errors on Payload Flits. Subsequent errors injected on the same sequence number ('Number of Flits between Injected Errors' is ignored for this case). <p>Note: The 100b value can be used to test Replay number Rollover rules.</p> <ul style="list-style-type: none"> • All other encodings are reserved <p>Default value is 000b.</p>
31	RsvdP	Reserved
35:32	RW	<p>Tx Retry Error Threshold If the number of independent retry events exceeds this threshold, Adapter must log this in 'Number of Retries Exceeded Threshold' and trigger Retrain on RDI. RDI state status going to Retrain also clears the internal count of independent retry events. Default value is 0h.</p>
63:36	RsvdP	Reserved

9.5.4.5 PHY Test/Compliance Register Block

Certain register bits described in this section take effect only when the PHY enters "PHY Compliance" mode. This mode is entered when bit 0 of 'Physical Layer Compliance Control 1' register is written and PHY subsequently enters PHYRETRAIN state. The latter happens when SW retrains the link. These register bits are tagged with @PHY-Compliance for easy readability and intuitive understanding.

Transition to TRAINERROR @PHY-Compliance does not reset any registers.

SW is required to place the Adapter in one of the Compliance modes (defined in the Adapter Compliance Control register) before enabling @PHY-Compliance.

All modules of a Link must be in @PHY-Compliance at the same time. The Link behavior is undefined if a subset of modules of a Link are in @PHY-Compliance and others are not. All registers in this section are replicated, one per module, as follows:

- Module 0 registers start at Offset 000h from PHYOFF
- Module 1 registers start at Offset 400h from PHYOFF
- Module 2 registers start at Offset 800h from PHYOFF
- Module 3 registers start at Offset C00h from PHYOFF

If certain modules are not implemented, those registers become reserved (as shown with gray boxes in Figure 9-6).

9.5.4.5.1 Physical Layer Compliance Control 1 (Offsets 000h, 400h, 800h, and C00h from PHYOFF)

Table 9-78. Physical Layer Compliance Control 1 (Sheet 1 of 2)

Bit	Attributes	Description
0	RW	Compliance Enable for Physical Layer Setting this bit to 1b puts the Physical Layer in "PHY Compliance" on the next entry into PHYRETRAIN state. Even if RDI status moves to Active, it does not assert p1_trdy to the Adapter in this mode. Default is 0b.
1	RW	Scrambling Disabled @PHY-Compliance, when set to 1b, Physical Layer disables scrambling. Default is 0b.
2	RW	PHY Compliance Operation Trigger @PHY-Compliance, transitioning this bit from 0b-to-1b starts one iteration of the Link training basic operations set by 'PHY Compliance Operation Type'. 'PHY Compliance Operation Type' field identifies which of the Link training basic operations is performed. 'Training Setup 1', 'Training Setup 2', 'Training Setup 3', and 'Training Setup 4' registers determine the parameters to be used for this. Default is 0b.

Table 9-78. Physical Layer Compliance Control 1 (Sheet 2 of 2)

Bit	Attributes	Description
5:3	RW	<p>PHY Compliance Operation Type @PHY-Compliance, where the Link training basic operation (see Section 4.5.1) is performed when 'PHY Compliance Operation Trigger' transitions from 0b to 1b</p> <ul style="list-style-type: none"> • 000b = No operation • 001b = Transmitter initiated Data-to-Clock point test (see Section 4.5.1.1) • 010b = Transmitter initiated Data-to-Clock eye width sweep (see Section 4.5.1.2) • 011b = Receiver initiated Data-to-Clock point training (see Section 4.5.1.3) • 100b = Receiver initiated Data-to-Clock width sweep training (see Section 4.5.1.4) • All other encodings are reserved
7:6	RsvdP	Reserved
9:8	RW	<p>Rx Vref Offset Enable @PHY-Compliance:</p> <ul style="list-style-type: none"> • 00b = No change to trained Rx Vref value • 01b = Add Rx Vref offset to trained Rx Vref value (up to maximum permitted Vref value) • 10b = Subtract Rx Vref offset to trained Rx Vref value (down to minimum permitted Rx Vref, any negative value to be terminated at 0) • 11b = Reserved
17:10	RW	<p>Rx Vref Offset @PHY-Compliance, when 'Rx Vref Offset Enable' is set to 01b or 10b, this is the value that needs to be added or subtracted as defined in 'Rx Vref Offset Enable'. The Rx Vref value, after applying the Rx Vref offset, is expected to be monotonically increasing/decreasing with increasing/decreasing values of Rx Vref offset relative to the trained value and must have sufficient range to cover the input eye mask range defined in Chapter 5.0. Rx Vref Offset will be applied during Tx or Rx Data to Point Training and the Physical Layer must compare the per Lane errors with 'Max error Threshold in per-Lane comparison', and aggregate Lane errors with 'Max Error Threshold in Aggregate Comparison' in the 'Training Setup 4' register. If the errors measured are greater than the corresponding threshold, then the device must set the Rx Vref offset status register to "failed". Software must increase or decrease the Rx Vref Offset by one from the previous value. Default is 00h.</p>
63:18	RsvdP	Reserved

9.5.4.5.2 Physical Layer Compliance Control 2 (Offsets 008h, 408h, 808h, and C08h from PHYOFF)

Table 9-79. Physical Layer Compliance Control 2

Bit	Attributes	Description
0	RW	Even UI Compare Mask @PHY-Compliance, if this bit is set, any compare results for even UIs are masked (i.e., not counted toward error in per Lane or aggregate comparison (see Section 4.4)), where Even UI refers as to a Unit Interval data eye, the first data UI and every subsequent alternate UI. <ul style="list-style-type: none"> • 0b = No even UI compare result masking • 1b = Even UI compare result masked Default is 0b.
1	RW	Odd UI Compare Mask @PHY-Compliance, if this bit is set, any compare results for odd UIs are masked (i.e., not counted toward error in per Lane or aggregate comparison (see Section 4.4)), where Odd UI refers as to a Unit Interval data eye, the second data UI and every subsequent alternate UI. <ul style="list-style-type: none"> • 0b = No odd UI compare result masking • 1b = Odd UI compare results masked Default is 0b.
2	RW	Track Enable If @PHY-Compliance { If this bit is set, Track Transmission is enabled during one of the operations set by 'PHY compliance operation type'. Track transmission complies with descriptions in Section 5.5.1 . } Else { The appropriate sideband handshakes as described in Section 4.6 needs to be followed irrespective of the value of this bit }
3	RW	Compare Setup <ul style="list-style-type: none"> • 0b = Aggregate comparison • 1b = Per Lane comparison Default is 0b. See Section 4.4 for more details.
31:4	RsvdP	Reserved

9.5.4.5.3 Physical Layer Compliance Status 1 (Offsets 010h, 410h, 810h, and C10h from PHYOFF)

Table 9-80. Physical Layer Compliance Status 1

Bit	Attributes	Description
0	RO	PHY in Compliance mode If (@PHY-Compliance) 1b. Else 0b.
1	RO	PHY Compliance operation status If (@PHY-Compliance) { This bit is set to 1b if 'PHY compliance operation type' in 'Physical Layer Compliance Control 1' register is 001b, 010b, 011b, or 100b and hardware has performed the required operation. Else the bit is cleared to 0b. }
3:2	RW1C	Rx Vref Offset Operation Status @PHY-Compliance: <ul style="list-style-type: none"> • 00b = Device does not support applying any Rx Vref Offset value • 01b = 'Rx Vref Offset' has not been applied • 10b = Rx Vref Offset has been successfully applied • 11b = Did not apply 'Rx Vref Offset' as the resulting value exceeds the value supported by hardware Default is 00b.
31:4	RsvdZ	Reserved

9.5.4.5.4 Physical Layer Compliance Status 2 (Offsets 018h, 418h, 818h, and C18h from PHYOFF)

Table 9-81. Physical Layer Compliance Status 2

Bit	Attributes	Description
31:0	RW1C	Aggregate Error Count @PHY-Compliance, this is the Error count of aggregate error comparison when 'PHY Compliance Operation Type' is 001b or 011b (performing point tests). Default is 0000 0000h.
39:32	RO	Supported Rx Vref Range Up Max step count supported up from the trained Rx Vref value for Vref margining.
47:40	RO	Supported Rx Vref Range Down Max step count supported down from the trained Rx Vref value for Vref margining.
55:48	RO	Trained Value for Rx Vref Rx Vref as trained, in resolution counts.
63:56	RO	Vref Step Count Resolution Increase in Vref value in mV between two consecutive encodings in ascending order.

9.5.4.5.5 Physical Layer Compliance Status 3 (Offsets 020h, 420h, 820h, and C20h from PHYOFF)

Table 9-82. Physical Layer Compliance Status 3

Bit	Attributes	Description
63:0	RO	<p>Per Lane Comparison Result Per Lane comparison result in PHY Compliance when 'PHY Compliance Operation Type' is 001b or 011b (performing point tests) and 'Comparison Setup' is 1b (Per Lane comparison)</p> <p>[63:0]: Compare Results of all Logical Data Lanes (0h Fail (Errors > Max Error Threshold), 1h Pass (Errors <= Max Error Threshold))</p> <p>UCIE-A {RD_L[63], RD_L[62], ..., RD_L[1], RD_L[0]}</p> <p>UCIE-A x32 {32'h0, RD_L[31], RD_L[30], ..., RD_L[1], RD_L[0]}</p> <p>UCIE-S {48'h0, RD_L[15], RD_L[14], ..., RD_L[1], RD_L[0]}</p> <p>Default is all 0s.</p>

9.5.5 Implementation Specific Register Blocks

These are left to be vendor defined. There is a separate implementation specific register Block for D2D Adapter and PHY. These register blocks should carry the same header as defined in [Table 9-26](#), at offset 0h of the register block. And the VendorID should be set to the specific vendor's ID and the 'VendorID register block' field set to 2h or 3h to indicate that it is a vendor specific register block. The other fields in that header are set by the vendor to track their revision number and the block length. Max length cannot exceed 1MB in size and length is always in multiples of 4KB. Implementations are highly encouraged to pack registers and reduce length of the region as much as possible.

9.6

UCIE Link Registers in Streaming Mode and System SW/FW Implications

IMPLEMENTATION NOTE

While the SW view of Protocol Layer for streaming protocols is implementation-specific, it is strongly recommended that UCIE link-related registers defined in this chapter be implemented as-is for streaming mode solutions as well. If a streaming mode solution chooses to support the industry-standard PCIe hierarchical tree model for enumeration/control, it must be compliant with the enumeration model and registers defined in this chapter. A UCIE port in such an implementation would expose UCIE link registers consistent with the RP/DSP or EP/USP functionality it represents.

In some streaming mode solutions, it might be desirable to implement UCIE link as a fully symmetric link, such as in a Symmetric Multi-Processing system that uses UCIE as a D2D interconnect. In such solutions, there is no notion of Upstream Port or Downstream Port on a UCIE link and also typically system firmware knows the D2D link connections a priori and it is able to configure them without requiring any "link discovery" mechanisms. It is recommended that both ends of the link implement UCIE registers defined for a Root Port, in such streaming mode solutions. Note that in this model, several link-related features become fully symmetric as well. For example, link training, mailbox trigger, and direct link-event/error reporting to Software are now possible from either end of the link. Whether such symmetric UCIE links are exposed to OS for native management, or system FW fully manages these links, is a system-architecture choice. Exposing such links natively to the OS could be in the form of exposing each side as an ACPI device or in the form of an FW intermediary that emulates a traditional PCIe hierarchical tree model for the symmetric link. Such choices are implementation-specific and could depend on the extent of OS support for symmetric topology.

9.7 MSI and MSI-X Capability in Hosts/Switches for UCIe Interrupt

Follow the base spec for details, but MSI/MSI-X capability implemented in host and switch must request 2 vectors for UCIe usage - 1 for Link status events and 1 for Link error events. Note that in MSI scenario, OS might not always allot both the requested vectors and in that case both the Link Status and Link error events use the same MSI vector number. The MSI designs must also support the Pending and Mask bits. MSI capability in UiRB must always set the 'Next Capability Pointer' field to 0h. SW must check for a value of 0005h in Bytes 0 and 1 of a capability to infer that it is an MSI capability. SW must terminate the capability linked list in UiRB when it sees the MSI Capability.

9.8 UCIe Early Discovery Table (UEDT)

Table 9-83. UEDT Header

Field	Byte Offset	Length in Bytes	Description
Signature	00h	4	Signature for the UCIe Early Discovery Table (UEDT).
Length	04h	4	Length, in bytes, of the entire UEDT.
Revision	08h	1	Value is 1h for the first UCIe instance.
Checksum	09h	1	Entire table must sum to 0.
OEM ID	0Ah	6	OEM ID
OEM Table ID	10h	8	Manufacturer Model ID
OEM Revision	18h	4	OEM Revision
Creator ID	1Ch	4	Vendor ID of the utility that created this table.
Creator Revision	20h	4	Revision of the utility that created this table.
UEDT Structure[n]	24h	Varies	A list of UEDT structures for this implementation. <ul style="list-style-type: none"> • 0h = UCIe Link structure (UCLS) • All other encodings are reserved

Table 9-84. UCIe Link Structure (UCLS) (Sheet 1 of 2)

Field	Byte Offset	Length in Bytes	Description
Type	00h	1	Signature for the UCIe Early Discovery Table (UEDT).
Revision	01h	1	Value is 1h for the first UCLS definition.
Record Length	02h	2	Length of this record, in bytes.
UID	04h	4	Host Bridge Unique ID. Used to associate a UCLS instance with a Host Bridge instance. The value of this field shall match the output of UID under the associated Host Bridge in ACPI namespace.
UCIe Stack Size	08h	4	<ul style="list-style-type: none"> • 1h = One RP • 2h = Two RPs
Reserved	0Ch	4	Reserved
Base	10h	8	Base address of UiRB, aligned to a 4-KB boundary.

Table 9-84. UCIe Link Structure (UCLS) (Sheet 2 of 2)

Field	Byte Offset	Length in Bytes	Description
Length	18h	8	Can range anywhere from 12 KB to 2 MB, in multiples of 4 KB.
DF1	20h	1	Device Function of the PCIe/CXL RP 1 associated with the UCLS.
DF2	21h	1	Device Function of the PCIe/CXL RP 2 (if multi-stack implementation) associated with the UCLS.

§ §

10.0 Interface Definitions

This chapter will cover the details of interface operation and signal definitions for the Raw Die-to-Die Interface (RDI), as well as the Flit-Aware Die-to-Die Interface (FDI). Common rules across RDI and FDI are covered as a separate section. The convention used in this chapter is that “assertion” of a signal is for 0b to 1b transition, and “de-assertion” of a signal is for 1b to 0b transition. A “pulse” of “n” cycles for a signal is defined as an event where the signal transitions from 0b to 1b, stays 1b for “n” clock cycles, and subsequently returns to 0b. A receiver sampling this signal on the same clock as the transmitter will see it being asserted for “n” clock cycles. If a value of “n” is not specified, it is interpreted as a value of one. In the context of error signals defined as pulses, the receiving logic for error logging must treat the rising edge as a new event indication and not rely on the length of the pulse.

In this chapter, interface reset/domain reset also applies to all forms of Conventional Reset defined in *PCIe Base Specification*, if the Protocol is PCIe or CXL. In the sections that follow, “UCIE Flit mode” refers to scenarios in which the Link is not operating in Raw Format, and “UCIE Raw Format” or “Raw Format” refers to scenarios in which the Link is operating in Raw Format.

10.1 Raw Die-to-Die Interface (RDI)

This section defines the signal descriptions and functionality associated with a single instance of Raw Die-to-Die Interface (RDI). A single instance could be used for a configuration associated with a single Die-to-Die module (i.e., one Die-to-Die Adapter for one module), or a single instance is also applicable for configurations where multiple modules are grouped together for a single logical Die-to-Die Link (i.e., one Die-to-Die Adapter for multiple modules). [Figure 10-1](#) shows example configurations using RDI.

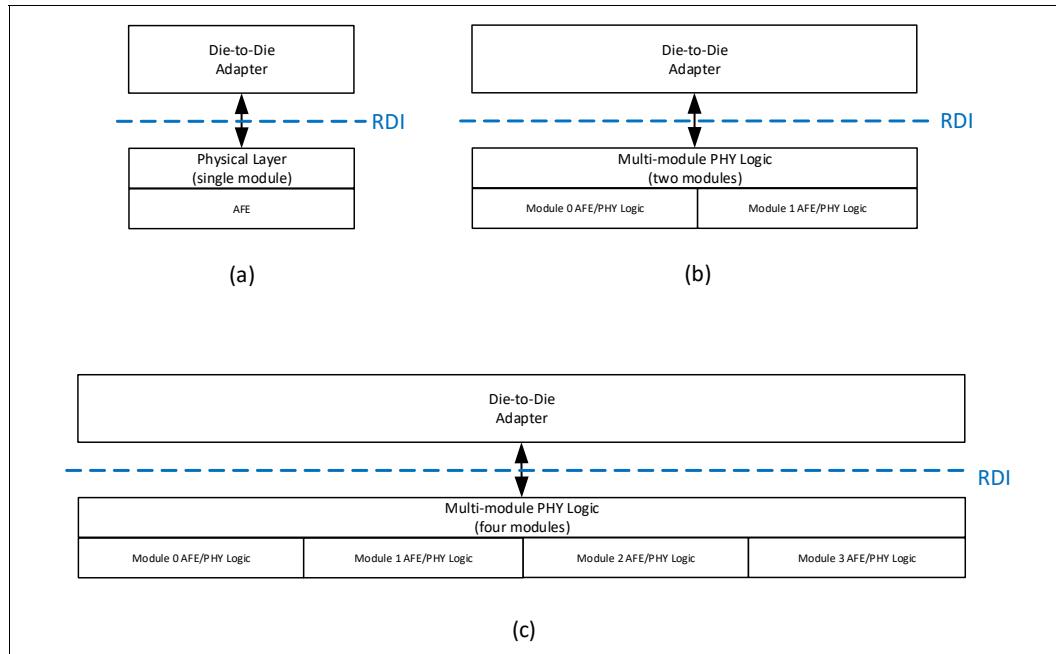
Figure 10-1. Example configurations using RDI

Table 10-1 lists the RDI signals and their descriptions. All signals are synchronous with **lclk**.

In Table 10-1:

- **pl_*** indicates that the signal is driven away from the Physical Layer to the Die-to-Die Adapter.
- **lp_*** indicates that the signal is driven away from the Die-to-Die Adapter to the Physical Layer.

Table 10-1. RDI signal list (Sheet 1 of 5)

Signal Name	Signal Description
lclk	The clock at which RDI operates.
lp_irdy	Adapter to Physical Layer signal indication that the Adapter has data to send. This must be asserted if lp_valid is asserted and the Adapter wants the Physical Layer to sample the data. lp_irdy must not be presented by the Adapter when pl_state_sts is Reset except when the status transitions from LinkError to Reset. On a LinkError to Reset transition, it is permitted for lp_irdy to be asserted for a few clocks but it must be de-asserted eventually. Physical Layer must ignore lp_irdy when status is Reset.
lp_valid	Adapter to Physical Layer indication that data is valid on the corresponding lp_data bytes.
lp_data[NBYTES-1:0][7:0]	Adapter to Physical Layer data, where 'NBYTES' equals number of bytes determined by the data width for the RDI instance.
lp_retimer_crd	When asserted at a rising clock edge, it indicates a single credit return from the Adapter to the Physical Layer for the Retimer Receiver buffers. Each credit corresponds to 256B of mainband data. This signal must NOT assert for dies that are not UCIE Retimers.
pl_trdy	The Physical Layer is ready to accept data. Data is accepted by the Physical Layer when pl_trdy , lp_valid , and lp_irdy are asserted at the rising edge of lclk . This signal must only be asserted if pl_state_sts is Active or when performing the pl_stallreq/lp_stallack handshake when the pl_state_sts is LinkError (see Section 10.3.3.7).

Table 10-1. RDI signal list (Sheet 2 of 5)

Signal Name	Signal Description
pl_valid	Physical Layer to Adapter indication that data is valid on pl_data .
pl_data[NBYTES-1:0][7:0]	Physical Layer to Adapter data, where NBYTES equals the number of bytes determined by the data width for the RDI instance.
pl_retimer_crd	When asserted at a rising clock edge, it indicates a single credit return from the Retimer to the Adapter. Each credit corresponds to 256B of mainband data. This signal must NOT assert if the remote Link partner is not a Retimer.
lp_state_req[3:0]	Adapter request to Physical Layer to request state change. Encodings as follows: 0000b: NOP 0001b: Active 0100b: L1 1000b: L2 1001b: LinkReset 1011b: Retrain 1100b: Disabled All other encodings are reserved.
lp_linkerror	Adapter to Physical Layer indication that an error has occurred which requires the Link to go down. Physical Layer must move to LinkError state and stay there as long as lp_linkerror=1 . The reason for having this be an indication decoupled from regular state transitions is to allow immediate action on part of the Adapter and Physical Layer in order to provide the quickest path for error containment when applicable (for example, a viral error escalation must map to the LinkError state). The Adapter must OR internal error conditions with lp_linkerror received from Protocol Layer on FDI.
pl_state_sts[3:0]	Physical Layer to Adapter Status indication of the Interface. Encodings as follows: 0000b: Reset 0001b: Active 0011b: Active.PMNAK 0100b: L1 1000b: L2 1001b: LinkReset 1010b: LinkError 1011b: Retrain 1100b: Disabled All other encodings are reserved. The status signal is permitted to transition from Physical Layer autonomously when applicable. For example the Physical Layer asserts the Retrain status when it decides to enter retraining either autonomously or when requested by remote agent.
pl_inband_pres	Physical Layer to the Adapter indication that the Die-to-Die Link has finished training and is ready for RDI transition to Active and Stage 3 of bring up. Once it transitions to 1b, this must stay 1b until Physical Layer determines the Link is down (i.e., the Link Training State Machine transitions to TrainError or Reset).

Table 10-1. RDI signal list (Sheet 3 of 5)

Signal Name	Signal Description
pl_error	<p>Physical Layer to the Adapter indication that it has detected a framing related error which is recoverable through Link Retrain. An example is where the Physical Layer received an invalid encoding on the Valid Lane. It is a pulse of one or more cycles that must occur only when RDI is in Active state. It is permitted to de-assert at the same clock edge where the state transitions away from Active state.</p> <p>It is pipelined with the receive data path such that the error indication reaches the Adapter before or at the same time as the corrupted data. Physical Layer is expected to go through Retrain flow after this signal has been asserted and it must not send valid data to Adapter until the Link has retrained.</p> <p>It is permitted for the Physical Layer to squash the pl_valid internally for the corrupted data. Once pl_error is asserted, pl_valid should not be asserted (without pl_error assertion in the same cycle) until the state status has transitioned to Active after completing a successful Retrain entry and exit.</p> <p>If pl_error=1 and pl_valid=1 in the same clock cycle, the Adapter must discard the corresponding Flit (even if it is only partially received when pl_error asserted). In UCIE Flit mode, when retry is enabled, it is the responsibility of the Adapter to ensure data integrity for Flits forwarded to FDI, and that they are canceled following the rules of pl_filit_cancel if they are suspected of corruption (see Section 10.2). A couple of examples are given below:</p> <ul style="list-style-type: none"> • For 68B Flit Format, the Adapter could discard partially received Flits, but in 256B Latency optimized modes, it could have processed one half correctly, and the error may have happened on the other half, and so it has to track that and process future flits accordingly. • Another example is if it is not doing store/forward and only received 64B of a 128B half, and pl_error happened before receiving the remaining 64B of the 128B half, it needs to send dummy data for the second 64B and do a pl_filit_cancel for that half of the Flit. <p>In UCIE Flit mode with Retry enabled for the Adapter, Retrain exit would naturally result in a Replay of any partially received Flits eventually (see Section 3.8).</p> <p>In UCIE Flit mode with Retry disabled, the Adapter must map pl_error assertion to an Uncorrectable Internal Error and escalate it accordingly.</p> <p>If the Link is operating in Raw Format, the Adapter forwards pl_error to the Protocol Layer such that it is pipeline matched to the data bus, and Protocol Layer handles it in an implementation-specific manner.</p>
pl_cerror	<p>Physical Layer to the Adapter indication that a correctable error was detected that does not affect the data path and will not cause Retrain on the Link. In UCIE Flit mode with Retry enabled, the Adapter must OR the pl_error and pl_cerror signals for Correctable Internal Error Logging.</p> <p>In UCIE Flit mode with Retry disabled or when the Link is operating in Raw Format, the Adapter must only use pl_cerror for Correctable Internal Error Logging.</p> <p>It is a pulse of one or more cycles which can occur in any RDI state. If it is a state in which clock gating is permitted, it is the responsibility of the Physical Layer to perform the clock gating exit handshake with the Adapter before asserting this signal. Clock gating can resume once pl_cerror de-asserts and all other conditions permitting clock gating are satisfied.</p>
pl_nferror	<p>Physical Layer to the Adapter indication that a non-fatal error was detected. There is no architecturally defined error condition for the Physical Layer currently asserting this signal; however, the signal is provided on the interface for any implementation-specific non-fatal errors. The Adapter treats this in the same manner as when it received a Sideband Non-Fatal Error Message from the remote Link partner.</p> <p>It is a pulse of one or more cycles that can occur in any RDI state. If it is a state where clock gating is permitted, it is the responsibility of the Physical Layer to perform the clock gating exit handshake with the Adapter before asserting this signal. Clock gating can resume after pl_nferror is de-asserted and all other conditions permitting clock gating have been met.</p>
pl_trainerror	<p>Indicates a fatal error from the Physical Layer. Physical Layer must transition pl_state_sts to LinkError if not already in LinkError state.</p> <p>This must be escalated to upper Protocol Layers based on the mask and severity programming of Uncorrectable Internal Error in the Adapter. Implementations are permitted to map any fatal error to this signal that require upper layer escalation (or interrupt generation) depending on system-level requirements.</p> <p>It is a level signal that can assert in any RDI state but remains asserted until RDI exits the LinkError state to Reset state.</p>

Table 10-1. RDI signal list (Sheet 4 of 5)

Signal Name	Signal Description
pl_physinrecenter	Physical Layer indication to Adapter that the Physical Layer is training or retraining. If this is asserted during a state where clock gating is permitted, the pl_clk_req / lp_clk_ack handshake must be performed with the upper layer. The upper layers are permitted to use this to update the "Link Training/Retraining" bit in the PCIe Link Status register.
pl_stallreq	Physical Layer request to Adapter to align Transmitter at Flit boundary and not send any new Flits to prepare for state transition. See Section 10.3.2 .
lp_stallack	Adapter to Physical Layer indication that the Flits are aligned and stalled (if pl_stallreq was asserted). It is strongly recommended that this response logic be on a global free running clock, so the Adapter can respond to pl_stallreq with lp_stallack even if other significant portions of the Adapter are clock gated. See Section 10.3.2 .
pl_speedmode[2:0]	Current Link speed. The following encodings are used: 000b: 4GT/s 001b: 8GT/s 010b: 12GT/s 011b: 16GT/s 100b: 24GT/s 101b: 32GT/s other encodings are reserved. The Adapter must only consider this signal to be relevant when the RDI state is Active or Retrain. For multi-module configurations, all modules must operate at the same speed.
pl_lnk_cfg[2:0]	Current Link Configuration. Indicates the current operating width of a module. 000b: x4 001b: x8 010b: x16 011b: x32 100b: x64 101b: x128 110b: x256 other encodings are reserved. This is the width of the PCIe physical die-to-die Link which may be composed of one to four modules. For PCIe-S the maximum encoding would be x64, for PCIe-A the maximum encoding would be x128 for PCIe-A x32 and x256 for PCIe-A x64. The Adapter must only consider this signal to be relevant when the RDI state is Active or Retrain. This signal indicates the total width across all Active Modules corresponding to the RDI instance.
pl_clk_req	Request from the Physical Layer to remove clock gating from the internal logic of the Adapter. This is an asynchronous signal relative to lclk from the Adapter's perspective since it is not tied to lclk being available in the Adapter. Together with lp_clk_ack , it forms a four-way handshake to enable dynamic clock gating in the Adapter. When dynamic clock gating is supported, the Adapter must use this signal to exit clock gating before responding with lp_clk_ack . If dynamic clock gating is not supported, it is permitted for the Physical Layer to tie this signal to 1b.
lp_clk_ack	Response from the Adapter to the Physical Layer acknowledging that its clocks have been un gated in response to pl_clk_req . This signal is only asserted when pl_clk_req is asserted, and de-asserted after pl_clk_req has de-asserted. When dynamic clock gating is not supported by the Adapter, it must stage pl_clk_req internally for one or more clock cycles and turn it around as lp_clk_ack . This way it will still participate in the handshake even though it does not support dynamic clock gating.

Table 10-1. RDI signal list (Sheet 5 of 5)

Signal Name	Signal Description
<code>lp_wake_req</code>	<p>Request from the Adapter to remove clock gating from the internal logic of the Physical Layer. This is an asynchronous signal from the Physical Layer's perspective since it is not tied to <code>lclk</code> being available in the Physical Layer. Together with <code>pl_wake_ack</code>, it forms a four-way handshake to enable dynamic clock gating in the Physical Layer.</p> <p>When dynamic clock gating is supported, the Physical Layer must use this signal to exit clock gating before responding with <code>pl_wake_ack</code>.</p> <p>If dynamic clock gating is not supported, it is permitted for the Adapter to tie this signal to 1b.</p>
<code>pl_wake_ack</code>	<p>Response from the Physical Layer to the Adapter acknowledging that its clocks have been un gated in response to <code>lp_wake_req</code>. This signal is only asserted after <code>lp_wake_req</code> has asserted, and is de-asserted after <code>lp_wake_req</code> has de-asserted.</p> <p>When dynamic clock gating is not supported by the Physical Layer, it must stage <code>lp_wake_req</code> internally for one or more clock cycles and turn it around as <code>pl_wake_ack</code>. This way it will still participate in the handshake even though it does not support dynamic clock gating.</p>
<code>pl_cfg[NC-1:0]</code>	<p>This is the sideband interface from the Physical Layer to the Adapter. See Chapter 7.0 for packet format details. NC is the width of the interface. Supported values are 8, 16, and 32.</p> <p>Register accesses must be implemented by hardware to be atomic regardless of the width of the interface (i.e., all 32 bits of a register must be updated in the same cycle for a 32-bit register write, and similarly all 64 bits of a register must be updated in the same cycle for a 64-bit register write).</p>
<code>pl_cfg_vld</code>	When asserted, indicates that <code>pl_cfg</code> has valid information that should be consumed by the Adapter.
<code>pl_cfg_crd</code>	<p>Credit return for sideband packets from the Physical Layer to the Adapter for sideband packets. Each credit corresponds to 64 bits of header and 64 bits of data. Even transactions that do not carry data or carry 32 bits of data consume the same credit and the Physical Layer returns the credit once the corresponding transaction has been processed or deallocated from its internal buffers. See Section 7.1.3.1 for additional flow control rules. A value of 1 sampled at a rising clock edge indicates a single credit return.</p> <p>Because the advertised credits are design parameters, the Adapter transmitter updates the credit counters with initial credits on domain reset exit, and no initialization credits are returned over the interface.</p> <p>Credit returns must follow the same rules of clock gating exit handshakes as the sideband packets to ensure that no credit returns are dropped by the receiver of the credit returns.</p>
<code>lp_cfg[NC-1:0]</code>	<p>This is the sideband interface from Adapter to the Physical Layer. See Chapter 7.0 for details. NC is the width of the interface. Supported values are 8, 16, and 32.</p> <p>Register accesses must be implemented by hardware to be atomic regardless of the width of the interface (i.e., all 32 bits of a register must be updated in the same cycle for a 32-bit register write, and similarly all 64 bits of a register must be updated in the same cycle for a 64-bit register write).</p>
<code>lp_cfg_vld</code>	When asserted, indicates that <code>lp_cfg</code> has valid information that should be consumed by the Physical Layer.
<code>lp_cfg_crd</code>	<p>Credit return for sideband packets from the Adapter to the Physical Layer for sideband packets. Each credit corresponds to 64 bits of header and 64 bits of data. Even transactions that do not carry data or carry 32 bits of data consume the same credit and the Adapter returns the credit once the corresponding transaction has been processed or deallocated from its internal buffers. See Section 7.1.3.1 for additional flow control rules. A value of 1 sampled at a rising clock edge indicates a single credit return.</p> <p>Because the advertised credits are design parameters, the Physical Layer transmitter updates the credit counters with initial credits on domain reset exit, and no initialization credits are returned over the interface.</p> <p>Credit returns must follow the same rules of clock gating exit handshakes as the sideband packets to ensure that no credit returns are dropped by the receiver of the credit returns.</p>

Signals in Table 10-2 apply only when supporting MPM over sideband. The choice for whether these signals run on the **lclk** or the **Mgmt_Clk** is implementation-specific.

Table 10-2. RDI Config interface extensions for Management Transport (Sheet 1 of 3)

Signal Name	Signal Description
pm_param_done	Management transport negotiation phase completed. Signal de-asserts after being asserted for two clocks. This signal asserts when MBINIT.PARAM management transport negotiation phase completes. Note that this signal is asserted even if MBINIT.PARAM Configuration or SBFE exchanges indicate no support for management transport in the partner chiplet.
pm_param_local_count[N-1:0]	Number of modules that successfully negotiated Management transport on transmit side. This field is sampled only when pm_param_done signal is asserted. 000b: 0 modules 001b: 1 module 010b: 2 modules 011b: 3 modules 100b: 4 modules Others: Reserved N=2 for 1, 2, or 3 modules scenarios, and N=3 for 4 modules scenario.
pm_param_remote_count[N-1:0]	Number of modules that successfully negotiated Management transport on receive side. This field is sampled only when pm_param_done signal is asserted. 000b: 0 modules 001b: 1 module 010b: 2 modules 011b: 3 modules 100b: 4 modules Others: Reserved N=2 for 1, 2, or 3 modules scenarios, and N=3 for 4 modules scenario.
mp_mgmt_init_done	Indication from Management Port Gateway that initialization phase completed (successfully or unsuccessfully). This signal is used by the PHY to advance the state machine state beyond MBINIT.PARAM, if other conditions allow. Signal de-asserts after being asserted for two clocks. The PHY should not depend on this signal for advancing the state machine when the management path is already up or when the partner chiplet indicated no support for management transport.
mp_mgmt_init_start	A two-clock trigger pulse from Management Port Gateway to PHY to start negotiation on the sideband links. Management Port Gateway must ensure that the mp_mgmt_up signal is de-asserted when this signal is pulsed. This signal forces the link state machine to RESET state (if it is not already there) and hence can bring the mainband link down from link up state. The standard TRAINERROR flow applies here as well for transitioning the state machine to RESET if the state machine is not already in that state when this signal is pulsed.
mp_mgmt_up	Indication from Management Port Gateway that is signaled along with mp_mgmt_init_done , that Management Transport Initialization Phase completed successfully (mp_mgmt_up=1) or unsuccessfully (mp_mgmt_up=0). This is used by PHY to set the SB_MGMT_UP flag.
mp_mgmt_port_gateway_ready	Indication to PHY that Management Port Gateway is ready for management transport path initialization. The PHY uses this as one of the conditions to trigger or respond to a trigger for Management Transport path initialization. This asserts after the Management Port Gateway is ready for management path setup after Management Reset. Once asserted, this signal de-asserts on a Management Port Gateway reset (either because of management domain reset or after a heartbeat timeout or an 'Init Done' Timeout or any fatal error on sideband) condition.
mp_stall_after_mbinit_param	Management Port Gateway asserts this signal concurrent with asserting mp_mgmt_port_gateway_ready to indicate to the PHY that it must stall the training on the receive side using an {MBINIT.PARAM SBFE resp} sideband message stall encoding as described in Section 4.5.3.3.1.2 . This signal remains asserted until the MPG determines that stalling is no longer necessary (i.e., that it is okay for link initialization to proceed). When de-asserted, the receive side training does not cause a "stall". When a sideband-only link is negotiated, this signal is not used by the PHY to determine the Link training state machine progress.

Table 10-2. RDI Config interface extensions for Management Transport (Sheet 2 of 3)

Signal Name	Signal Description
<code>pm_cfg_credit[N-1:0]</code>	This is credit return for the Flow control buffers over RDI (see Section 8.2.5.1.1) used by the Management Port Gateway to transmit management packets to the remote Management Port Gateway. Each credit corresponds to 64 bits of buffer space. Physical Layer returns the credit once the corresponding transaction has been deallocated from its internal buffers. See Section 8.2.5.1.1 for additional flow control rules. Because the advertised credits are design parameters, the Management Port Gateway transmitter updates the credit counters with initial credits on Management reset exit or on 'Heartbeat timeout', and no initialization credits are returned over the interface for these conditions. Credit returns must follow the same rules of clock gating exit handshakes as the sideband packets to ensure that no credit returns are dropped by the receiver of the credit returns. There is a signal per RxQ-ID in the design and hence N can be 1, 2, 3, or 4.
<code>mp_rxqid[N-1:0]</code>	RxQ-ID associated with the message. Has meaning when <code>mp_mgmt_pkt</code> signal is asserted on a RDI transfer. Used by PHY to steer the packet to the correct SB link. On encapsulated MTPs and PM Req messages, this carries the far-end Rx queue's RxQ-ID. On Credit return, Init Done and PM Ack messages this carries the RxQ-ID of the local Rx queue associated with the message. N is either 2 (for 4 modules links scenarios) or 1 (1 or 2 modules links scenarios). There is a fixed mapping in the PHY between this value and a physical SB link and the mapping is determined post successful completion of management transport negotiation on the transmit side. The chosen SB link for a given RxQ-ID must be one of the SB links that successfully trained for management transport on the transmit side.
<code>pm_rxqid[N-1:0]</code>	RxQ-ID associated with the message. Has meaning when <code>pm_mgmt_pkt</code> signal is asserted on a RDI transfer. Used by Management Port Gateway to internally steer the packet to the correct RxQ. N is either 2 (for 4 modules/sideband-only links scenarios) or 1 (1 or 2 modules/sideband-only links scenarios). Valid for all MPM config bus transmissions. PHY uses the RxQ-ID from the first credit return message received from a given sideband link to drive these signals on config interface. These signals are undefined for SoC Capabilities message. The captured RxQ-ID value is reset only when the management path is reinitialized.
<code>mp_wake_req</code>	Request from the Management Port Gateway to remove clock gating from the internal logic of the Physical Layer that handles management transport traffic. This is an asynchronous signal from the Physical Layer's perspective since it is not tied to lclk being available in the Physical Layer. Together with <code>pm_wake_ack</code> , it forms a four-way handshake to enable dynamic clock gating in the Physical Layer for logic that handles management transport traffic. When dynamic clock gating is supported, the Physical Layer must use this signal to exit clock gating before responding with <code>pm_wake_ack</code> . If dynamic clock gating is not supported, Management Port Gateway must tie this signal to 1.
<code>pm_wake_ack</code>	Response from the Physical Layer to the Management Port Gateway acknowledging that its clocks have been un gated in response to <code>mp_wake_req</code> . This signal is only asserted after <code>mp_wake_req</code> has asserted, and is de-asserted after <code>mp_wake_req</code> has de-asserted. When dynamic clock gating is not supported by the Physical Layer, it must stage <code>mp_wake_req</code> internally for one or more clock cycles and turn it around as <code>pm_wake_ack</code> . This way it will still participate in the handshake even though it does not support dynamic clock gating.
<code>pm_clk_req</code>	Request from the Physical Layer to remove clock gating from the internal logic of the Management Port Gateway. This is an asynchronous signal relative to <code>lclk/Mgmt_clk</code> from the Management Port Gateway perspective because it is not tied to <code>lclk/Mgmt_clk</code> being available in the Management Port Gateway. Together with <code>mp_clk_ack</code> , it forms a four-way handshake to enable dynamic clock gating in the Management Port Gateway. When dynamic clock gating is supported, the Management Port Gateway must use this signal to exit clock gating before responding with <code>mp_clk_ack</code> . If dynamic clock gating is not supported, Physical Layer must tie this signal to 1.

Table 10-2. RDI Config interface extensions for Management Transport (Sheet 3 of 3)

Signal Name	Signal Description
mp_clk_ack	Response from the Management Port Gateway to the PHY acknowledging that its clocks have been un gated in response to pm_clk_req . This signal is asserted only when pm_clk_req is asserted, and de-asserted after pm_clk_req has de-asserted. When dynamic clock gating is not supported by the Management Port Gateway, it must stage pm_clk_req internally for one or more clock cycles and turn it around as mp_clk_ack . This way it will still participate in the handshake even though it does not support dynamic clock gating. When supporting dynamic clock gating of the Management Port Gateway, PHY must ensure that pulsed signals (e.g., pm_param_done), are delivered only after the mp_clk_ack is set to ensure that the Management Port Gateway saw those pulses.
mp_mgmt_pkt	During a valid RDI data transfer to PHY, this signal indicates whether the transfer is for an MPM. 0: Link management packet. 1: MPM. Used by PHY to steer the packet to the correct RDI credit buffer.
pm_mgmt_pkt	During a valid RDI data transfer from PHY, this signal indicates whether the transfer is for an MPM. 0: Link management packet. 1: MPM. Used by the Management Port Gateway to steer the packet to RxQ buffers or to D2D Adapter.
pm_so	When asserted, indicates to Management Port Gateway that SO mode was negotiated. On ports that have sideband-only link physically present, this can be tied off to 1.
Mgmt_clk	Optional clock used for the Configuration interface on the RDI for implementations in which the main RDI clock is not available for Management Transport path initialization.
pm_fatal_error	Set by any sideband link fatal error indication, such as parity error on a sideband packet. Cleared by a Management Reset.
mp_fatal_error	Used by Management Port Gateway to instruct the PHY to transition to TRAINERROR state. This is a two-clock pulse.

10.1.1 Interface reset requirements

RDI does not define a separate interface signal for reset; however, it is required that the logic entities on both sides of RDI are in the same reset domain and the reset for each side is derived from the same source. Because reset may be staggered due to SoC routing, all signals coming out of reset must be driven to 0, unless otherwise specified.

10.1.2 Interface clocking requirements

RDI requires both sides of the interface to be on the same clock domain. The clock domain for the sideband interface (***cfg***) is the same as the mainband signals when Management Transport is not supported. When Management Transport is supported, the sideband interface is permitted to be on a separate **Mgmt_clk** domain.

Each side is permitted to internally instantiate clock-crossing FIFOs if needed, as long as it does not violate the requirements at the interface itself.

It is important to note that back pressure is not possible from the Adapter to the Physical Layer on the main data path. So any clock-crossing-related logic internal to the Adapter must take this into consideration.

For example, for a 64-Lane module with a maximum speed of 16 GT/s, the RDI could be 64B wide running at 2 GHz to be exactly bandwidth matched.

10.1.3 Dynamic clock gating

Dynamic coarse clock gating is permitted in the Adapter and Physical Layer when `pl_state_sts` is Reset, LinkReset, Disabled, or PM. This section defines the rules around entry and exit of clock gating. Note that clock gating is not permitted in LinkError state; it is expected that for UCIe usages, error handlers will be enabled to make sure the Link is not stuck in LinkError state if the intent is save power for Links in error state.

10.1.3.1 Rules and description for `lp_wake_req/pl_wake_ack` handshake

Adapter can request removal of clock gating of the Physical Layer by asserting `lp_wake_req` (asynchronous to `lclk` availability in the Physical Layer). All Physical Layer implementations must respond with a `pl_wake_ack` (synchronous to `lclk`). The extent of internal clock ungating when `pl_wake_ack` is asserted is implementation-specific, but `lclk` must be available by this time to enable RDI signal transitions from the Adapters. The Wake Req/Ack is a full handshake and it must be used for state transition requests (on `lp_state_req` or `lp_linkerror`) when moving away from a state in which clock gating is permitted. It must also be used for sending packets on the sideband interface.

Rules for this handshake:

1. Adapter asserts `lp_wake_req` to request ungating of clocks by the Physical Layer.
2. The Physical Layer asserts `pl_wake_ack` to indicate that clock gating has been removed. There must be at least one clock cycle bubble between `lp_wake_req` assertion and `pl_wake_ack` assertion.
3. `lp_wake_req` must de-assert before `pl_wake_ack` de-asserts. It is the responsibility of the Adapter to control the specific scenario of de-assertion. As an example, when performing the handshake for a state request, it is permitted to keep `lp_wake_req` asserted until it observes the desired state status. Adapter is also permitted to keep `lp_wake_req` asserted through states where clock gating is not permitted in the Physical Layer (i.e., Active, LinkError, or Retrain).
4. `lp_wake_req` should not be the only consideration for Physical Layer to perform clock gating, it must take into account `pl_state_sts` and other internal or Link requirements before performing global and/or local clock gating.
5. When performing `lp_wake_req/pl_wake_ack` handshake for `lp_state_req` transitions or `lp_linkerror` transition, the Adapter is permitted to not wait for `pl_wake_ack` before changing `lp_state_req` or `lp_linkerror`.
6. When performing `lp_wake_req/pl_wake_ack` handshake for `lp_cfg` transitions, Adapter must wait for `pl_wake_ack` before changing `lp_cfg` or `lp_cfg_vld`. Because `lp_cfg` can have multiple transitions for a single packet transfer, it is necessary to make sure that the Physical Layer clocks are up before transfer begins.

10.1.3.2 Rules and description for `pl_clk_req/lp_clk_ack` handshake

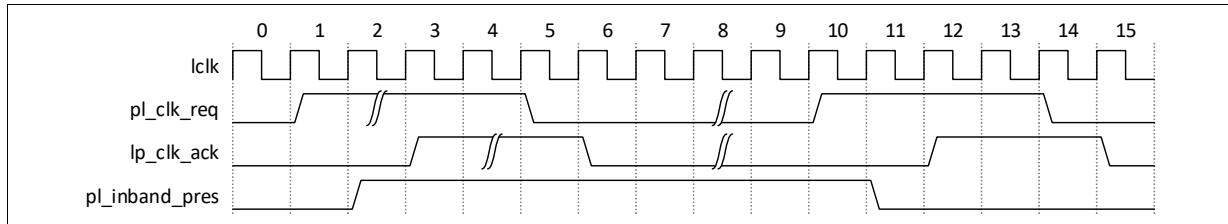
Physical Layer is permitted to initiate `pl_clk_req/lp_clk_ack` handshake at any time and the Adapter must respond.

Rules for this handshake:

1. Physical Layer asserts `pl_clk_req` to request removal of clock gating by the Adapter. This can be done anytime, and independent of current RDI state.
2. The Adapter asserts `lp_clk_ack` to indicate that clock gating has been removed. There must be at least one clock cycle bubble between `pl_clk_req` assertion and `lp_clk_ack` assertion.

3. **pl_clk_req** must de-assert before **lp_clk_ack**. It is the responsibility of the Physical Layer to control the specific scenario of de-assertion, after the required actions for this handshake are completed.
4. **pl_clk_req** should not be the only consideration for the Adapter to perform clock gating, it must take into account **pl_state_sts** and other protocol-specific requirements before performing trunk and/or local clock gating.
5. The Physical Layer must use this handshake to ensure transitions of **pl_inband_pres** have been observed by the Adapter. Since **pl_inband_pres** is a level oriented signal (once asserted it stays asserted during the lifetime of Link operation), the Physical Layer is permitted to let the signal transition without waiting for **lp_clk_ack**. When this is done during initial Link bring up, it is strongly recommended for the Physical Layer to keep **pl_clk_req** asserted until the state status transitions away from Reset to a state where clock gating is not permitted.

Figure 10-2. Example Waveform Showing Handling of Level Transition

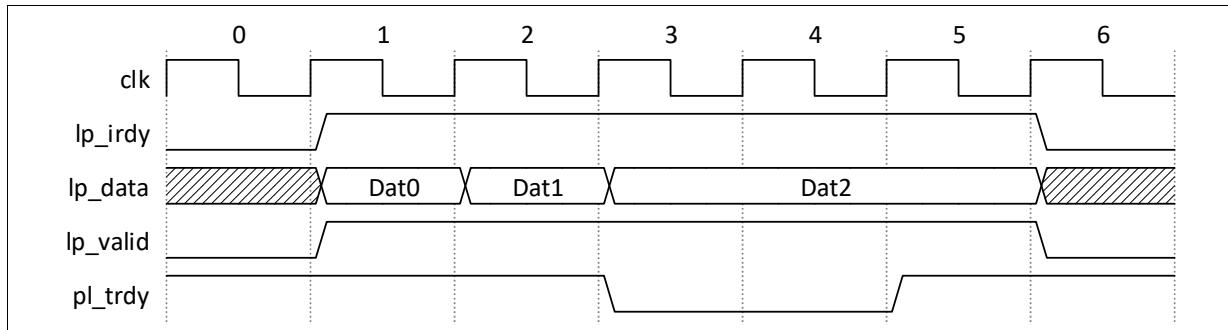


6. The Physical Layer must also perform this handshake before transition to LinkError state from Reset or PM state (when the LinkError transition occurs by the Physical Layer without being directed by the Adapter). It is permitted to assert **pl_clk_req** before the state change, in which case it must stay asserted until the state status transitions. It is also permitted to assert **pl_clk_req** after the state status transition, but in this case Physical Layer must wait for **lp_clk_ack** before performing another state transition.
7. The Physical Layer must also perform this handshake when the status is PM and remote Link partner is requesting PM exit. For exit from Reset or PM states to a state that is not LinkError, it is required to assert **pl_clk_req** before the status change, and in this case it must stay asserted until the state status transitions away from Reset or PM.
8. When clock-gated in RESET states, Adapters that rely on dynamic clock gating to save power must wait in clock gated state for **pl_inband_pres**=1. The Physical Layer will request clock gating exit when it transitions **pl_inband_pres**, and the Adapter must wait for **pl_inband_pres** assertion before requesting **lp_state_req** = ACTIVE. If **pl_inband_pres** de-asserts while **pl_state_sts** = RESET, then the Adapter is permitted to return to clock-gated state after moving **lp_state_req** to NOP.
9. Physical Layer must also perform this handshake for sideband traffic to Adapter. When performing the handshake for **pl_cfg** transitions, Physical Layer must wait for **lp_clk_ack** before changing **pl_cfg** or **pl_cfg_vld**. Because **pl_cfg** can have multiple transitions for a single packet transfer, it is necessary to make sure that the Adapter clocks are up before transfer begins.

10.1.4 Data Transfer

As indicated in the signal list descriptions, when Adapter is sending data to the Physical Layer, data is transferred when **lp_irdy**, **pl_trdy**, and **lp_valid** are asserted. Figure 10-3 shows an example waveform for data transfer from the Adapter to the Physical Layer. Data is transmitted on clock cycles 1, 2, and 5. No assumption should be made by Adapter about when **pl_trdy** can de-assert or for how many cycles it remains de-asserted before it is asserted again, unless explicitly guaranteed by the Physical Layer. If a Flit transfer takes multiple clock cycles, the Adapter is not permitted to insert bubbles in the middle of a Flit transfer. This means that **lp_valid** and **lp_irdy** must be asserted continuously until the Flit transfer is complete. Of course, data transfer can stall because of **pl_trdy** de-assertion.

Figure 10-3. Data Transfer from Adapter to Physical Layer



As indicated in the signal list descriptions, when the Physical Layer is sending data to the Adapter, there is no backpressure mechanism, and data is transferred whenever **pl_valid** is asserted. The Physical Layer is permitted to insert bubbles in the middle of a Flit transfer and the Adapter must be able to handle that.

IMPLEMENTATION NOTE

For the transmit side of the Physical Layer for data sent over the UCIe Link, it must ensure that if the Adapter has a continuous stream of packets to transmit (**lp_irdy** and **lp_valid** do not de-assert), it does not insert bubbles in valid frames on the Physical Link.

For the Runtime Link Testing feature with parity insertion, the Adapter as a receiver of parity bytes is permitted to issue a {ParityFeature.Nak} if software sets up a number of parity byte insertions ("Number of 64 Byte Inserts" field in the "Error and Link Testing Control" register) that does not amount to 256B or a multiple of the RDI width (to save the implementation cost of barrel shifting the parity bytes). For example, if the RDI width is 64B then either 64B, 128B, or 256B of inserted parity bytes are okay, but if the RDI width is 256B or larger, then it is better to always have 256B of inserted parity bytes so that it matches the data transfer granularity of Flits.

IMPLEMENTATION NOTE

It is permitted to use **lp_irdy** as an early indication that the valid data will be resuming imminently, and the Physical Layer needs to ungate clocks and assert **pl_trdy** when it is ready to receive data. A couple of examples are shown in [Figure 10-4](#) and [Figure 10-5](#). Note that **pl_trdy** could have asserted as early as Clock Cycle 1 in [Figure 10-4](#).

Figure 10-4. **lp_irdy** asserting two cycles before **lp_valid**

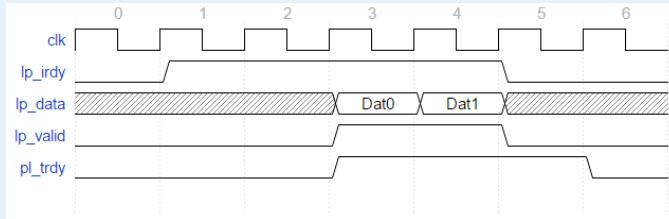
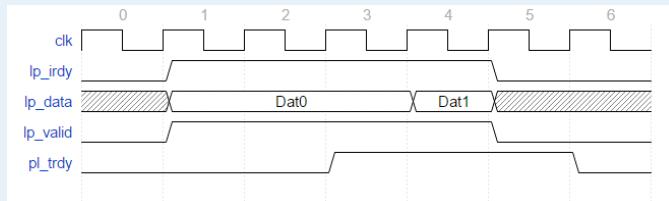


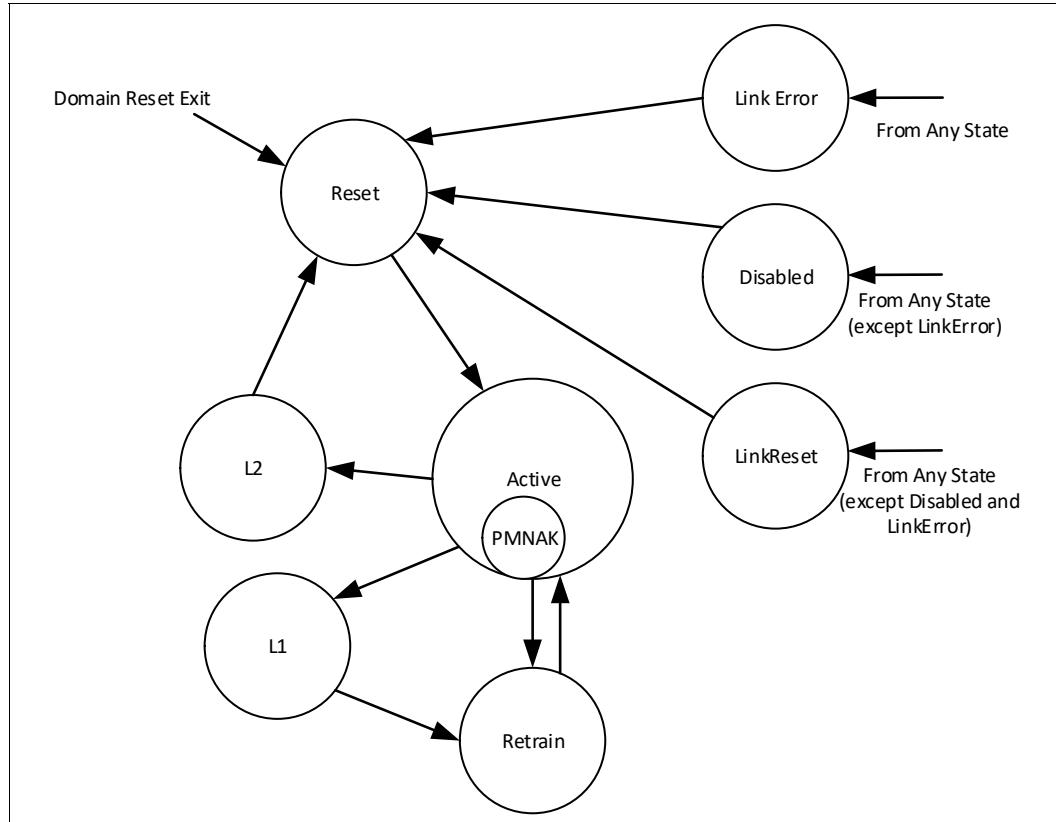
Figure 10-5. **lp_irdy** asserting at the same cycle as **lp_valid**



10.1.5 RDI State Machine

Figure 10-6 shows the RDI state machine.

Figure 10-6. RDI State Machine



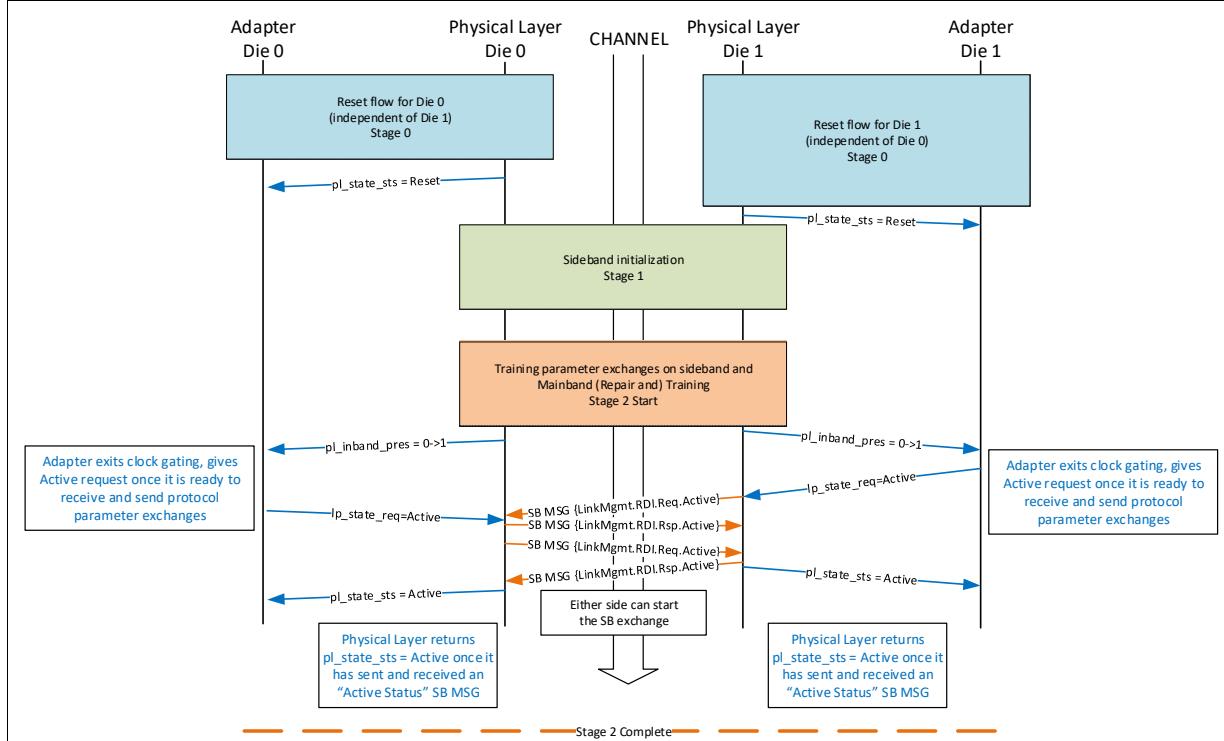
10.1.6 RDI bring up flow

Figure 10-7 shows an example flow for Stage 2 of the Link bring up highlighting the transitions on RDI. This stage requires sequencing on RDI that coordinates the state transition from Reset to Active.

1. Once Physical Layer has completed Link training, it must do the `pl_clk_req` handshake with the Adapter and reflect `pl_inband_pres=1` on RDI. Note that the `pl_clk_req` handshake is not shown in the example flow in Figure 10-7
2. This is the trigger for Adapter to request Active state. It must perform the `lp_wake_req` handshake as described in Section 10.1.3. Note that the `lp_wake_req` handshake is not shown in the example flow in Figure 10-7.
3. Only after sampling `lp_state_req = Active`, the Physical Layer must send the `{LinkMgmt.RDI.Req.Active}` sideband message to remote Link partner's Physical Layer.
4. The Physical Layer must respond to the `{LinkMgmt.RDI.Req.Active}` sideband message with a `{LinkMgmt.RDI.Rsp.Active}` sideband message. The `{LinkMgmt.RDI.Rsp.Active}` sideband message must only be sent after the Physical Layer has sampled `lp_state_req = Active` from its local RDI.
5. Once the Physical Layer has sent and received the `{LinkMgmt.RDI.Rsp.Active}` sideband message, it must transition `pl_state_sts` to Active.
6. This opens up the Adapter to transition to Stage 3 of the bring up flow.

Steps 3 to 5 are referred to as the “Active Entry handshake” and must be performed for every entry to Active state. Active.PMNAK to Active transition is not considered here because Active.PMNAK is only a sub-state of Active.

Figure 10-7. Example flow of Link bring up on RDI



10.1.7 RDI PM flow

This section defines the rules for PM entry, exit and abort flows as they apply to handshakes on the RDI. The rules for L1 and L2 are the same, except that exit from L2 is to Reset state, whereas exit from L1 is to Retrain state. This section uses PM to denote L1 or L2. A “PM Request” sideband message is {LinkMgmt.RDI.Req.L1} or {LinkMgmt.RDI.Req.L2}. A “PM Response” sideband message is {LinkMgmt.RDI.Rsp.L1} or {LinkMgmt.RDI.Rsp.L2}.

- Regardless of protocol, the PM entry or exit flow is symmetric on RDI. Both Physical Layer must issue PM entry request through a sideband message once the conditions of PM entry have been satisfied. PM entry is considered successful and complete once both sides have received a valid “PM Response” sideband message. [Figure 10-8](#) shows an example flow for L1. Once the RDI status is PM, the Physical Layer can transition itself to a power savings state (turning off the PLL for example). Note that the sideband logic and corresponding PLL needs to stay on even during L1 state.
- All the Adapter state machines (Adapter LSMs) in the Adapter must have moved to the corresponding PM state before the Adapter requests PM entry from remote Link partner. Adapter LSM in PM implies the retry buffer of the Adapter must be empty, and it must not have any new Flits (or Ack/Nak) pending to be scheduled. Essentially there should be no traffic on mainband when PM entry is requested by the Adapter to the Physical Layer. The Adapter is permitted to clock gate its sideband logic once RDI status is PM and there are no outstanding transactions or responses on sideband. Physical Layer must do `p1_clk_req` handshake (if `p1_clk_req` is not already asserted or status is not Active) before forwarding sideband requests from the Link to the Adapter.
- Adapter requests PM entry by transitioning `lp_state_req` to the corresponding PM encoding. Once requested, the Adapter cannot change this request until it observes PM, Active.PMNAK, Retrain, or LinkError state on `lp_state_sts`. While requesting PM state, if the Adapter receives Active request from the Protocol Layer, or a PM exit request for the Adapter LSM on sideband, it must sink the message but delay processing it until `lp_state_sts` has resolved. Once the RDI state is resolved, the Adapter must first bring it back to Active before processing the other requests.
 - If the resolution is PM (upon successful PM entry) and the Protocol Layer needs to exit PM (or there is a pending Protocol Layer Active request from remote Link partner), then the Adapter must initiate PM exit flow on RDI by requesting `lp_state_req` = Active. All PM entry-related handshakes must have finished prior to this (this is when the Physical Layer on both sides of the Link have received a valid “PM Response” sideband message).
 - If the resolution is Active.PMNAK, the Adapter must initiate a request of Active on RDI. Once the status moves to Active, the Adapter is permitted to re-request PM entry (if all conditions of PM entry are still met). [Figure 10-9](#) shows an example of PM abort flow. The PM request could have been from either side.
 - If the resolution is LinkError, then the Adapter must propagate this to Protocol Layers. This also resets any outstanding PM handshakes.
- Physical Layer initiates a “PM Request” sideband message once it samples the corresponding PM encoding on `lp_state_req` and has completed the StallReq/Ack handshake with its Adapter.
- Once a Physical Layer receives a “PM request” sideband message, it must respond to it within 2 us:
 - If its local Adapter is requesting the corresponding PM state, it must respond with the corresponding “PM Response” sideband message. If the current status is not PM, it must transition `lp_state_sts` to PM after responding to the sideband message.
 - If the current `lp_state_sts` = PM, it must respond with “PM Response” sideband message.

- If **p1_state_sts** = Active and **lp_state_req** = Active and it remains this way for 1us after receiving the “PM Request” sideband message, it must respond with {LinkMgmt.RDI.Rsp.PMNAK} sideband message.
- If a Physical Layer receives a “PM Response” sideband message in response to a “PM Request” sideband message, it must transition **p1_state_sts** on its local RDI to PM (if it is currently in Active state). If the current state is not Active, no action needs to be taken.
- If a Physical Layer receives a {LinkMgmt.RDI.Rsp.PMNAK} sideband message in response to a “PM Request” sideband message, it must transition **p1_state_sts** on its local RDI to Active.PMNAK state if it is currently in Active state. If it is not in Active state, no action needs to be taken. The Physical Layer is permitted to retry PM entry handshake (if all conditions of PM entry are satisfied) at least 2 us after receiving the {LinkMgmt.RDI.Rsp.PMNAK} sideband message OR if it received a corresponding “PM Request” sideband message from the remote Link partner.
- PM exit is initiated by the Adapter requesting Active on RDI. This triggers the Physical Layer to initiate PM exit by sending a {LinkMgmt.RDI.Req.Active} sideband message. Physical Layer must make sure it has finished any Link retraining steps before it responds with the {LinkMgmt.RDI.Rsp.Active} sideband message. [Figure 10-10](#) shows an example flow of PM exit on RDI.
 - PM exit handshake completion requires both Physical Layers to send as well as receive a {LinkMgmt.RDI.Rsp.Active} sideband message. Once this has completed, the Physical Layer is permitted to transition **p1_state_sts** to Active on RDI.
 - If **p1_state_sts** = PM and a {LinkMgmt.RDI.Req.Active} sideband message is received, the Physical Layer must initiate **p1_clk_req** handshake with the Adapter, and transition **p1_state_sts** to Retrain. This must trigger the Adapter to request Active on **lp_state_req** (if not already doing so), and this in turn triggers the Physical Layer to send {LinkMgmt.RDI.Req.Active} sideband message to the remote Link partner. [Figure 10-11](#) shows an example of the L1 exit flow on RDI and its interaction with the LTSM in the Physical Layer. It is permitted for the LTSM to begin the Link PM exit and retraining flow when a {LinkMgmt.RDI.Req.Active} sideband message is received or when the Adapter requests Active on RDI. The timeout counters for the Active Request sideband message handshake must begin only after LTSM is in the LINKINIT state. L2 exit follows a similar flow for cases in which graceful exit is required without domain reset; however, the L2 exit is via Reset state on RDI, and not Retrain. Exit conditions from Reset state apply for L2 exit (i.e., a NOP -> Active transition is required on **lp_state_req** for the Physical Layer to exit Reset state on RDI).

Note that the following figures are examples for L1, and do not show the **lp_wake_req**, **p1_clk_req** handshakes. Implementations must follow the rules outlined for these handshakes in previous sections.

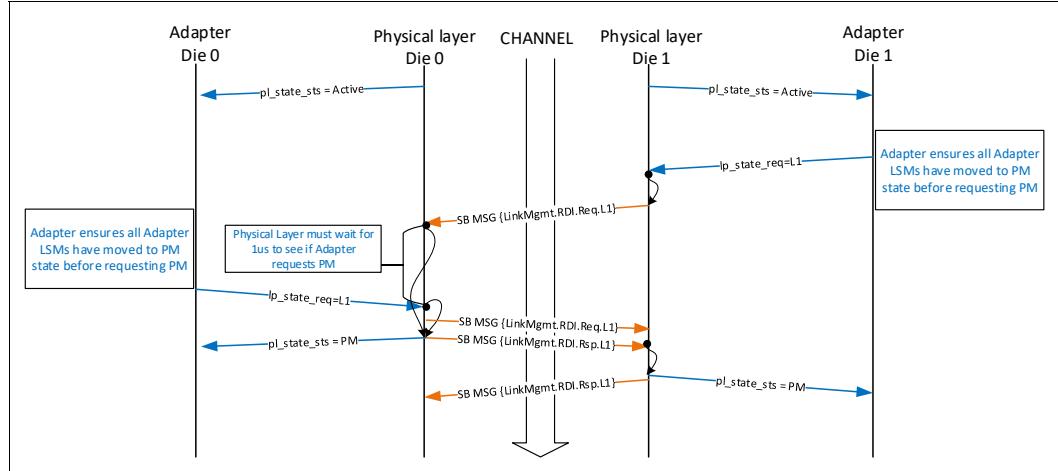
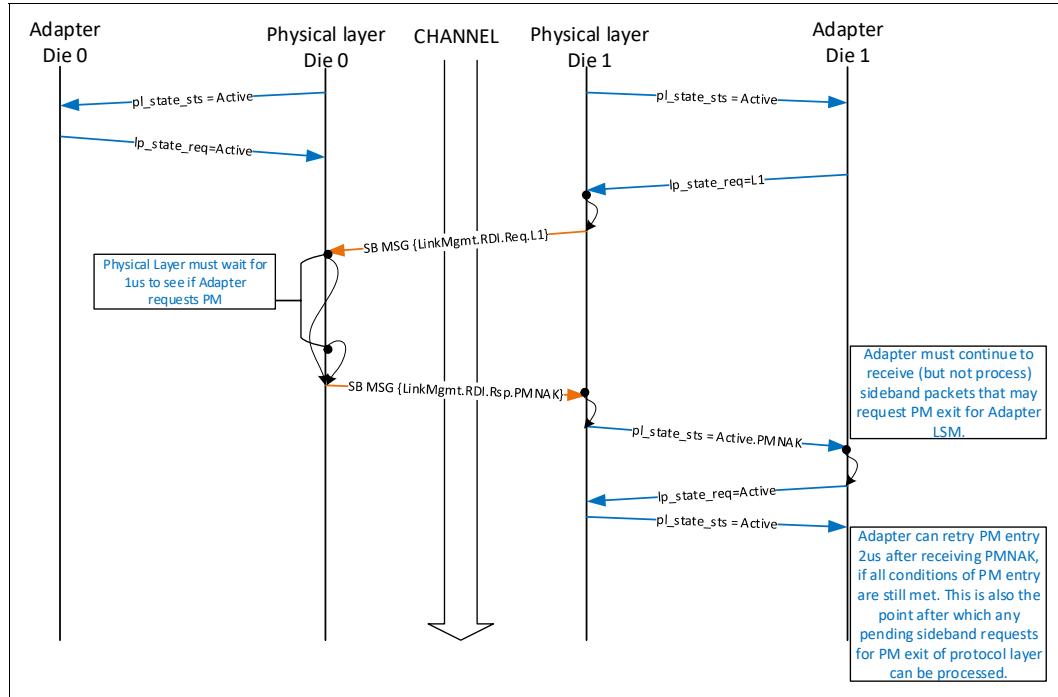
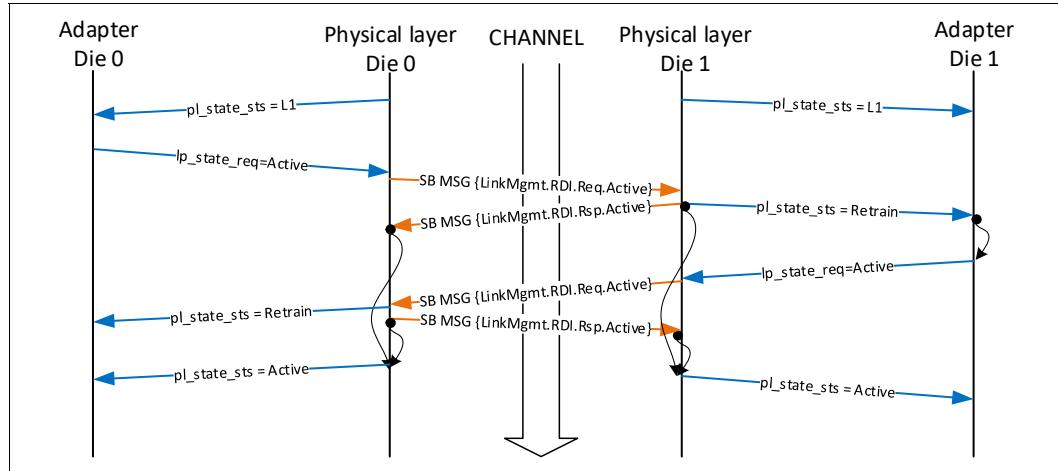
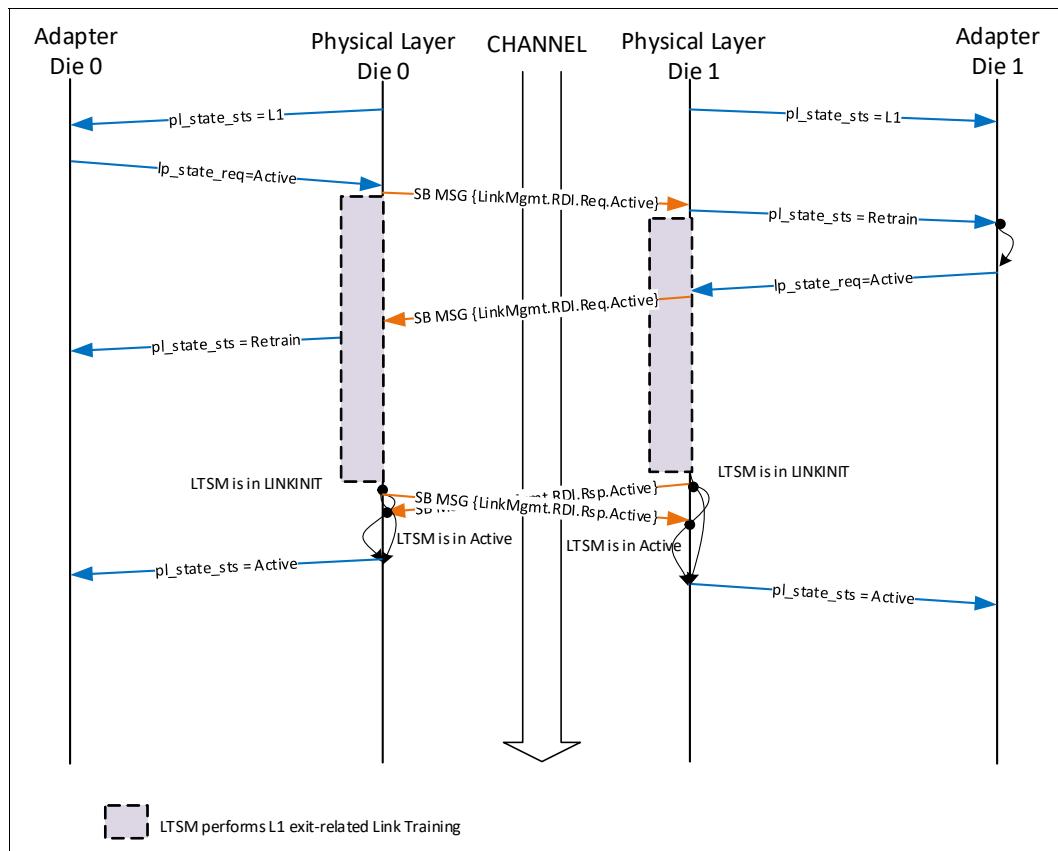
Figure 10-8. Successful PM entry flow**Figure 10-9. PM Abort flow**

Figure 10-10. PM Exit flow**Figure 10-11. RDI PM Exit Example Showing Interactions with LTSM**

10.2 Flit-Aware Die-to-Die Interface (FDI)

This section defines the signal descriptions and functionality associated with a single instance of Flit-Aware Die-to-Die Interface (FDI). A single instance is used for a Protocol Layer to Adapter connection. However, a single Adapter can host multiple protocol stacks using multiple instances of FDI.

[Figure 10-12](#) shows example configurations using multiple instances of FDI.

Figure 10-12. Example configurations using FDI

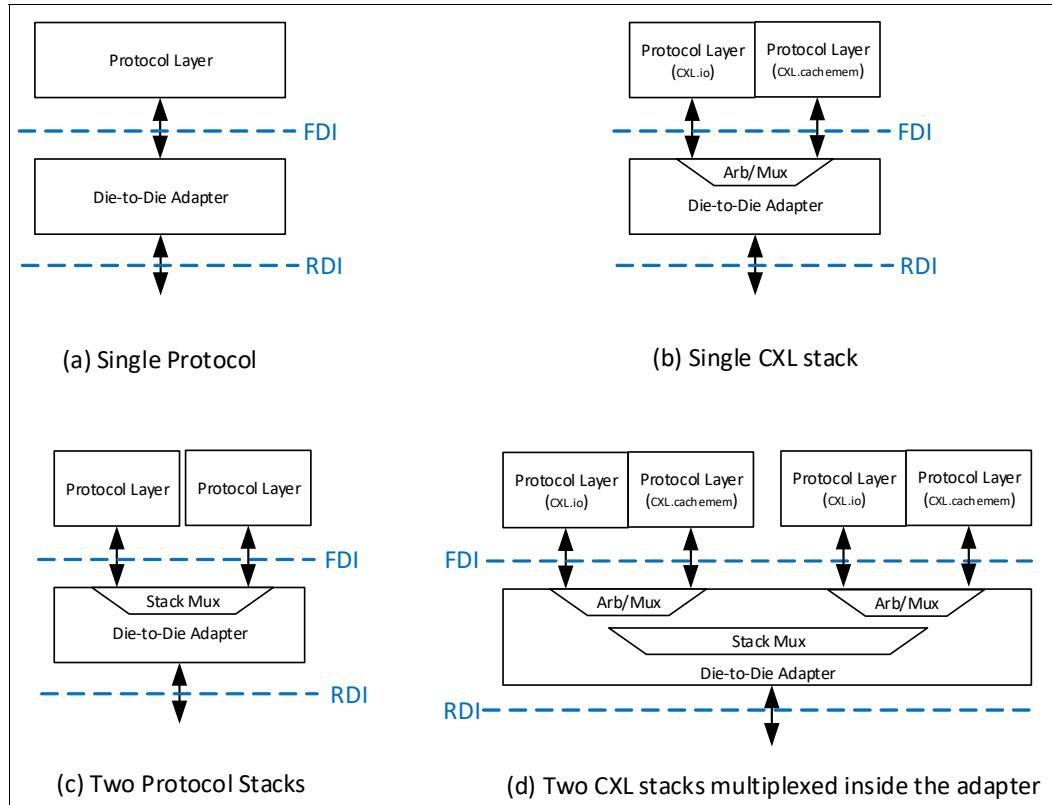


Table 10-3 lists the FDI signals and their descriptions. All signals are synchronous with `lclk`.

In Table 10-3:

- `p1_*` indicates that the signal is driven away from the Die-to-Die Adapter to the Protocol Layer.
- `lp_*` indicates that the signal is driven away from the Protocol Layer to the Die-to-Die Adapter.

Note: The same signal-naming convention as RDI is used to highlight that RDI signal list is a proper subset of FDI signal list.

Signal encodings pertaining to ‘Management Transport protocol’ are applicable only when Management Transport protocol was successfully negotiated on the mainband. Otherwise, those encodings are reserved. Also, `dm_*` signals in [Table 10-3](#) are applicable only when supporting Management Transport path over the mainband (“dm” is an abbreviation for “d2d_adapter-to-management_port_gateway”).

Table 10-3. FDI signal list (Sheet 1 of 8)

Signal Name	Signal Description
lclk	The clock at which FDI operates.
lp_irdy	Signal indicating that the Protocol Layer potentially has data to send. This must be asserted if lp_valid is asserted and the Protocol Layer wants the Adapter to sample the data. lp_irdy must not be presented by the Protocol Layer when pl_state_sts is Reset except when the status transitions from LinkError to Reset. On a LinkError to Reset transition, it is permitted for lp_irdy to be asserted for a few clocks but it must be deasserted eventually. Physical Layer must ignore lp_irdy when status is Reset.
lp_valid	Protocol Layer to Adapter indication that data is valid on the corresponding lp_data bytes.
lp_data[NBYTES-1:0][7:0]	Protocol Layer to Adapter data, where 'NBYTES' equals number of bytes determined by the data width for the FDI instance.
lp_retimer_crd	When asserted at a rising clock edge, it indicates a single credit return for the Retimer Receiver buffer. Each credit corresponds to 256B of mainband data (including Flit header and CRC, etc.). This signal must NOT assert if a Retimer is not present. On FDI, this is an optional signal. It is permitted to have the Receiver buffers in the Protocol Layer for Raw Format only. If this is not exposed to Protocol Layer, Adapter must track credit at 256B granularity even for Raw Format and return credits to Physical Layer on RDI. When this is exposed on FDI, the Adapter must have the initial credits knowledge through other implementation specific means in order to advertise this to the remote Link partner during parameter exchanges.
lp_corrupt_crc	This signal is only applicable for CXL.cachemem in UCIE Flit Mode (i.e., the Adapter doing Retry) for CXL 256B Flit Mode. It is meant as a latency optimization that enables detection and containment for viral or poison using the Adapter to corrupt CRC of outgoing Flit. It is recommended to corrupt CRC by performing a bitwise XOR of the computed CRC with the syndrome 138EH. The syndrome was computed such that no 1-bit or 2-bit errors alias to this syndrome, and it has the least probability of aliasing with 3-bit errors. For Standard 256B Flits, Protocol Layer asserts this along with lp_valid for the last chunk of the Flit that needs containment. Adapter corrupts CRC for both of the 128B halves of the Flit which had this set. It also must make sure to overwrite this flit (with the next flit sent by the Protocol Layer) in the Tx Retry buffer. For Latency-Optimized 256B Flits, Protocol Layer asserts this along with lp_valid for the last chunk of the 128B Flit half that needs containment. If lp_corrupt_crc is asserted on the first 128B half of the Flit, Protocol Layer must assert it on the second 128B half of the Flit as well. The very next Flit from the Protocol Layer after this signal has been asserted must carry the information relevant for viral, as defined in the CXL specification. If this was asserted on the second 128B half of the Flit only, it is the responsibility of the Protocol Layer to send the first 128B half exactly as before, and insert the viral information in the second half of the Flit. Adapter corrupts CRC for the 128B half of the Flit which had this set. It also must make sure to overwrite this flit (with the next flit sent by the Protocol Layer) in the Tx Retry buffer.
lp_dllp[NDLLP-1:0]	Protocol Layer to Adapter transfer of DLLP bytes. This is not used for 64B Flit Mode, CXL.cachemem or Streaming protocols. For a 64B data path on lp_data , it is recommended to assign NDLLP >= 8, so that 1 DLLP per Flit can be transferred from the Protocol Layer to the Adapter on average. The Adapter is responsible for inserting DLLP into DLP bytes 2:5 if the Flit packing rules permit it. See Section 10.2.4.1 for additional rules.
lp_dllp_valid	Indicates valid DLLP transfer on lp_dllp . DLLP transfers are not subject to backpressure by pl_trdy (the Adapter must have storage for different types of DLLP and this can be overwritten so that the latest DLLPs are sent to remote Link partner). DLLP transfers are subject to backpressure by pl_stallreq - Protocol Layer must stop DLLP transfers at DLLP Flit aligned boundary before giving lp_stallack or requesting PM.
lp_dllp_ofc	Indicates that the corresponding DLLP bytes on lp_dllp follow the Optimized_Update_FC format. It must stay asserted for the entire duration of the DLLP transfer on lp_dllp .

Table 10-3. FDI signal list (Sheet 2 of 8)

Signal Name	Signal Description
lp_stream[7:0]	Protocol Layer to Adapter signal that indicates the stream ID to use with data. Each stream ID maps to a unique protocol and stack. It is relevant only when lp_valid is 1. 00h: Reserved 01h: Stack 0: PCIe 02h: Stack 0: CXL.io 03h: Stack 0: CXL.cachemem 04h: Stack 0: Streaming protocol 05h: Stack 0: Management Transport protocol 11h: Stack 1: PCIe 12h: Stack 1: CXL.io 13h: Stack 1: CXL.cachemem 14h: Stack 1: Streaming protocol 15h: Stack 1: Management Transport protocol Other encodings are Reserved.
pl_trdy	The Adapter is ready to accept data. Data is accepted by the Adapter when pl_trdy , lp_valid , and lp_irdy are asserted at the rising edge of lclk . This signal must be asserted only if pl_state_sts is Active or when performing the pl_stallreq / lp_stallack handshake when the pl_state_sts is LinkError (see Section 10.3.3.7).
pl_valid	Adapter to Protocol Layer indication that data is valid on pl_data .
pl_data[NBYTES-1:0] [7:0]	Adapter to Protocol Layer data, where NBYTES equals the number of bytes determined by the data width for the FDI instance.
pl_retimer_crd	When asserted at a rising clock edge, it indicates a single credit return from the Retimer. Each credit corresponds to 256B of mainband data (including Flit header and CRC, etc.). This signal must NOT assert if a Retimer is not present. On FDI, this is an optional signal. It is permitted to expose these credits to Protocol Layer for Raw Format only. If this is not exposed to Protocol Layer, Adapter must track credit at 256B granularity even for Raw Format and back-pressure the Protocol Layer using pl_trdy . When this is exposed on FDI, the Adapter converts the initial credits received from the Retimer over sideband to credit returns to the Protocol Layer on this bit after Adapter LSM has moved to Active state.
pl_dllp[NDLLP-1:0]	Adapter to Protocol Layer transfer of DLLP bytes. This is not used for 68B Flit mode, CXL.cachemem or Streaming protocols. For a 64B data path on pl_data , it is recommended to assign NDLLP >= 8, so that 1 DLLP per Flit can be transferred from the Adapter to the Protocol Layer, on average. The Adapter is responsible for extracting DLLP from DLP Bytes 2:5 if a Flit Marker is not present. The Adapter is also responsible for indicating Optimized_Update_FC format by setting pl_dllp_ofc = 1 for the corresponding transfer on FDI.
pl_dllp_valid	Indicates valid DLLP transfer on pl_dllp . DLLPs can be transferred to the Protocol Layer whenever valid Flits can be transferred on pl_data . There is no backpressure and the Protocol Layer must always sink DLLPs.
pl_dllp_ofc	Indicates that the corresponding DLLP bytes on pl_dllp follow the Optimized_Update_FC format. It must stay asserted for the entire duration of the DLLP transfer on pl_dllp .

Table 10-3. FDI signal list (Sheet 3 of 8)

Signal Name	Signal Description
pl_stream[7:0]	Adapter to Protocol Layer signal that indicates the stream ID to use with data. Each stream ID maps to a unique protocol. It is relevant only when pl_valid is 1. 00h: Reserved 01h: Stack 0: PCIe 02h: Stack 0: CXL.io 03h: Stack 0: CXL.cachemem 04h: Stack 0: Streaming protocol 05h: Stack 0: Management Transport protocol 11h: Stack 1: PCIe 12h: Stack 1: CXL.io 13h: Stack 1: CXL.cachemem 14h: Stack 1: Streaming protocol 15h: Stack 1: Management Transport protocol Other encodings are Reserved.
pl_flit_cancel	Adapter to Protocol Layer indication to dump a Flit. This enables latency optimizations on the Receiver data path when CRC checking is enabled in the Adapter. It is not applicable for Raw Format or 68B Flit Format. For Standard 256B Flit, it is required to have a fixed number of clock cycle delay between the last chunk of a Flit transfer and the assertion of pl_flit_cancel . This delay is fixed to be 1 cycle (i.e., the cycle after the last chunk transfer of a Flit). When this signal is asserted, Protocol Layer must not consume the associated Flit. For Latency-Optimized 256B Flits, it is required to have a fixed number of clock cycle delay between the last chunk of a 128B half Flit transfer and the assertion of pl_flit_cancel . This delay is fixed to be 1 cycle (i.e., the cycle after the last transfer of the corresponding 128B chunk). When this signal is asserted, Protocol Layer must not consume the associated Flit half. When this mode is supported, Protocol Layer must support it for all applicable Flit Formats associated with the corresponding protocol. Adapter must guarantee this to be a single cycle pulse when dumping a Flit or Flit half. It is the responsibility of the Adapter to ensure that the canceled Flits or Flit halves are eventually replayed on the interface without cancellation in the correct order once they pass CRC after Retry etc. See Section 10.2.5 for examples. When operating in UCIe Flit mode, it is permitted to use this signal to also cancel valid NOP Flits for the Protocol Layer to prevent forwarding these to the Protocol Layer. However for interoperability, if a Protocol Layer receives a NOP Flit without a corresponding pl_flit_cancel , it must discard these Flits.
lp_state_req[3:0]	Protocol Layer request to Adapter to request state change. Encodings as follows: 0000b: NOP 0001b: Active 0100b: L1 1000b: L2 1001b: LinkReset 1011b: Retrain 1100b: Disabled All other encodings are reserved.
lp_linkerror	Protocol Layer to Adapter indication that an error has occurred which requires the Link to go down. Adapter must propagate this request to RDI, and move the Adapter LSMs (and CXL vLSMs if applicable) to LinkError state once RDI is in LinkError state. It must stay there as long as lp_linkerror=1 . The reason for having this be an indication decoupled from regular state transitions is to allow immediate action on part of the Protocol Layer and Adapter in order to provide the quickest path for error containment when applicable (for example, a viral error escalation could map to the LinkError state)

Table 10-3. FDI signal list (Sheet 4 of 8)

Signal Name	Signal Description
pl_state_sts[3:0]	<p>Adapter to Protocol Layer Status indication of the Interface. Encodings as follows:</p> <ul style="list-style-type: none"> 0000b: Reset 0001b: Active 0011b: Active.PMNAK 0100b: L1 1000b: L2 1001b: LinkReset 1010b: LinkError 1011b: Retrain 1100b: Disabled <p>All other encodings are reserved.</p> <p>The status signal is permitted to transition from Adapter autonomously when applicable. For example the Adapter asserts the Retrain status when it decides to enter retraining either autonomously or when requested by remote agent.</p> <p>For PCIe/Streaming protocols, the Adapter LSM is exposed as pl_state_sts to the Protocol Layer. For CXL protocol, the ARB/MUX vLSM is exposed as pl_state_sts to the Protocol Layer.</p> <p>The Link Status is considered to be Up from Protocol Layer perspective when FDI status is Active, Active.PMNAK, Retrain, L1, or L2. The Link Status is considered Down for other states of FDI.</p>
pl_inband_pres	<p>Adapter to the Protocol Layer indication that the Die-to-Die Link has finished negotiation of parameters with remote Link partner and is ready for transitioning the FDI Link State Machine (LSM) to Active.</p> <p>Once it transitions to 1b, this must stay 1b until FDI moves to Active or LinkError. It stays asserted while FDI is in Retrain, Active, Active.PMNAK, L1, or L2. It must de-assert during LinkReset, Disabled or LinkError states.</p>
pl_error	<p>Adapter to the Protocol Layer indication that it has detected a framing related error. It is pipeline matched with the receive data path. It must also assert if pl_error was asserted on RDI by the Physical Layer for a Flit which the Adapter is forwarding to the Protocol Layer.</p> <p>In UCIe Flit Mode, it is permitted for Protocol Layer to use pl_error indication to log correctable errors when Retry is enabled from the Adapter. The Adapter must finish any partial Flits sent to the Protocol Layer and assert pl_filit_cancel in order to prevent consumption of that Flit by the Protocol Layer. Adapter must initiate Link Retrain on RDI following this, if it was a framing error detected by the Adapter.</p> <p>In UCIe Flit Mode, if Retry is disabled, the Adapter is responsible for mapping internally detected framing errors or Physical Layer received pl_error to an Uncorrectable Internal Error and escalate it as pl_trainerror if the mask and severity registers permit the escalation.</p> <p>If the Link is operating in Raw Format, the Adapter has no internal detection of framing errors, it just forwards any pl_error indication received from the Physical Layer on FDI such that it is pipeline matched to the data path.</p> <p>It is a pulse indication that can occur only when FDI receiver is Active (i.e. pl_rx_active_req = 1 & pl_rx_active_sts = 1).</p>
pl_cerror	<p>Adapter to the Protocol Layer indication that a correctable error was detected that does not affect the data path. The Protocol Layer must OR the pl_error and pl_cerror signals for Correctable Error Logging.</p> <p>Errors logged in the Correctable Error Status register are mapped to this signal if the corresponding mask bit in the Correctable Error Mask register is cleared to 0.</p> <p>It is a pulse of one or more cycles that can occur in any FDI state. If it is a state in which clock gating is permitted, it is the responsibility of the Adapter to perform the clock gating exit handshake with the Protocol Layer before asserting this signal. Clock gating can resume after pl_cerror is de-asserted and all other conditions permitting clock gating have been met.</p>

Table 10-3. FDI signal list (Sheet 5 of 8)

Signal Name	Signal Description
pl_nferror	Adapter to the Protocol Layer indication that a non-fatal error was detected. This is used by Protocol Layer for error logging and corresponding escalation to software. The Adapter must OR any internally detected errors with pl_nferror on RDI and forward the result on FDI. Errors logged in Uncorrectable Error Status Register are mapped to this signal if the corresponding Severity and Mask bits are cleared to 0.
pl_trainerror	It is a pulse of one or more cycles that can occur in any FDI state. If it is a state in which clock gating is permitted, it is the responsibility of the Adapter to perform the clock gating exit handshake with the Protocol Layer before asserting this signal. Clock gating can resume after pl_nferror is de-asserted and all other conditions permitting clock gating have been met.
pl_rx_active_req	Indicates a fatal error from the Adapter. Adapter must transition pl_state_sts to LinkError if not already in LinkError state. (Note that the Adapter first takes RDI to LinkError, and that LinkError is eventually propagated to all the FDI states). Implementations are permitted to map any fatal error to this signal that require upper layer escalation (or interrupt generation) depending on system level requirements. Errors logged in Uncorrectable Error Status Register are mapped to this signal if the corresponding Severity is set to 1 and the corresponding Mask bit is cleared to 0.
lp_rx_active_sts	It is a level signal that can assert in any FDI state but stays asserted until FDI exits the LinkError state to Reset state.
pl_protocol[3:0]	Adapter asserts this signal to request the Protocol Layer to open its Receiver's data path and get ready for receiving protocol data or Flits. The rising edge of this signal must be when pl_state_sts is Reset, Retrain or Active. Together with lp_rx_active_sts , it forms a four way handshake. See Section 10.2.7 for rules related to this handshake.
pl_protocol_flitfmt[3:0]	Protocol Layer responds to pl_rx_active_req after it is ready to receive and parse protocol data or Flits. Together with pl_rx_active_req , it forms a four way handshake. See Section 10.2.7 for rules related to this handshake.
	Adapter indication to Protocol Layer of the protocol that was negotiated during training. 0000b: PCIe without Management Transport 0011b: CXL.1 [Single protocol, i.e., CXL.io] without Management Transport 0100b: CXL.2 [Multi-protocol, Type 1 device] without Management Transport 0101b: CXL.3 [Multi-protocol, Type 2 device] without Management Transport 0110b: CXL.4 [Multi-protocol, Type 3 device] without Management Transport 0111b: Streaming protocol without Management Transport 1000b: PCIe with Management Transport 1001b: Management Transport 1011b: CXL.1 [Single protocol, i.e., CXL.io] with Management Transport 1100b: CXL.2 [Multi-protocol, Type 1 device] with Management Transport 1101b: CXL.3 [Multi-protocol, Type 2 device] with Management Transport 1110b: CXL.4 [Multi-protocol, Type 3 device] with Management Transport 1111b: Streaming protocol with Management Transport Other encodings are Reserved
	This indicates the negotiated Format. See Chapter 3.0 for the definitions of these formats. 0001b: <i>Format 1</i> : Raw Format 0010b: <i>Format 2</i> : 68B Flit Format 0011b: <i>Format 3</i> : Standard 256B End Header Flit Format 0100b: <i>Format 4</i> : Standard 256B Start Header Flit Format 0101b: <i>Format 5</i> : Latency-Optimized 256B without Optional Bytes Flit Format 0110b: <i>Format 6</i> : Latency-Optimized 256B with Optional Bytes Flit Format Other encodings are Reserved

Table 10-3. FDI signal list (Sheet 6 of 8)

Signal Name	Signal Description
<code>pl_protocol_vld</code>	Indication that <code>pl_protocol</code> , and <code>pl_protocol_fmt</code> have valid information. This is a level signal, asserted when the Adapter has determined the appropriate protocol, but must only de-assert again after subsequent transitions to LinkError or Reset state depending on the Link state machine transitions. Protocol Layer must sample and store <code>pl_protocol</code> and <code>pl_protocol_fmt</code> when <code>pl_protocol_vld</code> = 1 and <code>pl_state_sts</code> = Reset and <code>pl_inband_pres</code> = 1. It must treat this saved value as the negotiated protocol until <code>pl_state_sts</code> = Reset and <code>pl_inband_pres</code> = 0. The Adapter must ensure that if <code>pl_inband_pres</code> = 1, <code>pl_protocol_vld</code> = 1 and <code>pl_state_sts</code> = Reset, then <code>pl_protocol</code> and <code>pl_protocol_fmt</code> are the correct values that can be sampled by the Protocol Layer.
<code>pl_stallreq</code>	Adapter request to Protocol Layer to flush all Flits for state transition and not prepare any new Flits. See Section 10.2.6 for details.
<code>lp_stallack</code>	Protocol Layer to Adapter indication that the Flits are aligned and stalled (if <code>pl_stallreq</code> was asserted). It is strongly recommended that this response logic be on a global free running clock, so the Protocol Layer can respond to <code>pl_stallreq</code> with <code>lp_stallack</code> even if other significant portions of the Protocol Layer are clock gated.
<code>pl_physinrecenter</code>	Adapter indication to Protocol Layer that the Link is doing training or retraining (i.e., RDI has <code>pl_physinrecenter</code> asserted or the Adapter LSM has not moved to Active yet). If this is asserted during a state where clock gating is permitted, the <code>pl_clk_req</code> / <code>lp_clk_ack</code> handshake must be performed with the upper layer. The upper layers are permitted to use this to update the "Link Training/Retraining" bit in the UCIe Link Status register.
<code>pl_physinl1</code>	Adapter indication to Protocol Layer that the Physical Layer is in L1 power management state (i.e., RDI is in L1 state).
<code>pl_physinl2</code>	Adapter indication to Protocol Layer that the Physical Layer is in L2 power management state (i.e., RDI is in L2 state).
<code>pl_speedmode[2:0]</code>	Current Link speed. The following encodings are used: 000b: 4GT/s 001b: 8GT/s 010b: 12GT/s 011b: 16GT/s 100b: 24GT/s 101b: 32GT/s other encodings are reserved. The Protocol Layer must only consider this signal to be relevant when the FDI state is Active or Retrain. For multi-module configurations, all modules must operate at the same speed.
<code>pl_lnk_cfg[2:0]</code>	Current Link Configuration. Indicates the current operating width of a module. 000b: x4 001b: x8 010b: x16 011b: x32 100b: x64 101b: x128 110b: x256 other encodings are reserved. The Protocol Layer must only consider this signal to be relevant when the FDI state is Active or Retrain. This is the total width across all Active modules for the corresponding FDI instance.

Table 10-3. FDI signal list (Sheet 7 of 8)

Signal Name	Signal Description
pl_clk_req	<p>Request from the Adapter to remove clock gating from the internal logic of the Protocol Layer. This is an asynchronous signal from the Protocol Layer's perspective since it is not tied to lclk being available in the Protocol Layer. Together with lp_clk_ack, it forms a four-way handshake to enable dynamic clock gating in the Protocol Layer.</p> <p>When dynamic clock gating is supported, the Protocol Layer must use this signal to exit clock gating before responding with lp_clk_ack.</p> <p>If dynamic clock gating is not supported, it is permitted for the Adapter to tie this signal to 1b.</p>
lp_clk_ack	<p>Response from the Protocol Layer to the Adapter acknowledging that its clocks have been un gated in response to pl_clk_req. This signal is only asserted when pl_clk_req is asserted, and de-asserted after pl_clk_req has de-asserted.</p> <p>When dynamic clock gating is not supported by the Protocol Layer, it must stage pl_clk_req internally for one or more clock cycles and turn it around as lp_clk_ack. This way it will still participate in the handshake even though it does not support dynamic clock gating.</p>
lp_wake_req	<p>Request from the Protocol Layer to remove clock gating from the internal logic of the Adapter. This is an asynchronous signal relative to lclk from the Adapter's perspective since it is not tied to lclk being available in the Adapter. Together with pl_wake_ack, it forms a four-way handshake to enable dynamic clock gating in the Adapter.</p> <p>When dynamic clock gating is supported, the Adapter must use this signal to exit clock gating before responding with pl_wake_ack.</p> <p>If dynamic clock gating is not supported, it is permitted for the Protocol Layer to tie this signal to 1b.</p>
pl_wake_ack	<p>Response from the Adapter to the Protocol Layer acknowledging that its clocks have been un gated in response to lp_wake_req. This signal is only asserted after lp_wake_req has asserted, and is de-asserted after lp_wake_req has de-asserted.</p> <p>When dynamic clock gating is not supported by the Adapter, it must stage lp_wake_req internally for one or more clock cycles and turn it around as pl_wake_ack. This way it will still participate in the handshake even though it does not support dynamic clock gating.</p>
pl_cfg[NC-1:0]	<p>This is the sideband interface from the Adapter to the Protocol Layer. See Chapter 7.0 for details. NC is the width of the interface. Supported values are 8, 16, and 32.</p> <p>Register accesses must be implemented by hardware to be atomic regardless of the width of the interface (i.e., all 32 bits of a register must be updated in the same cycle for a 32-bit register write, and similarly all 64 bits of a register must be updated in the same cycle for a 64-bit register write).</p>
pl_cfg_vld	When asserted, indicates that pl_cfg has valid information that should be consumed by the Protocol Layer.
pl_cfg_crd	<p>Credit return for sideband packets from the Adapter to the Protocol Layer for sideband packets. Each credit corresponds to 64 bits of header and 64 bits of data. Even transactions that do not carry data or carry 32 bits of data consume the same credit and the Receiver returns the credit once the corresponding transaction has been processed or de-allocated from its internal buffers. See Section 7.1.3.1 for additional flow control rules. A value of 1 sampled at a rising clock edge indicates a single credit return.</p> <p>Because the advertised credits are design parameters, the Protocol Layer transmitter updates the credit counters with initial credits on domain reset exit, and no initialization credits are returned over the interface.</p> <p>Credit returns must follow the same rules of clock gating exit handshakes as the sideband packets to ensure that no credit returns are dropped by the receiver of the credit returns.</p>
lp_cfg[NC-1:0]	<p>This is the sideband interface from Protocol Layer to the Adapter. See Chapter 7.0 for details. NC is the width of the interface. Supported values are 8, 16, and 32.</p> <p>Register accesses must be implemented by hardware to be atomic regardless of the width of the interface (i.e., all 32 bits of a register must be updated in the same cycle for a 32-bit register write, and similarly all 64 bits of a register must be updated in the same cycle for a 64-bit register write).</p>
lp_cfg_vld	When asserted, indicates that lp_cfg has valid information that should be consumed by the Adapter.

Table 10-3. FDI signal list (Sheet 8 of 8)

Signal Name	Signal Description
<code>lp_cfg_crd</code>	Credit return for sideband packets from the Protocol Layer to the Adapter for sideband packets. Each credit corresponds to 64 bits of header and 64 bits of data. Even transactions that do not carry data or carry 32 bits of data consume the same credit and the Receiver returns the credit once the corresponding transaction has been processed or de-allocated from its internal buffers. See Section 7.1.3.1 for additional flow control rules. A value of 1 sampled at a rising clock edge indicates a single credit return. Because the advertised credits are design parameters, the Adapter transmitter updates the credit counters with initial credits on domain reset exit, and no initialization credits are returned over the interface. Credit returns must follow the same rules of clock gating exit handshakes as the sideband packets to ensure that no credit returns are dropped by the receiver of the credit returns.
<code>dm_param_exchange_done</code>	Signal resets to 0 on a Domain Reset. In single stack management transport implementations, this signal is asserted when adapter parameter exchange has completed between both sides and flit format/protocol have been finalized. It is reset whenever the link status=down. In multi-stack management transport implementations, this signal is asserted only when both stacks have completed their individual adapter parameter exchanges and protocol has been finalized (successfully or unsuccessfully) across both stacks. If at run time one of the active stacks enters link status=down condition, this signal de-asserts and asserts again only when the above condition is again met.
<code>dm_param_stack_count[N-1:0]</code>	Number of stacks that successfully negotiated Management Transport protocol. This field is sampled only when <code>dm_param_exchange_done</code> signal is asserted. If 68B Flit format was finalized, this field must be cleared to 00b. 00b: 0 stack 01b: 1 stack 10b: 2 stacks Others: reserved N=1 for single stack and 2 for 2 stacks.

10.2.1 Interface reset requirements

FDI does not define a separate interface signal for reset; however, it is required that the logic entities on both sides of FDI are in the same reset domain and the reset for each side is derived from the same source. Because reset may be staggered due to SoC routing, all signals coming out of reset must be driven to 0, unless otherwise specified. `lp_stream` and `p1_stream` are exceptions to this rule if they are tied off to their expected values at the time of integration. If `lp_stream` and `p1_stream` are not tied off, they must be driven to 0 when coming out of reset.

10.2.2 Interface clocking requirements

FDI requires both sides of the interface to be on the same clock domain. Moreover, the clock domain for sideband interface (`*cfg*`) is the same as the mainband signals.

Each side is permitted to instantiate clock crossing FIFOs internally if needed, as long as it does not violate the requirements at the interface itself.

It is important to note that there is no back pressure possible from the Protocol Layer to the Adapter on the main data path. So any clock crossing related logic internal to the Protocol Layer must take this into consideration.

10.2.3 Dynamic clock gating

Dynamic coarse clock gating is permitted in the Adapter and Protocol Layer when `p1_state_sts` is Reset, LinkReset, Disabled or PM states. This section defines the rules around entry and exit of clock gating. Note that clock gating is not permitted in LinkError states - it is expected that the UCIe usages

will enable error handlers to make sure the Link is not stuck in a LinkError state, if the intent is to save power when a Link is in an error state.

10.2.3.1 Rules and description for lp_wake_req/pl_wake_ack handshake

Protocol Layer can request removal of clock gating of the Adapter by asserting `lp_wake_req` (asynchronous to `lclk` availability in the Adapter). All Adapter implementations must respond with a `pl_wake_ack` (synchronous to `lclk`). The extent of internal clock ungating when `pl_wake_ack` is asserted is implementation-specific, but lclk must be available by this time to enable FDI transitions from the Protocol Layers. The Wake Req/Ack is a full handshake and it must be used for state transition requests (on `lp_state_req` or `lp_linkerror`) when moving away from a state in which clock gating is permitted. It must also be used for sending packets on the sideband interface.

Rules for this handshake:

1. Protocol Layer asserts `lp_wake_req` to request ungating of clocks by the Adapter.
2. The Adapter asserts `pl_wake_ack` to indicate that clock gating has been removed. There must be at least one clock cycle bubble between `lp_wake_req` assertion and `pl_wake_ack` assertion.
3. `lp_wake_req` must de-assert before `pl_wake_ack` de-asserts. It is the responsibility of the Protocol Layer to control the specific scenario of de-assertion. As an example, when performing the handshake for a state request, it is permitted to keep `lp_wake_req` asserted until it observes the desired state status. Protocol Layer is also permitted to keep `lp_wake_req` asserted through states where clock gating is not permitted in the Adapter (i.e., Active, LinkError or Retrain).
4. `lp_wake_req` should not be the only consideration for Adapter to perform clock gating, it must take into account `pl_state_sts` and other internal or Link requirements before performing global and/or local clock gating.
5. When performing `lp_wake_req/pl_wake_ack` handshake for `lp_state_req` transitions or `lp_linkerror` transition, the Protocol Layer is permitted to not wait for `pl_wake_ack` before changing `lp_state_req` or `lp_linkerror`.
6. When performing `lp_wake_req/pl_wake_ack` handshake for `lp_cfg` transitions, Protocol Layer must wait for `pl_wake_ack` before changing `lp_cfg` or `lp_cfg_vld`. Because `lp_cfg` can have multiple transitions for a single packet transfer, it is necessary to make sure that the Adapter clocks are up before transfer begins.

10.2.3.2 Rules and description for pl_clk_req/lp_clk_ack handshake

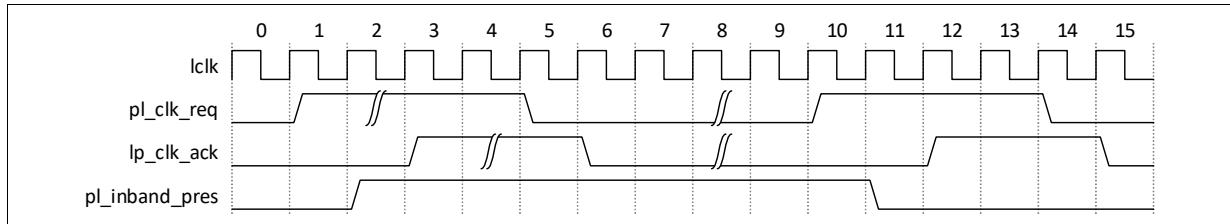
Adapter is allowed to initiate `pl_clk_req/lp_clk_ack` handshake at any time and the Protocol Layer must respond.

Rules for this handshake:

1. Adapter asserts `pl_clk_req` to request removal of clock gating by the Protocol Layer. This can be done anytime, and independent of current FDI state.
2. The Protocol Layer asserts `lp_clk_ack` to indicate that clock gating has been removed. There must be at least one clock cycle bubble between `pl_clk_req` assertion and `lp_clk_ack` assertion.
3. `pl_clk_req` must de-assert before `lp_clk_ack`. It is the responsibility of the Adapter to control the specific scenario of de-assertion, after the required actions for this handshake are completed.

4. `pl_clk_req` should not be the only consideration for the Protocol Layer to perform clock gating, it must take into account `pl_state_sts` and other protocol-specific requirements before performing trunk and/or local clock gating.
5. The Adapter must use this handshake to ensure transitions of `pl_inband_pres`, `pl_physin11`, `pl_physin12`, `pl_physinrecenter`, and `pl_rx_active_req` have been observed by the Protocol Layer. Since these are level oriented signals, the Adapter is permitted to let the signal transition without waiting for `lp_clk_ack`. When this is done during initial Link bring up, it is strongly recommended for the Adapter to keep `pl_clk_req` asserted until the state status transitions away from Reset to a state where clock gating is not permitted or until the state status is Reset and `pl_inband_pres` de-asserts.

Figure 10-13. Example Waveform Showing Handling of Level Transition



6. The Adapter must also perform this handshake before transition to LinkError state from Reset, LinkReset, Disabled or PM state (especially when the LinkError transition occurs by the Adapter without being directed by the Protocol Layer). It is permitted to assert `pl_clk_req` before the state change, in which case it must stay asserted until the state status transitions. It is also permitted to assert `pl_clk_req` after the state status transition, but in this case Adapter must wait for `lp_clk_ack` before performing another state transition.
7. The Adapter must also perform this handshake when the status is PM and remote Link partner is requesting PM exit. For exit from Reset, LinkReset, Disabled or PM states to a state that is not LinkError, it is required to assert `pl_clk_req` before the status change, and in this case it must stay asserted until the state status transitions away from Reset or PM.
8. The Adapter must also perform this handshake for sideband transfers from the Adapter to the Protocol Layer. When performing the handshake for `pl_cfg` transitions, Adapter must wait for `lp_clk_ack` before changing `pl_cfg` or `pl_cfg_vld`. Because `pl_cfg` can have multiple transitions for a single packet transfer, it is necessary to make sure that the Protocol Layer clocks are up before transfer begins.

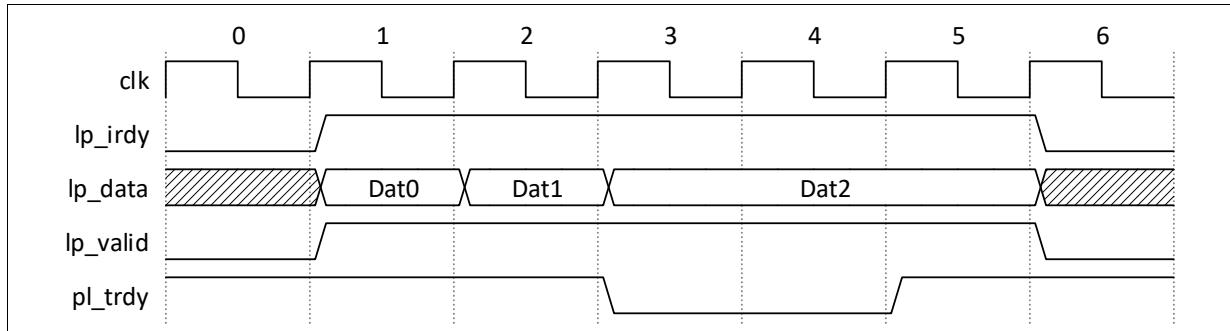
When clock-gated in Reset states, Protocol Layers that rely on dynamic clock gating to save power must wait in clock gated state for `pl_inband_pres`=1. The Adapter will request clock gating exit when it transitions `pl_inband_pres`, and the Protocol Layer must wait for `pl_inband_pres` assertion before requesting `lp_state_req` = ACTIVE. If `pl_inband_pres` de-asserts while `pl_state_sts` = Reset, then the Protocol Layer is permitted to return to clock-gated state after moving `lp_state_req` to NOP.

10.2.4 Data Transfer

As indicated in the signal list descriptions, when Protocol Layer is sending data to the Adapter, data is transferred when `lp_irdy`, `pl_trdy` and `lp_valid` are asserted. Figure 10-14 shows an example waveform for data transfer from the Protocol Layer to the Adapter. Data is transmitted on clock cycles 1, 2, and 5. No assumption should be made by Protocol Layer about when `pl_trdy` can de-assert or for how many cycles it remains de-asserted before it is asserted again, unless explicitly guaranteed by the Adapter. If a Flit transfer takes multiple clock cycles, the Protocol Layer is not permitted to insert

bubbles in the middle of a Flit transfer (i.e., `lp_valid` and `lp_irdy` must be asserted continuously until the Flit transfer is complete. Of course, data transfer can stall because of `p1_trdy` de-assertion).

Figure 10-14. Data Transfer from Protocol Layer to Adapter



As indicated in the signal list descriptions, when Adapter is sending data to the Protocol layer, there is no back-pressure mechanism, and data is transferred whenever `p1_valid` is asserted. The Adapter is permitted to insert bubbles in the middle of a Flit transfer and the Protocol Layer must be able to handle that.

10.2.4.1 DLLP transfer rules for 256B Flit Mode

For PCIe and CXL.io 256B Flits (both Standard and Latency-Optimized), FDI provides a separate signal for DLLP transfers from the Protocol Layer to the Adapter and vice-versa. Since the DLLPs have to bypass the Retry buffer, the separate signal enables the Adapter to insert DLLPs into the Flits from the Protocol Layer or the Retry buffer, if it is permitted to do so per the Flit packing rules of the corresponding Flit Format. Rules relevant for FDI operation (per FDI instance) are outlined below:

For the Transmitting side:

- Protocol Layer is responsible for sending the relevant DLLPs at the rate defined by the underlying Protocol to prevent timeouts of DLLP exchanges. If the Protocol Layer has no TLPs to send, it must insert NOP Flits to ensure that the Adapter gets an opportunity to insert the DLLP bytes.
- When transferring DLLP or Optimized_Update_FC, the least significant byte is sent over Byte 0 of the FDI bus, the next byte over Byte 1 and so on. When the transfer is over multiple chunks across FDI, Byte 0 is transferred on the first chunk LSB, Byte 1 following it and so on.
- The Adapter must have storage for at least 1 DLLP of every unique DLLP encoding (including Optimized_Update_FC) per supported VC that is possible for transfer to remote Link partner. The Adapter tracks pending DLLPs and schedules them on the next available opportunity for the relevant Flits. Credit update DLLPs must not be reordered for a VC by the Adapter. It is however permitted to discard a pending credit DLLP if the Protocol Layer presented a new credit DLLP of the same FC and VC. This extends to Optimized_Update_FC packets; i.e., it is permitted to discard any pending NP or P Update FC DLLPs, if the Protocol Layer transferred an Optimized_Update_FC for the corresponding VC.

On the Receiving side:

- The Adapter must extract DLLPs from received Flits of the corresponding protocol and forward them to the Protocol Layer. The FDI signal width of `p1_dllp` must be wide enough to keep up with the maximum rate of DLLPs that could be received from the Link.
- When transferring DLLP over multiple chunks across FDI, Byte 0 is transferred on the first chunk LSB, Byte 1 following it and so on.

- The Protocol Identifier corresponding to D2D Adapter in the Flit Header overlaps with the Flit usage of NOP Flits defined in PCIe and CXL specifications. The Adapter must check for available DLLPs in these Flits as well. All 0 bits in the DLLP byte positions indicate a NOP DLLP, and must not be forwarded to the Protocol Layer.

10.2.5 Examples of `pl_flit_cancel` Timing Relationships

In all the examples shown in this section, a 64B datapath on FDI is shown, and “F0Bytes” in the figures correspond to “Flit 0 Bytes”.

Figure 10-15 shows an example timing relationship for `pl_flit_cancel` and `pl_data` for Latency-Optimized Flits when the first Flit half fails CRC check. Both Flit halves are canceled by the Adapter in this example by asserting `pl_flit_cancel` one clock after the last chunk transfer of the corresponding Flit half. It is permitted for the Adapter to de-assert `pl_valid` on clock cycles 5 and 6 instead of canceling that Flit half; however, this might have implications to meeting physical design timing margins in the Adapter. The use of `pl_flit_cancel` allows the Adapter to perform the CRC check on the side without putting the CRC logic in the critical timing path of the data flow and thus permitting higher frequency operation for implementations. In the example shown, after replay flow the entire Flit is transferred to the Protocol Layer without canceling as CRC checks pass.

Figure 10-15. Example for `pl_flit_cancel` for Latency-Optimized Flits and CRC Error on First Flit Half

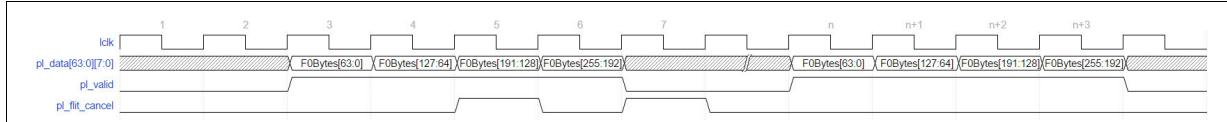


Figure 10-16 and **Figure 10-17** show examples of two possible implementations of timing relationship for `pl_flit_cancel` and `pl_data` for Latency-Optimized Flits when the second Flit half fails CRC check. In both cases, the first half of the Flit is consumed by the Protocol Layer because it is not canceled by the Adapter (the data transferred on clock cycles 3 and 4).

In the first case (shown in **Figure 10-16**), after the replay flow, CRC passes, and the Adapter ensures that the Protocol Layer does not re-consume the first half again by asserting `pl_flit_cancel` for it. In this case, `pl_valid` asserts for the entire Flit, but only the second half is consumed because the first half was canceled on clock cycle (n+2).

In the second case (shown in **Figure 10-17**), after the replay flow, CRC passes, and the Adapter ensures that the Protocol Layer does not re-consume the first half again by not asserting `pl_valid` for it.

Figure 10-16. Example for `pl_flit_cancel` for Latency-Optimized Flits and CRC Error on Second Flit Half

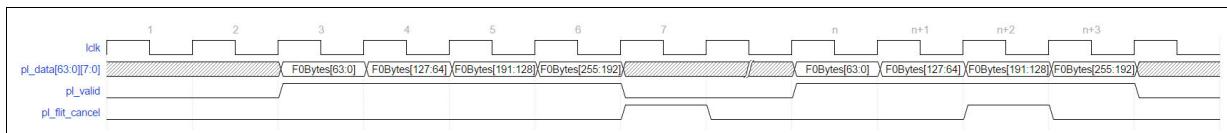


Figure 10-17. Example for `pl_flit_cancel` for Latency-Optimized Flits and CRC Error on Second Flit Half, Alternate Implementation Example

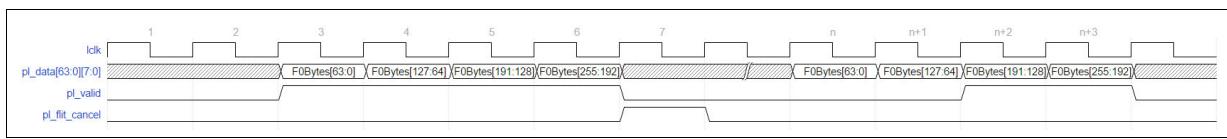
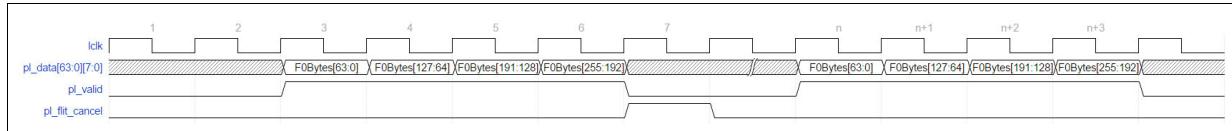


Figure 10-18 shows an example for a Standard 256B Flit. In this case, the CRC bytes are packed toward the end of the Flit and thus a CRC error on either of the two halves cancels the entire Flit. After replay flow, CRC passes, and the entire Flit is sent to the Protocol Layer without canceling it.

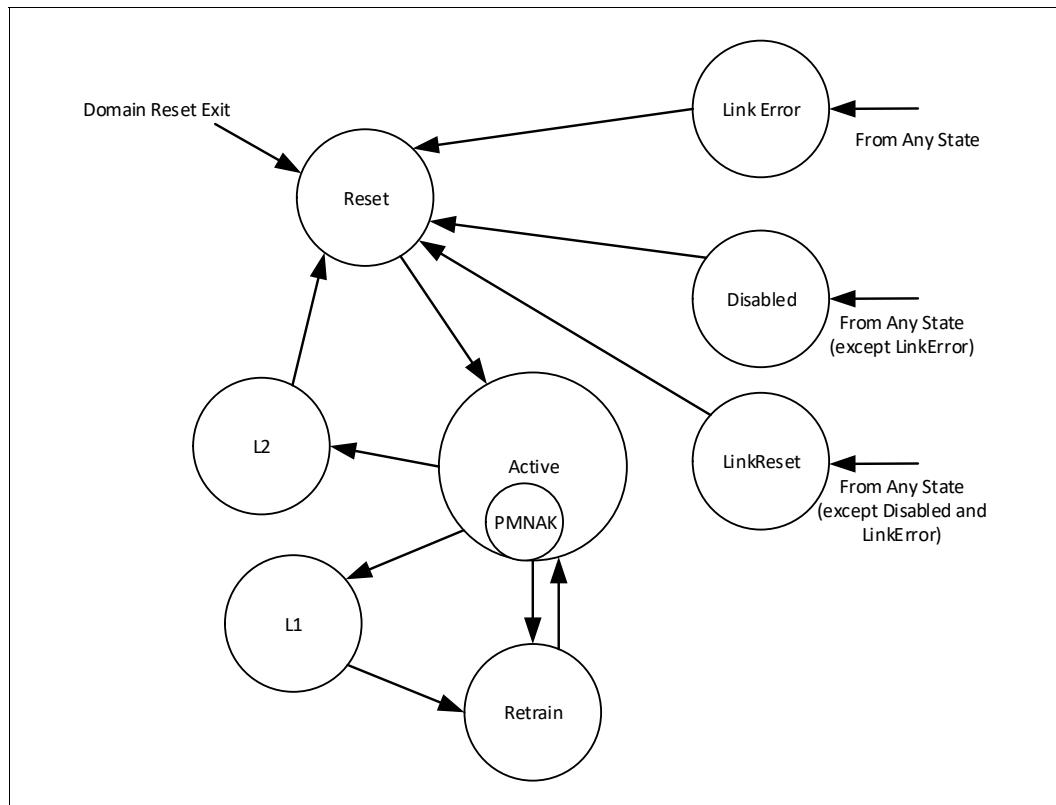
Figure 10-18. Example for pl_flt_cancel for Standard 256B Flits



10.2.6 FDI State Machine

Figure 10-19 shows the FDI state machine.

Figure 10-19. FDI State Machine



10.2.7 Rx_active_req/Sts Handshake

The Adapter negotiates Active state transitions on FDI using sideband messages when the Adapter LSM is exposed to the Protocol Layer. Since the sideband Link is running slower than the mainband Link, the Adapter needs to make sure that the Protocol Layer's Receiver is already in Active state (even though `pl_state_sts` might not have moved to Active yet) before responding to the Active request sideband message from remote Link partner. Rx_active_req/Sts handshake facilitates this.

When CXL is sent over UCIe, ARB/MUX functionality is performed by the Adapter and CXL vLSMs are exposed on FDI. Although ALMPs are transmitted over mainband, the interface to the Protocol Layer is FDI and it follows the rules of Rx_active_req/Sts Handshake as well.

Rules for this handshake:

1. The Adapter (or ARB/MUX) asserts `pl_rx_active_req` to trigger the Protocol Layer to open its Receiver's data path for receiving protocol data or Flits. This signal does not affect the Transmitter data path (it must wait for `pl_state_sts` to move to Active and follow the rules of `pl_trdy`). `pl_rx_active_req` should have a rising edge only when `lp_rx_active_sts` = 0 and `pl_state_sts` is Reset, Retrain or Active.
2. The Protocol Layer asserts `lp_rx_active_sts` after `pl_rx_active_req` has asserted and when it is ready to receive protocol data or Flits. There must be at least one clock cycle delay between `pl_rx_active_req` assertion and `lp_rx_active_sts` assertion to prevent a combinatorial loop.
3. When `pl_rx_active_req` = 1 and `lp_rx_active_sts` = 1, the Receiver is in Active state if `pl_state_sts` is Reset, Retrain, or Active.
4. `pl_rx_active_req` should have a falling edge only when `lp_rx_active_sts` = 1. This must trigger Protocol Layer to de-assert `lp_rx_active_sts`, and this completes the transition of the Receiver away from Active state.
5. For graceful exit from Active state (i.e., a transition to PM, Retrain, LinkReset or Disabled states), both `pl_rx_active_req` and `lp_rx_active_sts` must de-assert before `pl_state_sts` transitions away from Active.
6. If `pl_rx_active_req` = 0 while `pl_state_sts` = Active, the Adapter must guarantee no Flits would be sent to the Protocol Layer (for example, this can happen if the Adapter LSM or RDI is in Retrain, but the vLSM exposed to Protocol Layer is still in Active). Thus, it is permitted to perform this handshake even when the state status on FDI remains Active throughout.
7. For Active to LinkError transition, it is permitted for `pl_state_sts` to transition to LinkError before `pl_rx_active_req` de-asserts, but both `pl_rx_active_req` and `lp_rx_active_sts` must de-assert before `pl_state_sts` transitions away from LinkError.

10.2.8 FDI Bring up flow

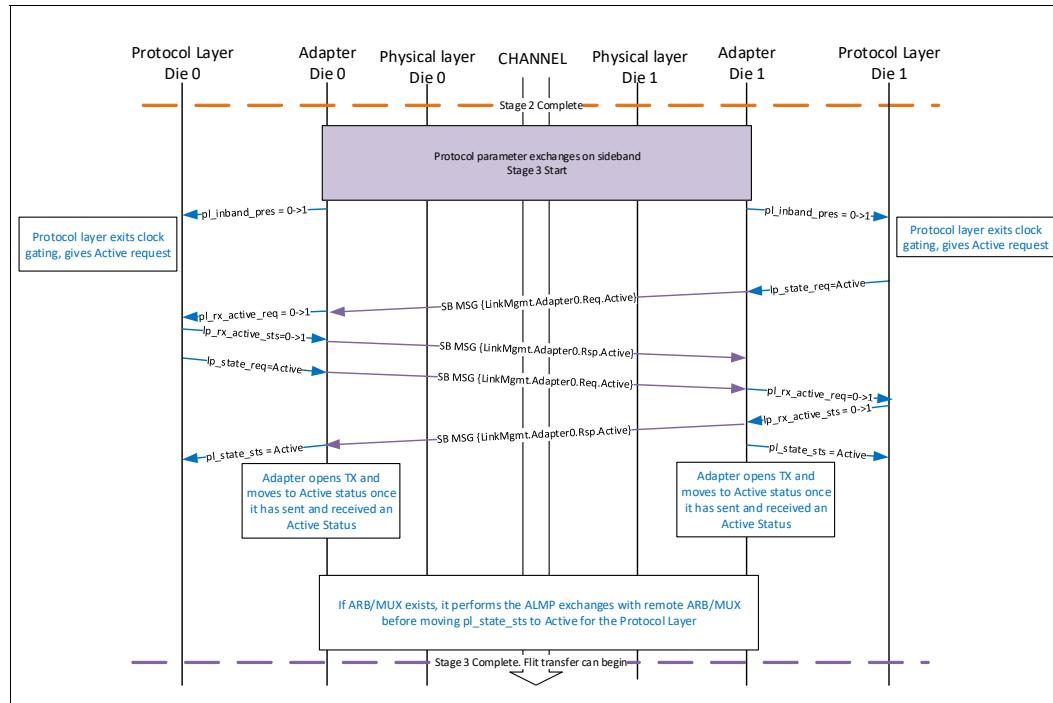
Figure 10-20 shows an example flow for Stage 3 of the Link bring up highlighting the transitions on FDI. This stage requires sequencing on FDI that coordinates the state transition from Reset to Active. If multiple stacks of protocol or ARB/MUX is present, the same sequence happens independently for each Protocol Layer stack. The flows on FDI are illustrated for Adapter 0 LSM in the sideband message encodings, however Adapter 1 LSM must send the sideband message encodings corresponding to Adapter 1 to its remote Link partner.

1. Once Adapter has completed transition to Active on RDI and successful parameter negotiation with the remote Link partner, it must do the `pl_clk_req` handshake with the Protocol Layer and reflect `pl_inband_pres`=1 on FDI. Note that the `pl_clk_req` handshake is not shown in the example flow in Figure 10-20
2. This is the trigger for Protocol Layer to request Active state. It is permitted for the Protocol Layer to wait until `pl_protocol_vld` = 1 before requesting Active. It must perform the `lp_wake_req` handshake as described in Section 10.2.3.1. Note that the `lp_wake_req` handshake is not shown in the example flow in Figure 10-20.
3. On sampling `lp_state_req` = Active, the Adapter must send the {LinkMgmt.Adapter0.Req.Active} sideband message to remote Link partner.

4. The Adapter must respond to the {LinkMgmt.Adapter.Req.Active} sideband message with a {LinkMgmt.Adapter.Rsp.Active} sideband message after making sure that the Protocol Layer's Receiver is ready. The {LinkMgmt.Adapter0.Rsp.Active} must only be sent after the Adapter has sampled `pl_rx_active_req = lp_rx_active_sts = 1`. As mentioned previously, the `pl_clk_req` handshake applies to `pl_rx_active_req` as well; it is permitted for the Adapter to keep `pl_clk_req` asserted continuously (once it has been asserted for `pl_inband_pres`) while doing the bring up flow. Note once the Adapter has sent the {LinkMgmt.Adapter0.Rsp.Active} sideband message, if it receives Flits from the remote Link partner, it must process them as applicable (i.e. for UCIE Flit mode, the Adapter must respond to the Sequence Number Handshake initiated by the remote Link or respond with Ack/Nak for Payload Flits. The Adapter will have to insert NOPs in case the `pl_state_sts` signal has not yet transitioned to Active).
5. If no ARB/MUX is present, once the Adapter has sent and received the {LinkMgmt.Adapter0.Rsp.Active} sideband message, it must transition `pl_state_sts` to Active for the Protocol Layer, and Flit transfer can begin (i.e., new Flits can be accepted from the Protocol Layer, and in UCIE Flit mode, the Adapter is permitted to initiate the Sequence Number Handshake Phase if it has not already done so as a result of Step 4).
6. If ARB/MUX is present, the sending and receipt of {LinkMgmt.Adapter0.Rsp.Active} sideband message opens up the ARB/MUX to perform ALMP exchanges over mainband and eventually transition the vLSMs to Active state.

Step 3 through **Step 6** constitute the “Active Entry Handshake” on FDI and must be performed for every entry to Active state. Active.PMNAK to Active transition is not considered here because Active.PMNAK is only a sub-state of Active.

Figure 10-20. FDI Bring up flow



10.2.9 FDI PM Flow

This section describes the sequencing and rules for PM entry and exit on FDI. The rules are the same for L1 or L2 entry. L1 exit transitions the state machine through Retrain to Active, whereas L2 exit transitions the state machine through Reset to Active. The flow illustrations in the section use L1 as an example. A “PM request” sideband message is {LinkMgmt.Adapter*.Req.L1} or {LinkMgmt.Adapter*.Req.L2}. A “PM Response” sideband message is {LinkMgmt.Adapter*.Rsp.L1} or {LinkMgmt.Adapter*.Rsp.L2}. The flows on FDI are illustrated for Adapter 0 LSM in the sideband message encodings; however, Adapter 1 LSM must send the sideband message encodings corresponding to Adapter 1 to its remote Link partner.

- The Protocol Layer requests PM entry on FDI after idle time criteria has been met. The criteria for idle time is implementation specific and could be dependent on the protocol. For PCIe and CXL.io protocols, PM DLLPs are **not** used to negotiate PM entry/exit when using D2D Adapter’s Retry buffer (i.e., UCIE Flit mode).
- If operating in UCIE Flit mode, ARB/MUX is present within the D2D Adapter, and it follows the rules of *CXL Specification* (for 256B Flit mode) to take the vLSMs to the corresponding PM state. Note that even for CXL 64B Flit mode, the same ALMP rules as 256B Flit mode are used. This is a simplification on UCIE, because ALMPs always go through the retry buffer. Once vLSMs are in the PM state, ARB/MUX requests the Adapter Link State Machine to enter the corresponding PM state. The Adapter Link State Machine transition to PM follows the same rules as outlined for Protocol Layer and Adapter below.
- If CXL or PCIe protocol has been negotiated, only the upstream port (UP) can initiate PM entry. This is done using a sideband message from the UP Adapter to the downstream port (DP) Adapter. DP Adapter must not initiate entry into PM. PM support is required for CXL and PCIe protocols. PM entry is considered successful and complete once UP receives a valid “PM Response” sideband message. [Figure 10-21](#) shows an example flow for CXL or PCIe protocol PM Entry on FDI and Adapter. Once the FDI status is PM for all Protocol Layers, the Adapter can request PM transition on RDI.
- If Streaming protocol has been negotiated, OR UCIE is in Raw Format, OR Management Transport protocol was negotiated over the mainband without CXL or PCIe, then both side Adapters must issue a PM entry request through a sideband message once the conditions of PM entry have been satisfied. PM entry is considered successful and complete once both sides have received a valid “PM Response” sideband message. [Figure 10-22](#) shows an example flow for symmetric protocols. Once the FDI status is PM for all Protocol Layers, the Adapter can request PM transition on the RDI.
- Protocol Layer requests PM entry once it has blocked transmission of any new Protocol Layer Flits, by transitioning `lp_state_req` to L1 or L2 encoding. Once requested, the Protocol Layer cannot change this request until it observes the corresponding PM state, Retrain, Active.PMNAK or LinkError state on `pl_state_sts`; unless it is a DP Protocol Layer for PCIe or CXL. Once the FDI state is resolved, the Adapter must first bring it back to Active before processing any new PM requests from the Protocol Layer.
 - If the resolution is PM (upon successful PM entry) and the Protocol Layer needs to exit PM (or there is a pending Protocol Layer Active request from remote Link partner) then the Protocol Layer must initiate PM exit flow on FDI by requesting `lp_state_req` = Active. All PM entry related handshakes must have finished prior to this (for CXL/PCIe protocols, this is when UP has received a valid “PM Response” sideband message. For symmetric protocols, this is when both sides Adapter have received a valid “PM Response” sideband message).
 - If the resolution is Active.PMNAK, the Protocol Layer must initiate a request of Active on FDI. Once the status moves to Active, the Protocol Layer is permitted to re-request PM entry (if all conditions of PM entry are still met). [Figure 10-23](#) shows an example of PM abort flow. The PM

request could have been from either side depending on the configuration. Protocol Layer must continue receiving protocol data or Flits while the status is Active or Active.PMNAK.

- DP Protocol Layer for PCIe or CXL is permitted to change request from PM to Active without waiting for PM or Active.PMNAK (the DP FDI will never have `p1_state_sts=Active.PMNAK` since it does not send “PM Request” sideband messages); however, it is still possible for the Adapter to initiate a stallreq/ack and complete PM entry if it was in the process of committing to PM entry when the Protocol Layer changed its request. In this scenario, the Protocol Layer will see `p1_state_sts` transition to PM and it is permitted to continue asking for the new state request.
- If the resolution is LinkError, then the Link is down and it resets any outstanding PM handshakes.
- Adapter (UP port only if CXL or PCIe protocol), initiates a “PM request” sideband message once it samples a PM request on `lp_state_req` and has completed the StallReq/Ack handshake with the corresponding Protocol Layer and its Retry buffer is empty of Flits from the Protocol Layer that is requesting PM (all pending Acknowledgments have been received).
- If the Adapter LSM moves to Retrain while waiting for a “PM Response” sideband message, it must wait for the response. Once the response is received, it must transition back to Active before requesting a new PM entry. Note that the transition to Active requires Active Entry handshake with the remote Link partner, and that will cause the remote partner to exit PM. If the Adapter LSM receives a “PM Request” sideband message after it has transitioned to Retrain, it must immediately respond with {LinkMgmt.Adapter0.Rsp.PMNAK}.

Note: The precise timing of the remote Link partner that is observing Link Retrain is unknown; thus, the safer thing to do is to go to Active and redo the PM handshake when necessary for this scenario. There is a small probability that there might be an exit from PM and re-entry back in PM under certain scenarios.

- Once the Adapter receives a “PM request” sideband message, it must respond to it within 2 us (the time is only counted during the Adapter LSM being in Active state):
 - if its local Protocol Layer is requesting PM, it must respond with the corresponding “PM Response” sideband message after finishing the StallReq/Ack handshake with its Protocol Layer and its Retry buffer being empty. If the current status is not PM, it must transition `p1_state_sts` to PM after responding to the sideband message.
 - If the current `p1_state_sts` = PM, it must respond with “PM Response” sideband message.
 - If `p1_state_sts` = Active and `lp_state_req` = Active and it remains this way for 1us after receiving the “PM Request” sideband message, it must respond with {LinkMgmt.Adapter0.Rsp.PMNAK} sideband message. The time is only counted during all the relevant state machines being in Active state.
 - If the Adapter receives a “PM Response” sideband message in response to a “PM Request” sideband message, it must transition `p1_state_sts` on its local FDI to PM (if it is currently in Active state).
 - If the Adapter receives a {LinkMgmt.Adapter0.Rsp.PMNAK} sideband message in response to a “PM Request” sideband message, it must transition `p1_state_sts` on its local FDI to Active.PMNAK state if it is currently in Active state. If it is not in Active state, no action needs to be taken. It is permitted to retry PM entry handshake (if all conditions of PM entry are satisfied) at least 2us after receiving the {LinkMgmt.Adapter0.Rsp.PMNAK} sideband message OR if it received a corresponding “PM Request” sideband message from the remote Link partner.
 - PM exit is initiated by the Protocol Layer requesting Active on FDI. After RDI is in Active, triggers the Adapter to initiate PM exit by performing the Active Entry handshakes on sideband.
- [Figure 10-24](#) shows an example flow of PM exit on FDI when Adapter LSM is exposed.

- PM exit handshake completion requires both Adapters to send as well as receive a {LinkMgmt.Adapter0.Rsp.Active} sideband message. Once this has completed, the Adapter is permitted to transition Adapter LSM to Active.
- If **pl_state_sts = PM** and a {LinkMgmt.Adapter0.Req.Active} sideband message is received, the Adapter must initiate **pl_clk_req** handshake with the Protocol Layer, and transition Adapter LSM to Retrain (For L2 exit, the transition is to Reset). This must trigger the Protocol Layer to request Active on **lp_state_req** (if not already doing so), and this in turn triggers the Adapter to send {LinkMgmt.Adapter0.Rsp.Active} sideband message to the remote Link partner.

Note that the following figures are examples and do not show the **lp_wake_req**, **pl_clk_req**, and/or **pl_rx_active_req** handshakes. Implementations must follow the rules outlined for these handshakes in previous sections.

Figure 10-21. PM Entry example for CXL or PCIe protocols

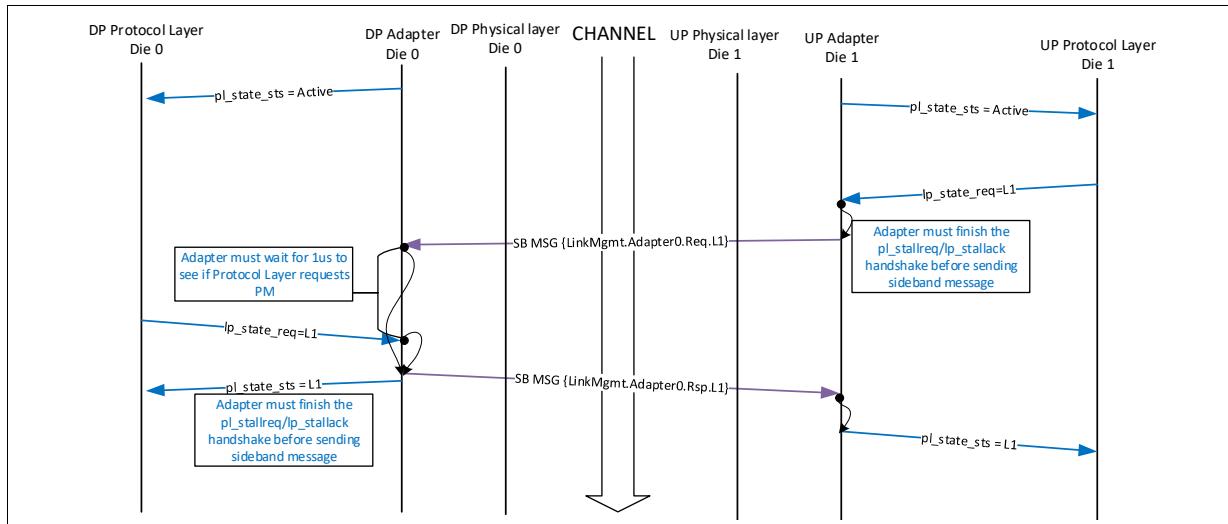


Figure 10-22. PM Entry example for symmetric protocol

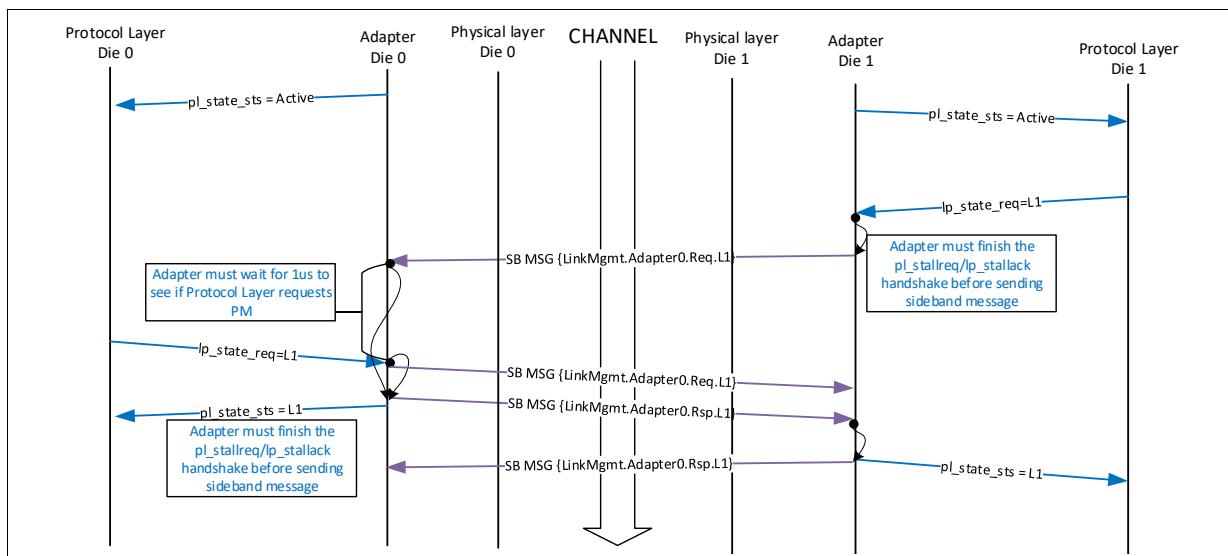
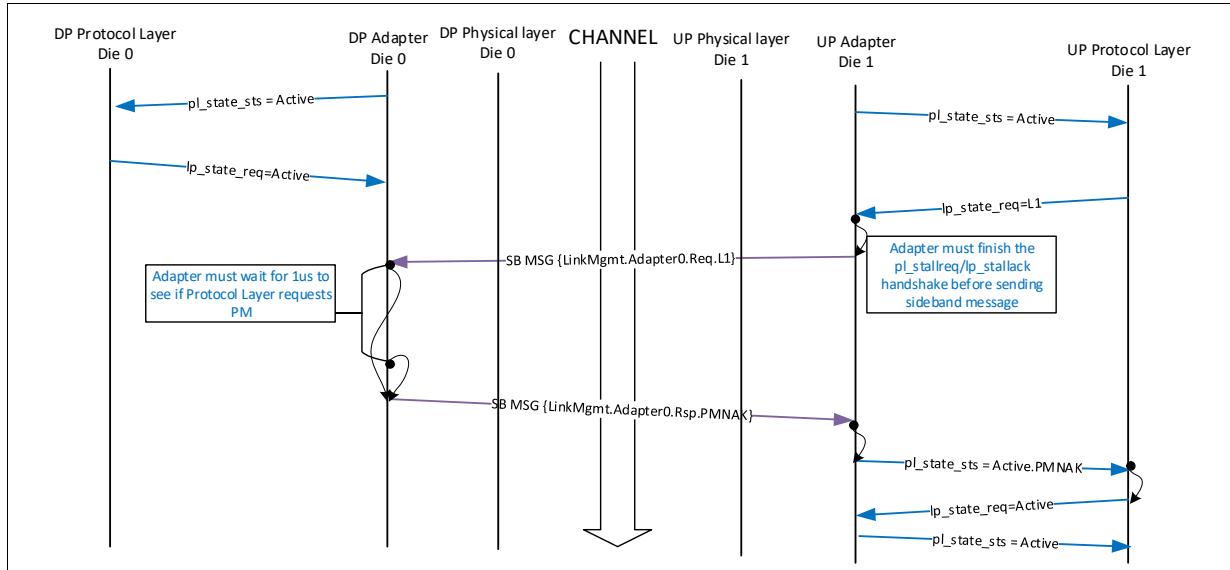
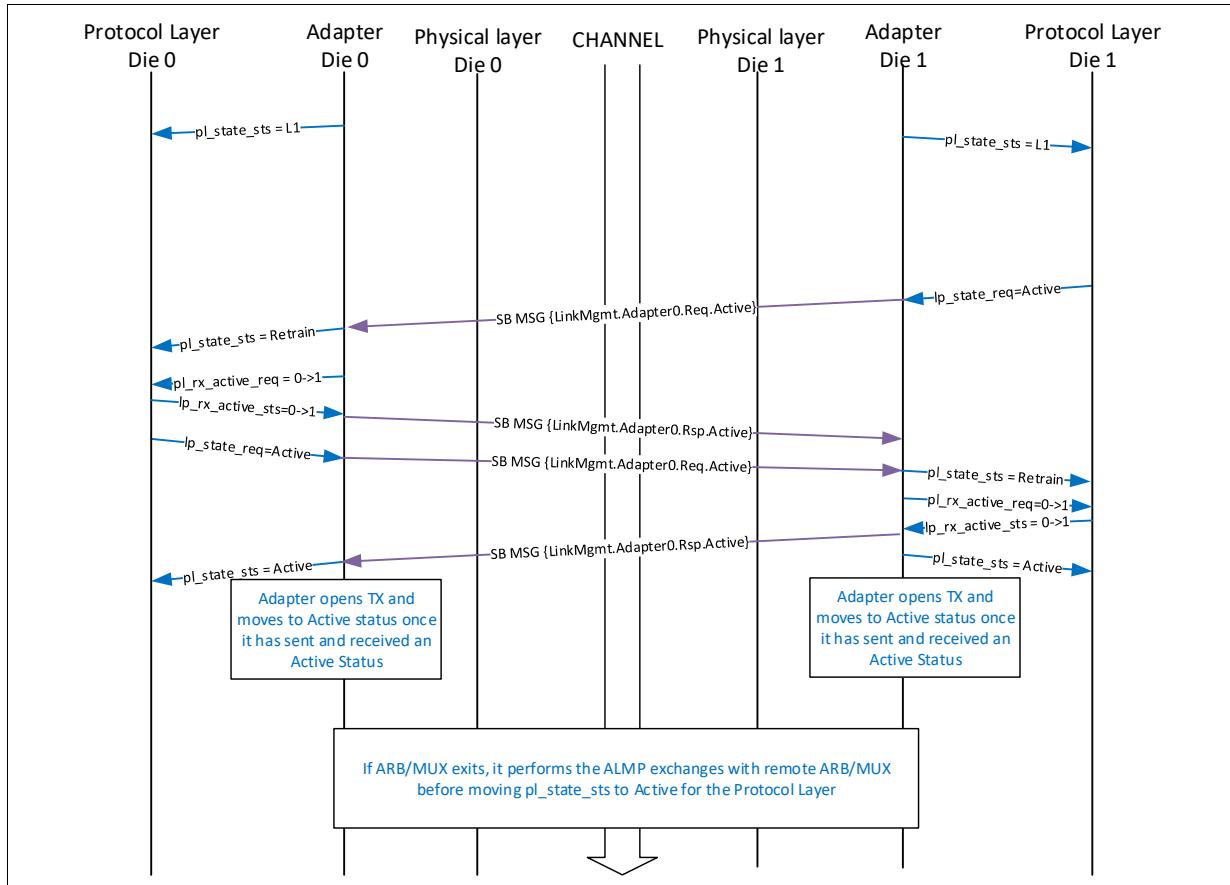


Figure 10-23. PM Abort Example**Figure 10-24. PM Exit Example**

10.3 Common rules for FDI and RDI

This section covers common set of rules applicable to FDI and RDI and cross-interactions between them. Any applicable differences are called out as well. To have common terminology for the common set of rules, Upper Layer is used to refer to Adapter for RDI, and Protocol Layer for FDI. Lower Layer is used to refer to Physical Layer for RDI and Adapter for FDI.

Because Active.PMNAK is a sub-state of Active, all rules that apply for Active are also applicable for Active.PMNAK; however the state status cannot move from Active.PMNAK directly to L1 or L2 due to the rules requiring the Upper Layer to request a transition to Active before requesting PM again.

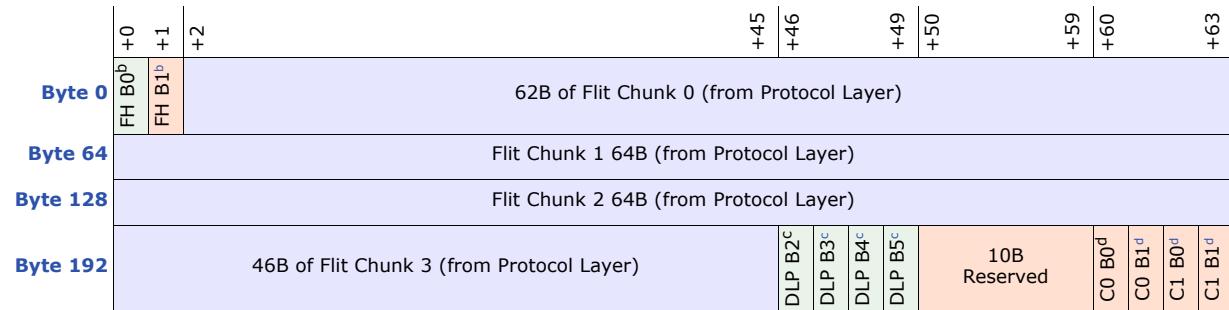
10.3.1 Byte Mapping for FDI and RDI

The Flit Format figures in [Chapter 3.0](#) show examples of how a Flit is laid out on a 64B datapath when sent over FDI or RDI. [Figure 10-25](#) shows an example of a CXL.io Standard 256B Start Header Flit for reference. Each Flit takes four data transfers across FDI or RDI when the data width is 64 Bytes. Each data transfer is referred to a Flit Chunk, numbered in increasing order within an entire Flit transfer.

For every data transfer, the Least Significant Byte from the corresponding Flit Chunk is mapped to Byte 0 on FDI (or RDI), the next Byte from the Flit is mapped to Byte 1 on FDI (or RDI), and so on. Within each Byte, bit 0 of the Byte from the Flit maps to bit 0 of the corresponding Byte on FDI (or RDI), and so on. The same mapping applies for both transmit and receive directions.

For example, in Transfer 0, Byte 0 of the Flit is mapped to Byte 0 of FDI (or RDI), Byte 1 of the Flit is mapped to Byte 1, and so on. In transfer 1, Byte 64 of the Flit is mapped to Byte 0 of FDI (or RDI), Byte 65 of the Flit is mapped to Byte 1 of FDI (or RDI) and so on. This example is illustrated in [Figure 10-26](#). Data transfers follow the rules outlined in [Section 10.1.4](#) for RDI and [Section 10.2.4](#) for FDI and hence do not necessarily correspond to consecutive clock cycles.

Figure 10-25. CXL.io Standard 256B Start Header Flit Format Example^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 10-26. FDI (or RDI) Byte Mapping for 64B Datapath to 256B Flits

Transfer (Rows)	FDI (or RDI) Bytes (Columns)							
	0	1	2	...	60	61	62	63
0	Flit Byte 0	Flit Byte 1	Flit Byte 2	...	Flit Byte 60	Flit Byte 61	Flit Byte 62	Flit Byte 63
1	Flit Byte 64	Flit Byte 65	Flit Byte 66	...	Flit Byte 124	Flit Byte 125	Flit Byte 126	Flit Byte 127
2	Flit Byte 128	Flit Byte 129	Flit Byte 130	...	Flit Byte 188	Flit Byte 189	Flit Byte 190	Flit Byte 191
3	Flit Byte 192	Flit Byte 193	Flit Byte 194	...	Flit Byte 252	Flit Byte 253	Flit Byte 254	Flit Byte 255

If the FDI or RDI datapath width is increased (or decreased), the Byte mapping follows the same convention of increasing order of Flit bytes mapped to increasing order of FDI (or RDI) bytes.

Figure 10-27 shows an illustration of a 128B data path.

Figure 10-27. FDI (or RDI) Byte Mapping for 128B Datapath to 256B Flits

Transfer (Rows)	FDI (or RDI) Bytes (Columns)											
	0	1	2	...	62	63	64	65	...	125	126	127
0	Flit Byte 0	Flit Byte 1	Flit Byte 2	...	Flit Byte 62	Flit Byte 63	Flit Byte 64	Flit Byte 65	...	Flit Byte 125	Flit Byte 126	Flit Byte 127
1	Flit Byte 128	Flit Byte 129	Flit Byte 130	...	Flit Byte 190	Flit Byte 191	Flit Byte 192	Flit Byte 193	...	Flit Byte 253	Flit Byte 254	Flit Byte 255

For 68B Flit Formats, the Protocol Layer transfers only 64B of payload information from the Flit over FDI (the Flit Header and CRC are inserted by the Adapter). Thus, if the datapath is 128B wide, two such transfers will happen at a given clock cycle as shown in Figure 10-28. The numbering in the figure still uses the Byte positions relative to the overall Flit, hence Byte 0 corresponds to Flit 0 Byte 2, etc. On the Transmit path, the Protocol Layer inserts empty slots (i.e., bytes with a value of 00h) to populate the entire width of the bus if the interface width is greater than 64B and there is insufficient payload information to transmit. The Adapter does the same on the Receive path.

Figure 10-28. FDI Byte Mapping for 128B Datapath for 68B Flit Format

Transfer (Rows)	FDI Bytes (Columns)											
	0	1	2	...	62	63	64	65	...	125	126	127
0	Flit 0 Byte 2	Flit 0 Byte 3	Flit 0 Byte 4	...	Flit 0 Byte 64	Flit 0 Byte 65	Flit 1 Byte 2	Flit 1 Byte 3	...	Flit 1 Byte 63	Flit 1 Byte 64	Flit 1 Byte 65

For 68B Flit Formats, Adapter inserts the Flit Header and CRC bytes, and performs the necessary shifting before transferring the bytes over RDI. Thus, if the data path is 128B wide, the byte mapping will follow as shown in Figure 10-29. The remainder of Flit 1 continues on the next transfer, etc. Given that the Adapter must insert PDS bytes before pausing the data stream, which makes the transfer a multiple of 256B, the transfer naturally aligns when the width of RDI is 64B, 128B, or 256B on both the Transmit and Receive directions. For wider than 256B interfaces, see the Implementation Note below.

Figure 10-29. RDI Byte Mapping for 128B Datapath for 68B Flit Format

Transfer (Rows)	RDI Bytes (Columns)											
	0	1	2	...	65	66	67	68	...	125	126	127
0	Flit 0 Byte 0	Flit 0 Byte 1	Flit 0 Byte 2	...	Flit 0 Byte 65	Flit 0 Byte 66	Flit 0 Byte 67	Flit 1 Byte 0	...	Flit 1 Byte 57	Flit 1 Byte 58	Flit 1 Byte 59
1	Flit 1 Byte 60	Flit 1 Byte 61	Flit 1 Byte 62

The frequency of operation of the interfaces along with the data width determines the maximum bandwidth that can be sustained across the FDI (or RDI) interface. For example, a 64B datapath at 2 GHz of clock frequency is required to sustain a 16 GT/s Link for an Advanced Package configuration with a single module. Similarly, to scale to 32 GT/s of Link speed operation for Advanced Package configuration with a single module, a 128B datapath running at 2 GHz would be required to support the maximum Link bandwidth.

The FDI (or RDI) byte mapping for the transmit or receive direction does not change for multi-module configurations. The MMPL logic within the Physical Layer is responsible for ensuring that the bytes are transmitted in the correct order to the correct module. Any byte swizzling or rearrangement to resolve module naming conventions, etc., is thus the responsibility of the MMPL logic.

IMPLEMENTATION NOTE — NBYTES

For Raw Format, the value of NBYTES is vendor-defined. This Implementation Note is for UCIe Flit mode.

It is strongly recommended that when operating in UCIe Flit mode, NBYTES is chosen to be one of 64, 128, 256, or 512 and is selected to get the best KPI (e.g., latency, area, etc.) for the desired bandwidth from the UCIe Link. If NBYTES is chosen to be larger than or equal to 512, it is strongly recommended that it is a multiple of 256 and is only done for the case of a four module Advanced Package Link designed for 16 GT/s or higher. Data transfer over the Link for all Flit formats defined in UCIe Flit mode are in a granularity of 256B, so aligning to a multiple of that avoids unnecessary shifting and corresponding tracking.

For situations in which the RDI or FDI data path is wider than 256B, the following considerations apply for interoperability:

- On the Transmit side, it is required to send valid data corresponding to the full width of the interface. For FDI, this would mean the Protocol Layer might need to pack a Protocol Flit with empty slots. For RDI, this would mean the Adapter might need to insert NOP Flits (for 68B Flit Format, PDS bytes are also included as valid data for this purpose).
- On the Receive side, for RDI:

It is possible that the Physical Layer has to wait to accumulate sufficient bytes before transmitting over RDI. The Physical Layer must accumulate data in multiples of 256B and if the accumulated data is less than the RDI width, it must wait for a sufficient gap in valid data transfer on the Physical Link (at least 16 UI for differential clock and 32 UI for quadrature clock) before transmitting this data on RDI. In this scenario, the accumulated data is sent on the lower significant bytes of the RDI, and any remaining bytes on the interface are assigned to all 0s.

For 256B Flit Formats, a Flit Header which is 0000h with a CRC of 0000h is silently discarded by the Adapter. It is also not included for the purposes of Runtime Link Testing.

For 68B Flit Formats, the Adapter is expected to keep track of the PDS bytes (because these are included in Runtime Link Testing). Any extra padding beyond that is silently discarded and not included for the purposes of Runtime Link Testing.

- On the Receive side, for FDI:

The Adapter must accumulate data in multiples of 256B before forwarding to the Protocol Layer. If the accumulated data is less than the FDI width, it gets sent on the lower significant bytes of the FDI, and any remaining bytes on the interface are assigned to 0b.

For 256B Flit Formats, a Flit Header of 0000h is a NOP for the Protocol Layer and is discarded.

For 68B Flit Formats, 00h are IDLE symbols for PCIe/CXL.io or Empty slots for CXL.cachemem, both of which get discarded by the Protocol Layer. For Streaming protocols that use 68B Flit Formats, it is recommended to use the same approach.

- `lp_corrupt_crc`, `pl_flit_cancel`, and `pl_error` apply to all the Flits that are transferred at the corresponding clock cycle. If applicable, it is recommended to set NDLLP to 32 for these applications and limit the DLLP throughput to be 1 per clock cycle on FDI.

10.3.2 Stallreq/Ack Mechanism

The Stallreq/Ack mechanism is used by the Lower Layer to interrupt the Flit transfers by the Upper Layer at a Flit boundary. On RDI, the Stallreq/Ack mechanism must be used when exiting Active state to Retrain, PM, LinkReset or Disabled states. On FDI, for UCIe Raw Format, the Stallreq/Ack mechanism must be used when exiting Active state to Retrain, PM, LinkReset or Disabled states. On FDI, for UCIe Flit Mode, the Stallreq/Ack mechanism must only be used when exiting Active State to a PM state. For other scenarios that exit Active state for UCIe Flit mode, the Adapter must simply de-assert **p1_trdy** at a Flit boundary before state transition.

The Stallreq/Ack mechanism is mandatory for all FDI and RDI implementations. **lp_stallack** assertion implies that Upper Layer has stalled its pipeline at a Flit aligned boundary.

The **p1_stallreq/lp_stallack** handshake is a four-phase sequence that follows the rules below:

1. The **p1_stallreq** and **lp_stallack** must be de-asserted before domain reset exit.
2. A rising edge on **p1_stallreq** must only occur when **lp_stallack** is de-asserted.
3. A falling edge on **p1_stallreq** must only occur when **lp_stallack** is asserted or when the domain is in reset.
4. A rising edge on **lp_stallack** must only occur when **p1_stallreq** is asserted.
5. A falling edge on **lp_stallack** must only occur when **p1_stallreq** is de-asserted or when domain is in reset.
6. When **lp_stallack** is asserted **lp_valid** and **lp_irdy** must both be de-asserted.
7. While **p1_stallreq** is asserted, any data presented on the interface must be accepted by the physical layer until the rising edge of **lp_stallack**. **p1_trdy** is not required to be asserted consecutively.
8. The logic path between **p1_stallreq** and **lp_stallack** must contain at least one flip-flop to prevent a combinatorial loop.
9. A complete stallreq/stallack handshake is defined as the completion of all four phases: Rising edge on **p1_stallreq**, rising edge on **lp_stallack**, falling edge on **p1_stallreq**, falling edge on **lp_stallack**.
10. It is strongly recommended that Upper Layer implements providing **lp_stallack** on a global free running clock so that it can finish the handshake even if the rest of its logic is clock gated.

To avoid performance penalties, it is recommended that this handshake be completed as quickly as possible while satisfying the above rules.

IMPLEMENTATION NOTE

In multiple places within this specification, for state transitions, it is referring to completing the Stallreq/Ack handshake before the state transition. In the context of state transitions, there are two acceptable ways to implement this from the lower layer:

- One implementation from the lower layer would follow the sequence:
 - i. Assert `pl_stallreq`.
 - ii. After `lp_stallack` is asserted, perform the necessary actions for state transition (including deassertion of `pl_trdy`).
 - iii. De-assert `pl_stallreq`. Once `lp_stallack` de-asserts, the state transition is considered complete.
- The alternate implementation from the lower layer would follow the sequence:
 - i. Assert `pl_stallreq`.
 - ii. After `lp_stallack` is asserted, de-assert `pl_trdy`.
 - iii. De-assert `pl_stallreq` and perform the necessary actions for state transition.

State transition is considered complete after `pl_state_sts` update and `lp_stallack` de-assertion.

10.3.3 State Request and Status

Table 10-4 describes the Requests considered by the Lower layer in each of the interface states. The Upper layer must take into account the interface state status and make the necessary request modifications.

The requests are listed on the Row and the state status is listed in the Column.

The entries in Table 10-4 denote the following:

- Yes: Indicates that the request is considered for next state transition by the lower layer.
- N/A: Not Applicable
- Ignore: Indicates that the request is ignored and has no effect on the next state transition.

Table 10-4. Requests Considered in Each State by Lower Layer

Request (Row) Versus Status (Column)	Reset	Active	L1	LinkReset	Retrain	Disable	L2	LinkError
NOP	Yes	Ignore	Ignore	Ignore	Ignore	Ignore	Ignore	Ignore
Active	Yes ^a	Ignore ^b	Yes	Yes	Yes	Yes	Yes	Yes
L1	Ignore	Yes	Ignore	Ignore	Ignore	Ignore	Ignore	Ignore
LinkReset	Yes ^a	Yes	Yes	Ignore	Yes	Ignore	Yes	Ignore
Retrain	Ignore	Yes	Ignore	Ignore	Ignore	Ignore	Ignore	Ignore
Disable	Yes ^a	Yes	Yes	Yes	Yes	Ignore	Yes	Ignore
L2	Ignore	Yes	Ignore	Ignore	Ignore	Ignore	Ignore	Ignore
LinkError (sideband wire)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

a. Requires request transition from NOP

b. If the Status is Active.PMNAK, then the Lower Layer transitions to Active upon sampling the Active Request.

10.3.3.1 Reset State rules

The Reset State can be entered on de-assertion of interface reset signal or from LinkReset/Disable/LinkError/L2 states. In Reset state, the physical layer is permitted to begin its initialization/training process.

The **p1_state_sts** is not permitted to exit Reset state until requested by the upper layer. The exit from Reset state is requested by the upper layer by changing the **lp_state_req** signal from NOP encoding value to the permitted next state encoding value.

The rules for Reset state transition are as follows:

1. Reset→Active: The lower layer triggers transitions to the Active state upon observing **lp_state_req == Reset (NOP)** for at least one clock while **p1_state_sts** is indicating Reset followed by observing **lp_state_req == Active**. The transition to Active is only completed once the corresponding Active Entry handshakes have completed on the Link. For RDI, it is when the Physical Layer has sent and received an Active Response sideband message to and from the remote Physical Layer respectively. For the Adapter LSM, it is when the Adapter has sent and received an Active Status sideband message to and from the remote Adapter respectively. For the ARB/MUX vLSM, it is when the ARB/MUX has sent and received an Active Status ALMP to and from the remote ARB/MUX respectively.
2. Reset→LinkReset: The lower layer transitions to the LinkReset state upon observing **lp_state_req == Reset (NOP)** for at least one clock while **p1_state_sts** is indicating Reset followed by observing **lp_state_req == LinkReset OR** when requested by remote Link partner through the relevant sideband message. The lower layer is permitted to transition through Active State, and when it does, Active state exit conditions apply.
3. Reset→Disabled: The lower layer transitions to the Disabled state upon observing **lp_state_req == Reset (NOP)** for at least one clock while **p1_state_sts** is indicating Reset followed by observing **lp_state_req == Disabled OR** when requested by the remote Link partner through the relevant sideband message. The lower layer is permitted to transition through Active State, and when it does, Active state exit conditions apply.
4. Reset→LinkError: The lower layer transitions to LinkError based on observing an internal request to move to the LinkError or **lp_linkerror** assertion. For RDI, this transition is permitted if requested by the remote Link partner through the relevant sideband message.

10.3.3.2 Active State rules

The Active state to next state transitions are described below.

The rules for Active State transition are as follows:

1. Active→Retrain: The Lower layer transitions to the Retrain state upon observing `lp_state_req == Retrain` or due to an internal request to retrain the Link while `pl_state_sts == Active`. This arc is not applicable for CXL vLSMs exposed on FDI (CXL Flit Mode with Retry in the Adapter).
2. Active→L1: The physical layer transitions to L1 based on observing `lp_state_req == L1` while in the Active state, if other conditions of PM entry have also been satisfied.
3. Active→L2: The physical layer transitions to L2 based on observing `lp_state_req == L2` while in the Active state, if other conditions of PM entry have also been satisfied.

It is permitted to have an Active.PMNAK to Retrain/LinkReset/Disable/LinkError transition for cases where Lower Layer is waiting for the Upper Layer to change the request to Active and the corresponding Link event triggers it. There is no scenario where there is a transition from Active.PMNAK to L1 or L2.

[Section 10.3.3.8](#) describes the transition from Active or Active.PMNAK to LinkReset, Disable, or LinkError states.

10.3.3.3 PM Entry/Exit Rules

See the PM entry and exit sequences in the RDI and FDI sections.

10.3.3.4 Retrain State Rules

Adapter requests Retrain on RDI if any of the following events occur:

- Software writes to the Retain bit and the Link is in Active state.
- Number of CRC or parity errors detected crosses a threshold. The specific algorithm for determining this is implementation specific.
- Protocol Layer requests Retrain (only applicable for UCIe Raw Format).
- any other implementation specific condition (if applicable).

Physical Layer triggers a Retrain transition on RDI if:

- Valid framing errors are observed
- Remote Physical Layer requests Retrain entry
- Adapter requests Retrain

Protocol Layer must not request Retrain on FDI, unless UCIe is operating in UCIe Raw Format.

A Retrain transition on RDI must always be propagated to Adapter LSMs that are in Active. Retrain transitions of the UCIe Link are not propagated to CXL vLSMs. Upon Retrain entry, the credit counter for UCIe Retimer (if present) must be reset to the value advertised during initial Link bring up (the value is given by the "Retimer_Credits" Parameter in the {AdvCap.Adapter} sideband message during initial Link bring up). The Retimer must drain or dump any Flits in flight or its internal transport buffers upon entry to Retrain. Additionally, the Retimer must trigger Retrain of the remote UCIe Link (across the Off-Package Interconnect).

Entry into Retrain state resets power management state for the corresponding state machine, and power management entry if required must be re-initiated after the interface enters Active state. If

there was an outstanding PM request that returns PM Response, the corresponding state machine must perform Active Entry handshakes to bring that state machine back to Active.

The rules for Retrain state transition are as follows:

1. Retrain→Active: If Retrain was entered from L1, the lower layer begins Active Entry handshakes upon observing `lp_state_req == Active` while `p1_state_sts == Retrain`. If Retrain was entered from Active, the lower layer begins Active Entry handshakes only after observing a NOP->Active transition on `lp_state_req`. Lower layer transitions to Active once the corresponding Active Entry handshakes have completed. Exit from Retrain on RDI requires the Active Entry handshakes to have completed between Physical Layers. Exit from Retrain on FDI must ensure that RDI has moved back to Active, and Active Entry handshakes have successfully completed between Adapters (for the Adapter LSM).
2. Transitional state: The lower layer is permitted to transition to the Active state upon observing `lp_state_req == LinkReset` or `Disabled` while `p1_state_sts == Retrain`. Following the entry into Active the lower layer is permitted to make a transition to the requested state.

[Section 10.3.3.8](#) describes Retrain exit to LinkReset, Disable, or LinkError states.

Note: The requirement to wait for NOP->Active transition ensures that the Upper Layer has a way to delay Active transition in case it is waiting for any relevant sideband handshakes to complete (for example the Parity Feature handshake).

10.3.3.5 LinkReset State Rules

LinkReset is used for reset flows (HotReset equivalent in PCIe, Protocol Layer must use this to propagate SBR to the device as well) to convey device and/or Link Reset across the UCIe Link.

Adapter triggers LinkReset transition upon observing a LinkReset request from the Protocol Layer, OR on receiving a sideband message requesting LinkReset entry from the remote Link partner OR an implementation specific internal condition (if applicable). Implementations must make best efforts to gracefully drain the Retry buffers when transitioning to LinkReset, however, entry to LinkReset must not timeout on waiting for the Retry buffer to drain. The Protocol Layer and Adapter must drain/flush their pipelines and retry buffer of the Flits for the corresponding Protocol Stack once the FDI state machines have entered LinkReset.

If all the FDI state machines and Adapter LSMS are in LinkReset, the Adapter triggers RDI to enter LinkReset as well.

The rules for LinkReset State transitions are as follows:

1. LinkReset→Reset: The lower layer transitions to the Reset state due to an internal request to move to Reset (example reset pin trigger) or `lp_state_req == Active` while `p1_state_sts == LinkReset` and all necessary actions with respect to LinkReset have been completed.
2. LinkReset→Disabled: The lower layer transitions to Disabled based on observing `lp_state_req == Disabled` or due to an internal request to move to Disabled while `p1_state_sts == LinkReset`.
3. Transitional State: The PHY is permitted to transition through Reset State, and when it does, Reset state exit conditions apply.
4. LinkReset→LinkError: The lower layer transitions to LinkError due to an internal request to move to LinkError or `lp_linkerror` assertion while `p1_state_sts == LinkReset`.

10.3.3.6 Disabled State Rules

Adapter triggers Disabled entry when any of the following events occur:

- Protocol Layer requests entry to Disabled state
- Software writes to the Link Disable bit corresponding to the underlying Protocol (e.g., the Link Disable bit in the Link Control register in PCIe)
- Remote Link partner requests entry to Disabled state through the relevant sideband message
- An implementation specific internal condition (if applicable)

Implementations must make best efforts to gracefully drain the Retry buffers when transitioning to Disabled, however, entry to Disabled must not timeout on waiting for the Retry buffer to drain. The Protocol Layer and Adapter must drain/flush their pipelines and retry buffer of the Flits for the corresponding Protocol Stack once the FDI state machines have entered Disabled.

If all the FDI state machines and Adapter LSMs are in Disabled, the Adapter triggers RDI to enter Disabled as well.

The rules for Disabled State are as follows:

- Disabled→Reset: The lower layer transitions to the Reset state due to an internal request to move to Reset (example reset pin trigger) or `lp_state_req == Active` while `p1_state_sts == Disabled` and all necessary actions with respect to Disabled transition have completed.
- Disabled→LinkError: The lower layer transitions to LinkError due to an internal request to move to LinkError or `lp_linkerror` assertion while `p1_state_sts == Disabled`.

10.3.3.7 LinkError State Rules

The lower layer enters LinkError state when directed by an `lp_linkerror` signal or due to Internal LinkError conditions. For RDI, the entry is also triggered if the remote Link partner requested LinkError entry through the relevant sideband message. It is not required to complete the stallreq/ack handshake before entering this state. However, for implementations where LinkError state is not a terminal state (terminal implies SoC needs to go through reset flow after reaching LinkError state), it is expected that software can come and retrain the Link after clearing error status registers, etc., and the following rules should be followed:

- If the lower layer decides to perform a `p1_stallreq/lp_stallack` handshake, it must provide `p1_trdy` to the upper layer to drain the packets. In cases where there is an uncorrectable internal error in the lower layer, these packets could be dropped and not transmitted on the Link.
- It is required for the upper layer to internally clean up the data path, even if `p1_trdy` is not asserted and it has sampled LinkError on `p1_state_sts` for at least one clock cycle.

The lower layer may enter LinkError state due to Internal LinkError requests such as when:

- Encountering uncorrectable errors due to hardware failure or directed by Upper Layer
- Remote Link partner requests entry into LinkError (RDI only)

The rules for LinkError state are as follows:

- LinkError→Reset: The lower layer transitions to Reset due to an internal request to move to Reset (e.g., reset pin assertion, or software clearing the error status bits that triggered the error) OR (`lp_state_req == Active` and `lp_linkerror = 0`, while `p1_state_sts == LinkError` AND minimum residency requirements are met AND no internal condition such as an error state requires the lower layer to remain in LinkError). Lower Layer must implement a minimum residency time in LinkError of 16 ms to ensure that the remote Link partner will be forced to enter

LinkError due to timeouts (to cover for cases where the LinkError transition happened and sideband was not functional).

10.3.3.8 Common State Rules

This section covers some of the common conditions for exit from Active, Retrain, L1, and L2 to LinkReset, Disable and LinkError states. For RDI, PM encoding and rules correspond to L1 in the text below.

The rules are as follows:

- [Active, Retrain, L1, L2]→LinkReset: The lower layer transitions to LinkReset based on observing `lp_state_req ==LinkReset` or due to an internal request to move to LinkReset or the remote Link partner requesting LinkReset over sideband.
- [Active, Retrain, L1, L2]→Disabled: The lower layer transitions to Disabled based on observing `lp_state_req ==Disabled` or due to an internal request to move to Disabled while `pl_state_sts == Active`, or the remote Link partner requesting Disabled over sideband.
- [Active, Retrain, L1, L2]→LinkError: The lower layer transitions to LinkError based on observing an internal request to move to the LinkError or `lp_linkerror` assertion, or the remote Link partner requesting LinkError over sideband. RDI must move to LinkError before propagating LinkError to all Adapter LSMs.

From a state machine hierarchy perspective, it is required for Adapter LSM to move to LinkReset, Disabled or LinkError before propagating this to CXL vLSMs. This ensures CXL rules are followed where these states are “non-virtual” from the perspective of CXL vLSMs.

Adapter LSM can transition to LinkReset or Disabled without RDI transitioning to these states. In the case of multi-protocol stacks over the same Physical Link/Adapter, each Protocol can independently enter these states without affecting the other protocol stack on the RDI.

If all the Adapter LSMs have moved to a common state of LinkReset/Disabled or LinkError, then RDI is taken to the corresponding state. If however, the Adapter LSMs are in different state combinations of LinkError, Disabled or LinkReset, the RDI is moved to the highest priority state. The priority order from highest to lowest is LinkError, Disabled, LinkReset. For a LinkError/LinkReset/Disabled transition on RDI, Physical Layer must initiate the corresponding sideband handshake to transition remote Link partner to the required state. If no response is received from remote Link partner for this message after 8ms, RDI transitions to LinkError.

If RDI moves to a state that is of a higher priority order than the current Adapter LSM, it is required for the Adapter to propagate that to the Adapter LSM using sideband handshakes to ensure the transition with the remote Link partner.

After transition from LinkError/LinkReset/Disable to Reset on RDI, the Physical Layer must not begin training unless the Physical Layer observes a NOP->Active transition on `lp_state_req` from the Adapter or observes one of the Link Training triggers defined in [Chapter 4.0](#). The Adapter should not trigger NOP->Active unless it receives this transition from the Protocol Layer or has internally decided to bring the Link Up. The Adapter must trigger this on RDI if the Protocol Layer has triggered this even if `pl_inband_pres = 0`. Thus, if the Protocol Layer is waiting for software intervention and wants to hold back the Link from training, it can delay the NOP->Active trigger on FDI. Upper Layers are permitted to transition `lp_state_req` back to NOP after giving the NOP->Active trigger in order to clock gate while waiting for `pl_inband_pres` to assert.

If RDI transitions to L2, the exit is through Reset, and complete Link Initialization and Training flow will occur (including a fresh Parameter Exchange for the Adapter). After transition from L2 to Reset on RDI, the LTSM will begin the Link PM exit and retraining flow when a {LinkMgmt.RDI.Req.Active}

sideband message is received or when the Adapter requests Active on RDI or it observes one of the Link Training triggers defined in [Chapter 4.0](#).

If the Adapter LSM transitions to L2, but RDI does not go to a Link down state (i.e. Reset, LinkReset, Disabled, LinkError), then this is a “virtual” L2 state. The exit from L2 for the Adapter LSM in this case will go through Reset for the Adapter LSM, but it does not result in a fresh Parameter Exchange for the Adapter, and the protocol parameters and the Flit Formats remain the same as prior to L2 entry. An example of this is if there are multiple stacks on the same Adapter, and only one of the FDIs transitions to L2.

IMPLEMENTATION NOTE — LINKRESET/DISABLED

LinkReset and Disabled flows are primarily provided as a means to notify the remote Link partner that the corresponding Protocol Layer intends to trigger the set of actions defined for these by the underlying protocol (e.g., in the case of PCIe and CXL, both of these result in a Conventional Reset on the Upstream Port as defined in the *PCIe Base Specification*). These are typically controlled and co-ordinated through software/firmware. Note that regardless of protocol, there is no hardware mechanism from the PCIe Adapter or Physical Layer to guarantee quiescence or graceful draining of transactions for the LinkReset or Disabled transitions. If this is required by the underlying protocol, it must be handled through software/firmware or other implementation-specific mechanisms outside the PCIe Adapter and Physical Layer.

If the RDI state is already in a Link down state (i.e., Reset, LinkReset, Disabled, LinkError) and the Link is not currently training (Adapter can infer this from `p1_physinrecenter`), then there is no need to notify the remote Link partner. Adapter or Physical Layer can complete the state transitions locally for this case. If RDI is in RESET and the Link is training, it is recommended to wait for training to complete before triggering a state transition with the remote Link partner to LinkReset or Disabled.

The following is written for Disabled state, but applies to both Disabled and LinkReset states.

- For PCIe or CXL protocols, the Downstream Port initiates the transition to Disabled. Because the Upstream Port goes through a Conventional Reset after transitioning to Disabled, the Upstream Port waits for Downstream Port to re-initiate Link Training once the corresponding SoC reset flow has finished.
- For Streaming protocols,
 - The initiating Protocol Layer transitions `lp_state_req` to Disabled. If the necessary conditions are met from the Adapter perspective (for example, attempting to drain the Retry buffer etc.), it forwards the request using the corresponding sideband message to the remote Link partner's Adapter.
 - On the remote Link partner, the Adapter transitions `p1_state_sts` to the requested state once the necessary conditions are met from the Adapter perspective (for example, attempting to drain the Retry buffer etc.). It also sends the corresponding sideband message response.
- If the Adapter needs to take the RDI to Disabled state, it is recommended to keep FDI `p1_state_sts` in Disabled state until that flow has completed. Otherwise, if the exit conditions for Disabled are met, it is permitted to transition to Reset state on FDI.
- Following this, the Protocol Layer on the remote Link partner in turn is permitted bring the FDI state back to Disabled if required by the underlying protocol. The Adapter must not trigger another sideband handshake for this scenario.
- The initiating Adapter transitions `p1_state_sts` to Disabled upon receiving the sideband message response.
- The Protocol Layers on either side of the Link can initiate an exit flow by requesting Active when `p1_state_sts` is Disabled, followed by a NOP->Active transition after the `p1_state_sts` is Reset.
- For configurations in which the Adapter is servicing multiple Protocol Layers, the Disabled or LinkReset handshakes are independent per Protocol Layer. In case the Adapter LSM has transitioned to Reset from Disabled or LinkReset for a given Protocol Layer, the Adapter must keep track of the most-recent previous state to determine the correct resolution for RDI state request.

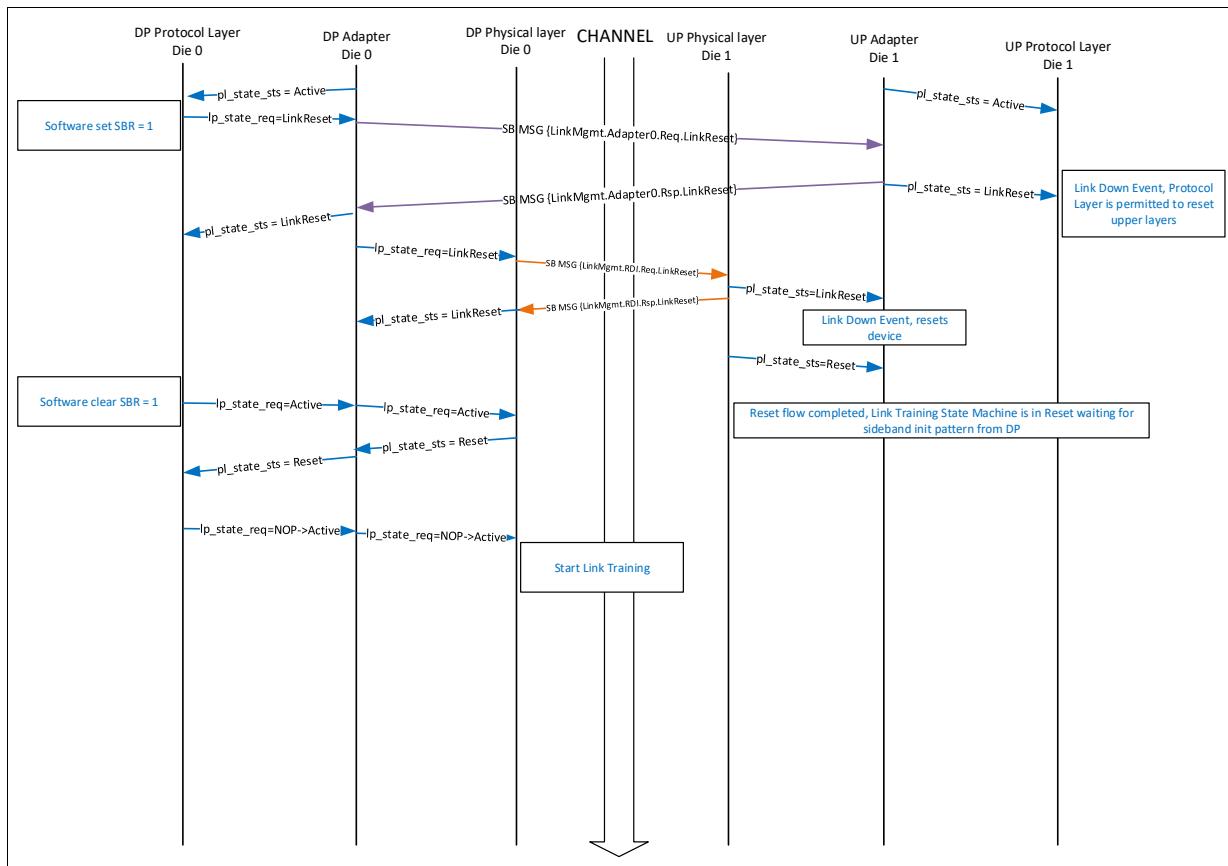
10.3.4 Example Flow Diagrams

10.3.4.1 LinkReset Entry and Exit

Figure 10-30 shows an example flow for LinkReset entry and exit. In the multi-protocol stack scenario, it is permitted for each protocol stack to independently transition to LinkReset. RDI is only transitioned to LinkReset if all the corresponding Adapter LSMs are in LinkReset. For the Link Down Event box, it is expected that SoC does not trigger the overall reset flow until the Physical Layer has completed all the relevant sideband handshakes with the remote Link partner that ensure the LTSMS are also in Reset state.

Figure 10-30 also shows the link reset flow for a PCIe/CXL.io protocol. If Management Transport protocol is supported and negotiated on the same stack as PCIe/CXL.io protocol, the Management Port Gateway must still follow the LinkReset flow and reset requirements that correspond to PCIe/CXL.io.

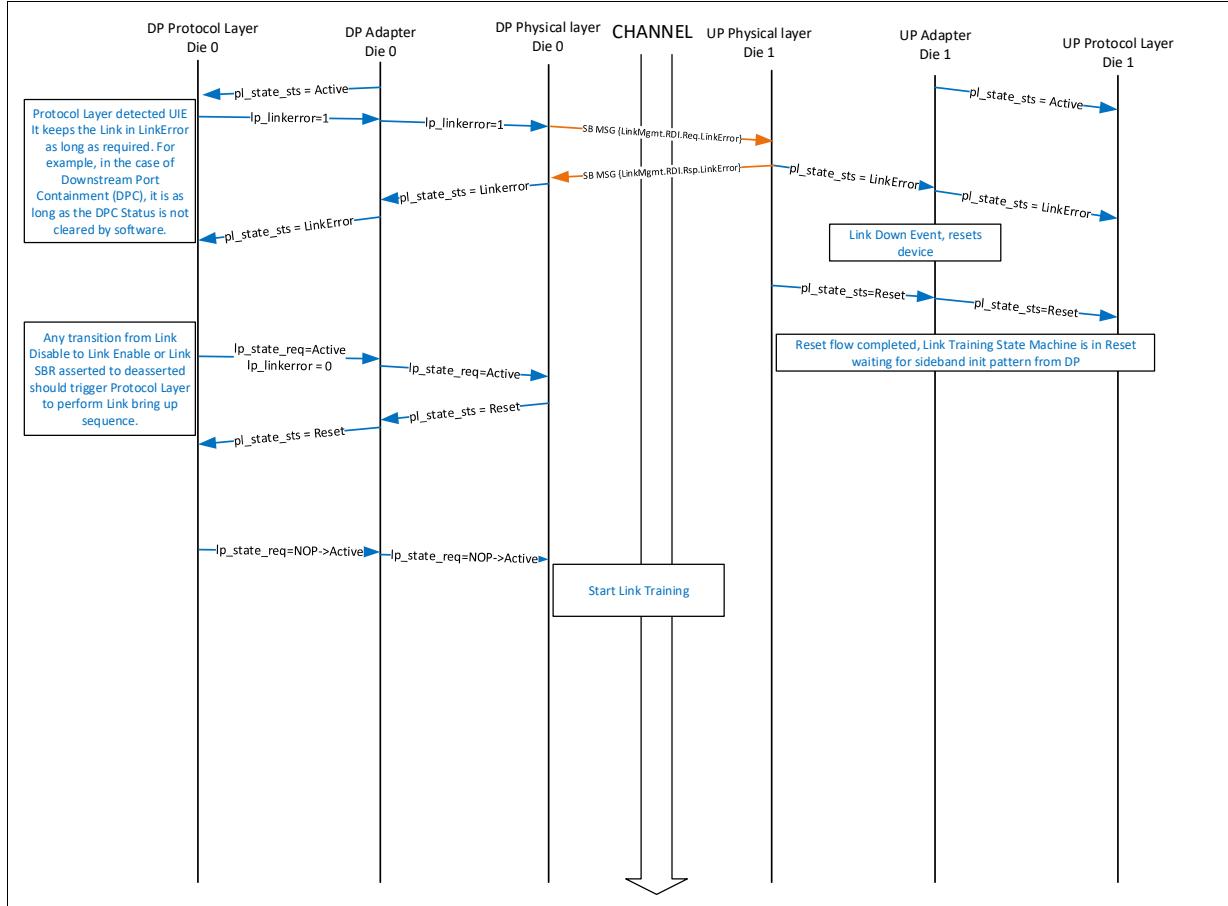
Figure 10-30. LinkReset Example



10.3.4.2 LinkError

Figure 10-31 shows an example of LinkError entry and exit when the Protocol Layer detected an uncorrectable internal error.

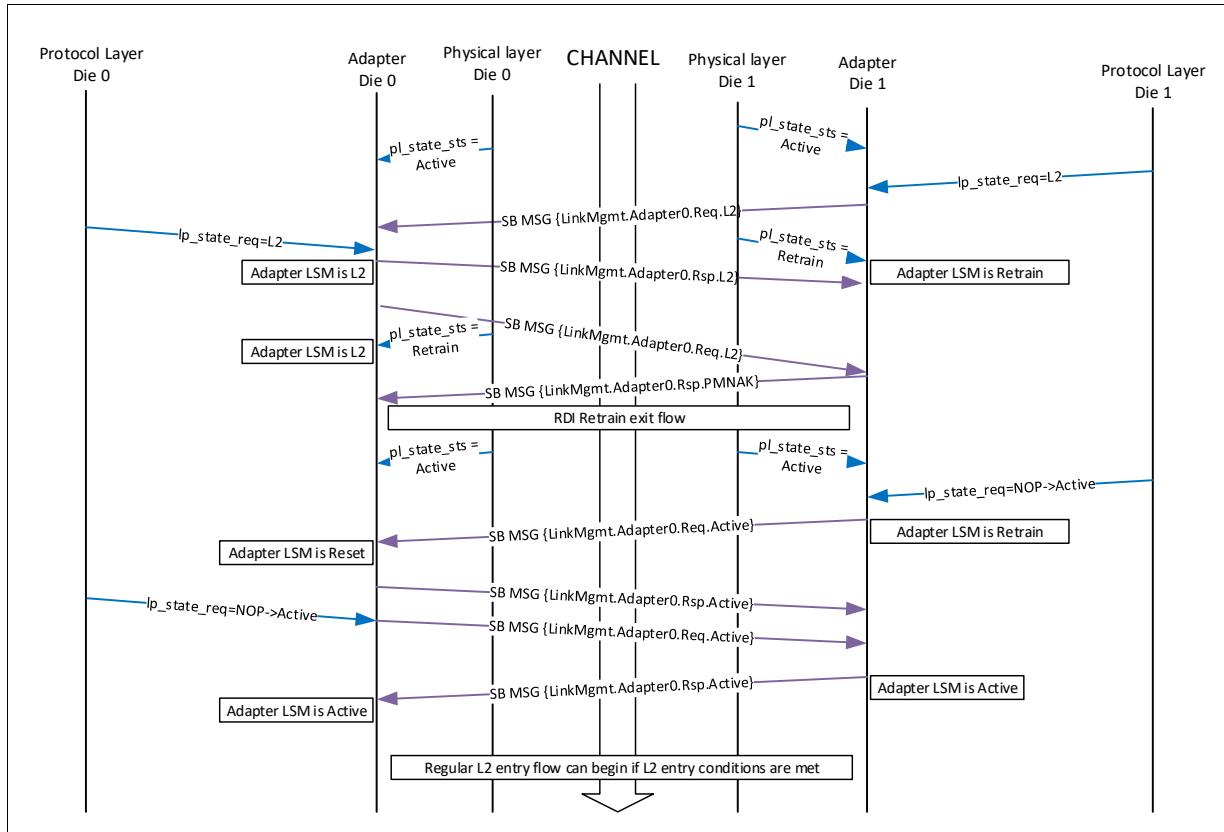
Figure 10-31. LinkError example



10.3.4.3 Example of L2 Cross Product with Retrain on RDI

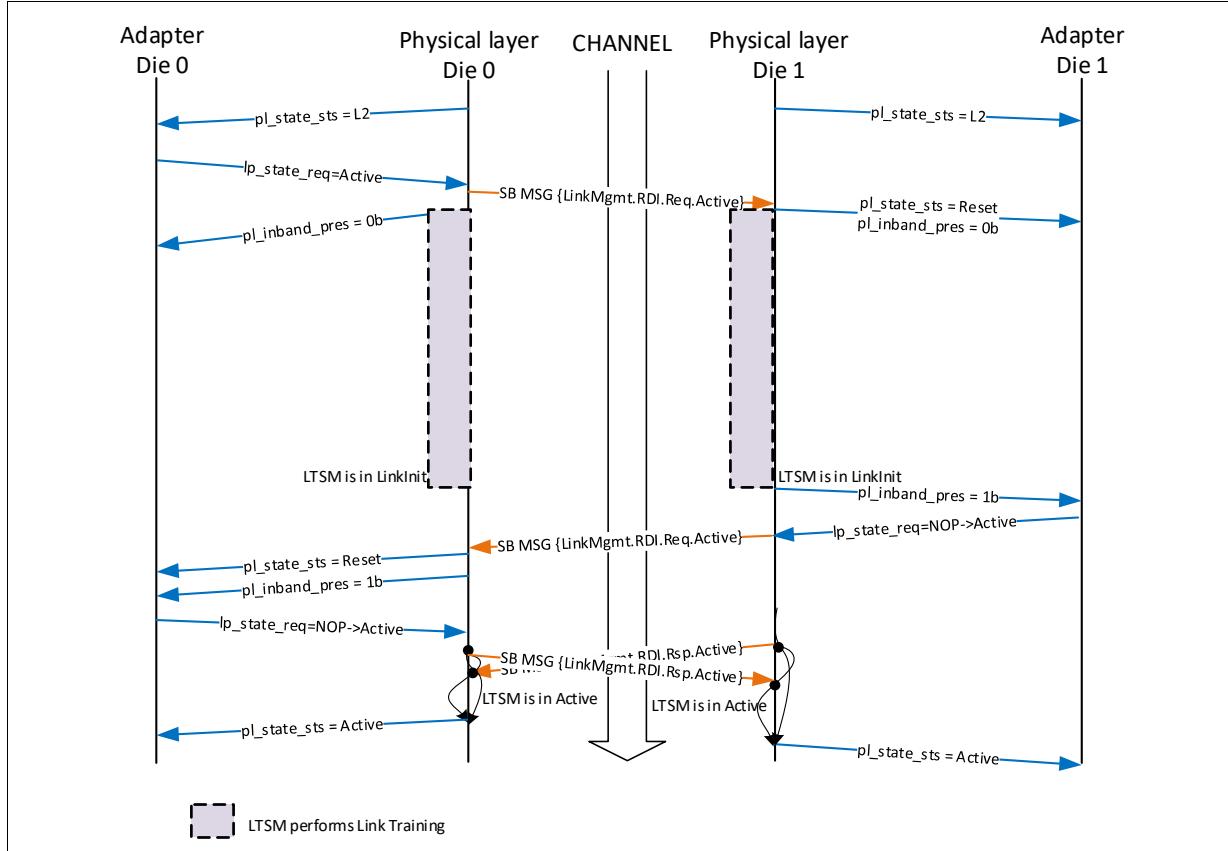
[Figure 10-32](#) shows an example of L2 entry cross product with Retrain transition on RDI (i.e., both flows happen to overlap in a meaningful way) and the corresponding events to resolve the state on either side of the Link.

Figure 10-32. Example of L2 Cross Product with Retrain on RDI



10.3.4.4 L2 Exit Example for RDI

Figure 10-33. L2 Exit Example for RDI



§ §

11.0 Compliance

The goal of Compliance testing is to validate the mainband supported features of a Device Under Test (DUT) against a known good reference UCIe implementation. Device support for Compliance Testing is optional, however a device that does not support capabilities listed in this chapter may not be able to participate in the Compliance program. Different layers of UCIe (Physical, Adapter, Protocol) will be checked independently with a suite of tests for compliance testing.

The system setup for compliance testing is composed of the following:

- Reference UCIe design (Golden Die): This is a known good UCIe implementation across all layers of the UCIe stack.
- DUT: One or more DUTs that will be tested with the reference design. It is required that these have cleared the testing requirements of die sort/pre-bond before they are brought for compliance testing.
- In the case of Advanced Package configuration, a known good silicon bridge or interposer that connects the Golden Die with the DUT. In the case of Standard Package configuration, a known good package for connecting the Golden Die to the DUT.

UCIe implementations that support compliance testing must implement the Compliance/Test Register Block as outlined in [Chapter 9.0](#) and adhere to the requirements outlined in this chapter.

The above components are integrated together in a test package (see [Figure 11-1](#)), which is then used for running Compliance and Interoperability tests.

UCIe sideband plays a critical role for enabling compliance testing by allowing compliance software to access registers from different UCIe components (e.g., Physical Layer, D2D Adapter, etc.) for setting up tests as well as monitoring status. It is expected that UCIe sideband comes up without requiring any FW initialization.

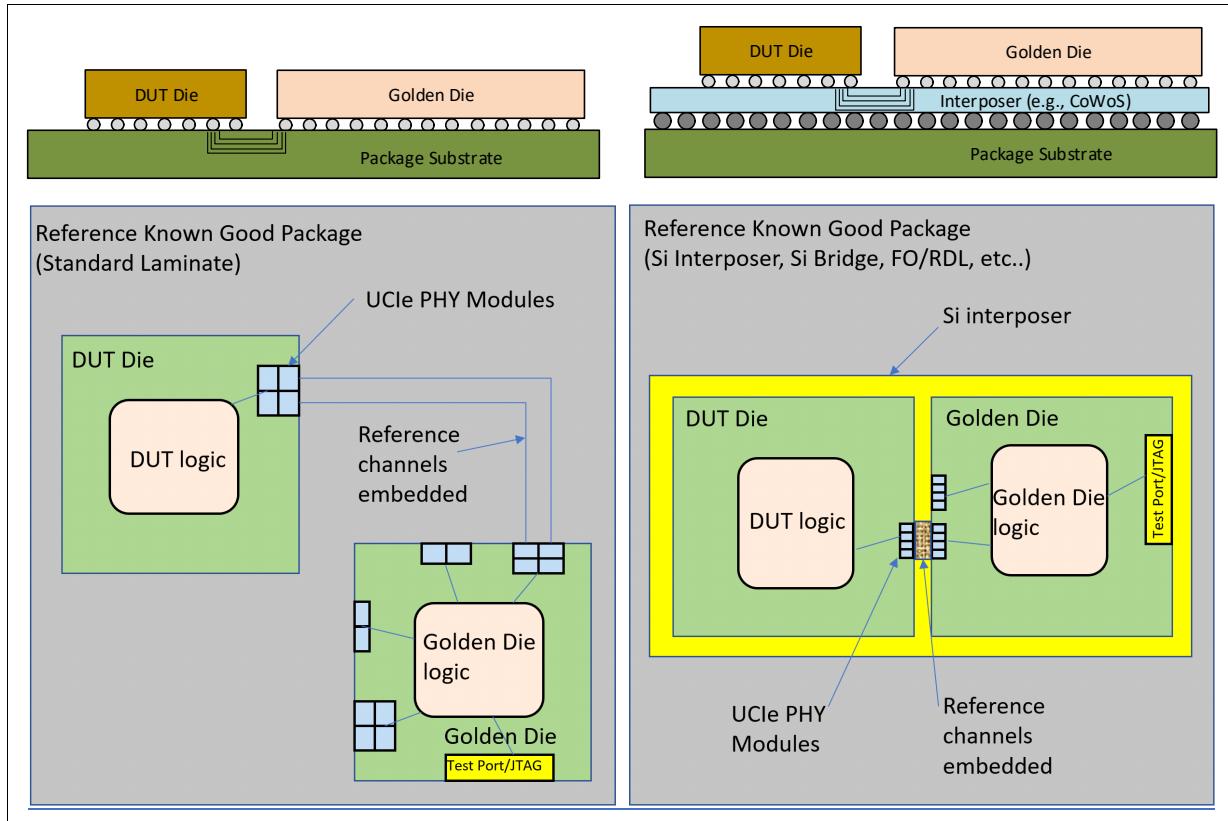
This specification defines the required hardware capabilities of the UCIe stack in the DUT. A separate document will be published later to describe the following:

- Compliance test setup, including the channel model and package level details
- Test details
- Golden Die details including form factor and system-level behavior.

This chapter uses the terms ‘software’ and ‘compliance software’ interchangeably. Any use of the term ‘software’ in this chapter means compliance software that is either running on the Golden Die, or on an external controller that is connected to the Golden Die via test/JTAG port.

Software, prior to testing compliance for any optional UCIe capability, must read the corresponding Capability register (e.g., PHY Capability register described in [Section 9.5.3.22](#)) to ensure that the DUT implements the capability.

Figure 11-1. Examples of Standard and Advanced Package setups for DUT and Golden Die Compliance Testing



11.1 Protocol Layer Compliance

Protocol Layer Compliance testing seeks to test the UCIe protocol stack for compliance to the associated protocol layer specification.

For PCIe and CXL Protocol Layers, UCIe leverages the protocol compliance defined in those specifications for the respective transaction layers. Implementations must follow the requirements and capabilities outlined in *PCIe Base Specification* and *CXL Specification*, respectively.

For Streaming protocols, because Protocol Layer interoperability is specific to the protocol being streamed, compliance testing of the Protocol Layer is beyond the scope of this specification.

11.2 Adapter Compliance

This Specification defines the hardware capabilities that are required in the DUT for exercising and testing the different functionalities of the Adapter. The Golden Die Adapter must have all the capabilities of the DUT, support all the Flit Formats from [Chapter 3.0](#), and must have the capability to inject both consistent and inconsistent sideband messages (for Parameter exchanges, Register accesses and state transitions) to test DUT behavior for various error scenarios (e.g., timeouts, etc.).

The capabilities listed in this section must be supported by the Adapter in the DUT if the Adapter supports any of the Flit Formats defined in [Chapter 3.0](#). These capabilities are applicable to Adapters of all UCIe device types (including Retimers). Each of the capabilities also have their respective

Control and Status registers, which are used to enable software to test various combinations of flows and test criteria.

- Ability to Inject Test or NOP Flits: On the Transmitter, the injection behavior is defined by the Flit Tx Injection Control register (see [Table 9-73](#)). For all injected Flits, CRC is computed, and if CRC error injection is enabled, CRC errors are injected accordingly. It is allowed for the Adapter to be set up to inject NOP Flits or Test Flits. NOP Flit follows the identical layout as defined in [Chapter 3.0](#). Test Flits carry a special encoding of 01b in bits [7:6] of Byte 1 of the Flit Header that is applicable for all Flit Formats that the Adapter supports. Unlike NOP Flits, Test Flits go through the Tx Retry buffer if Retry is enabled. One of the purposes of defining the Test Flits is to test the Retry Flows independently, regardless of whether the Protocol Layer is enabled. The Payload in these Flits carry specific patterns that are determined by the fields in the Flit Tx Injection Control register. Software is permitted to enable flit injection in mission mode as well while interleaving with regular Protocol Flits using the appropriate programming (see the register fields in [Table 9-73](#)). At the Receiver, these Flits are not forwarded to the Protocol Layer. The Receiver cancels these using the `pl_flit_cancel` signal on FDI or any other mechanism; however, CRC must be checked the same as with regular Flits, and any errors must trigger the Retry Flows as applicable.
- Injection of Link State Request or Response sideband messages. This is controlled using the Link State Injection registers defined in the Link State Injection Control Stack 0 and Link State Injection Control Stack 1 registers (see [Table 9-75](#) and [Table 9-76](#), respectively). Single Protocol stack implementations use the Stack 0 register. Software must place the Adapter in Compliance mode (by writing 10b to the 'Compliance Mode' field in the Adapter Compliance Control register).
- Retry injection control as defined in the Retry Injection Control register (see [Table 9-77](#)).

11.3 PHY Compliance

This specification defines the hardware capabilities that are required in the Device Under Test (DUT) for exercising and testing the different functionalities of the Physical Layer. The Golden Die must support capabilities to force timeouts on all applicable sideband messages as well as state residence timers.

The registers and associated functionality defined in [Section 9.5.4](#) and the UHM DVSEC Capability defined in [Section 9.5.3.36](#) are used for Compliance testing. These registers provide the following functionality:

- Timing margining
- Voltage margining, when supported
- BER measurement
- Lane-to-Lane skew for a given module at both the Receiver and Transmitter
- TX Equalization (EQ) as defined in [Section 5.3.3](#)

§ §

Appendix A CXL/PCIe Register applicability to UCIe

A.1 CXL Registers applicability to UCIe

All CXL-defined DVSECs fully apply in the context of UCIe when operating in Raw Format. When operating in non-Raw Format, a few register definitions need to be reinterpreted in the context of UCIe. See below for details. Note that regardless of the Raw Format or non-Raw Format, device/port configurations with CXL 1.1 compliance is not permitted as was discussed in [Chapter 9.0](#).

Table A-1. CXL Registers for UCIe devices

Register Block	Register	Bits	Comments
DVSEC Capability	DVSEC Flex Bus Port Control	3,4	See next row for how these bits are handled
	DVSEC Flex Bus Port Status	3, 4	This bit just mirrors bits 3, 4 in Flex bus port control register, to mimic legacy behavior.
		11, 12	Hardwired to 0
	From 14h-1Fh		N/A

A.2 PCIe Register applicability to UCIe

All PCIe specifications defined DVSEC apply in the context of UCIe as well. There are a few Link and PHY layer registers/bits though that are N/A or need to be reinterpreted in the context of UCIe. They are listed below.

Table A-2. PCIe Registers for UCIe devices

Register Block	Register	Bits	Comments
PCIe capability	PCI Express Capabilities Register	8	Slot implemented – set to 0. And hence follow rules for implementing other slot related registers/bits at various locations in the PCIe capability register set.
	Device capabilities Register	8:6	N/A and can be set to any value
	Link Capabilities Register	3:0	Max Link Speed: Set to 0011b indicating 8GT/s
		9:4	Max Link Width: 01 0000b, indicating x16
		11:10	ASPM support: 01b/11b encodings disallowed
		14:12	N/A
		17:15	L1 Exit Latency: Devices/Ports must set this bit based on whether they are connected to a retimer or not, and also the retimer based exit latency might not be known at design time as well. To assist with this, these bits need to be made HWInit from a device/port perspective so system FW can set this at boot time based on the specific retimer based latencies.
		18	N/A and hardwired to 0
	Link Control Register	6	HW ignores what is written here but follow any base spec rules for bit attributes.
		7	HW ignores what is written here but follow any base spec rules for bit attributes.
		8	Set to RO 0
		9	Set to RO 0
		10, 11, 12	HW ignores what is written in these bits but follows any base spec rules for bit attributes.
	Link Status Register	3:0	Current Link speed: Set to 0011b indicating 8GT/s
		9:4	Negotiated Link width: x16
		15	Hardwired to 0
	Link Capabilities 2 Register	7:1	Set to 000 0111b
		15:9	Set to 00h
PCIe Capability	Link Capabilities 2 Register	22:16	Set to 00h
		24:23	Set to 00b just to appear compliant
	Link Control 2 Register	3:0	Target Link speed: Writes to this register are ignored by UCIe hardware, but HW follows the base spec rules for bit attributes
		4	HW ignores what is written in this bit but follows any base spec rules for bit attributes.
		5	HW autonomous speed disable – Set to RO 0
		15:6	N/A for UCIe. HW should follow base spec rules for register bit attributes.
	Link Status 2 Register	9:0	Set to RO 0
PCIe Extended Capability	Secondary PCI Express Extended Capability	All	Implement per the base spec, but HW ignores all commands from SW and also sets all equalization control registry entries to 0.

A.3 PCIe/CXL registers that need to be part of D2D

- PCIe Link Control Register
 - Bits 13, 5, 4, and 1:0 are relevant for D2D operation
- CXL DVSEC Flex Bus Port Received Modified TS Data Phase1 Register
- CXL DVSEC Flex Bus Port Control
- CXL DVSEC Flex Bus Port Status
- CXL ARB/MUX registers

§ §

Appendix B AIB Interoperability

Implementations are permitted to design a superset stack to be interoperable with UCIe/AIB PHY. This section details the UCIe interoperability criteria with AIB.

B.1 AIB Signal Mapping

B.1.1 Data path

Data path signal mapping for AIB 2.0 and AIB 1.0 are shown in [Table B-1](#) and [Table B-2](#) respectively. AIB sideband is sent over an asynchronous path on UCIe main band.

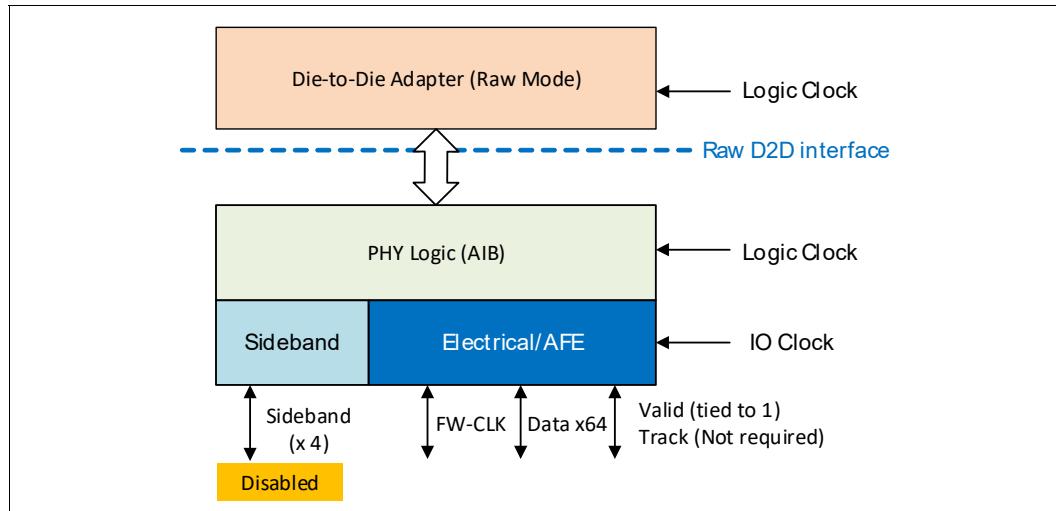
B.1.2 Always high Valid

Always high Valid is an optional feature that is only applicable to AIB interoperability applications. This must be negotiated prior to main band Link training through parameter exchange. Raw mode must be used in such applications.

B.1.3 Sideband

AIB sideband is sent using UCIe main band signals. UCIe sideband is not required in AIB interoperability mode and it is disabled (Transmitters are Hi-Z and Receivers are disabled).

Figure B-1. AIB interoperability



B.1.4 Raw Die-to-Die interface

AIB Phy logic block shown in Figure B-1 presents a subset of RDI to next layer up.

Note: More details will be shown in a later revision of this specification

Table B-1. AIB 2.0 Datapath mapping for Advanced Package

UCIE Interface	AIB 2.0	Note
TXDATA[39:0]	TX[39:0]	
TXDATA[47:40]	AIB Sideband Tx	Asynchronous path
TXDATA[63:48]	N/A	Disabled (Hi-Z)
RXDATA[39:0]	RX[39:0]	
RXDATA[47:40]	AIB Sideband Rx	Asynchronous path
RXDATA[63:48]	N/A	
TXDATASB	N/A	Disabled (Hi-Z)
RXDATASB		
TXCKSB		
RXCKSB		
TXDATASBRD		
RXDATASBRD		

Table B-2. AIB 1.0 Datapath mapping for Advanced Package

UCIE Interface	AIB 1.0	Note
TXDATA[19:0]	TX[19:0]	
TXDATA[42:20]	AIB Sideband Tx	Asynchronous path
TXDATA[63:43]	N/A	Disabled (Hi-Z)
RXDATA[19:0]	RX[19:0]	
RXDATA[42:20]	AIB Sideband Rx	Asynchronous path
RXDATA[63:43]	N/A	
TXDATASB	N/A	Disabled (Hi-Z)
RXDATASB		
TXCKSB		
RXCKSB		
TXDATASBRD		
RXDATASBRD		

B.2 Initialization

AIB Phy logic block shown in Figure B-1 contains all the AIB Link logic and state machines. Please see AIB specification (Section 2 and Section 3) for initialization flow.

B.3 Bump Map

Note: More details will be shown in a future revision this specification

§ §