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 UCIe 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 UCIe 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 **`pl_phyinrecenter`**), 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 **`pl_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 **`pl_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 **`pl_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 **`pl_state_sts`** is Disabled, followed by a NOP->Active transition after the **`pl_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.

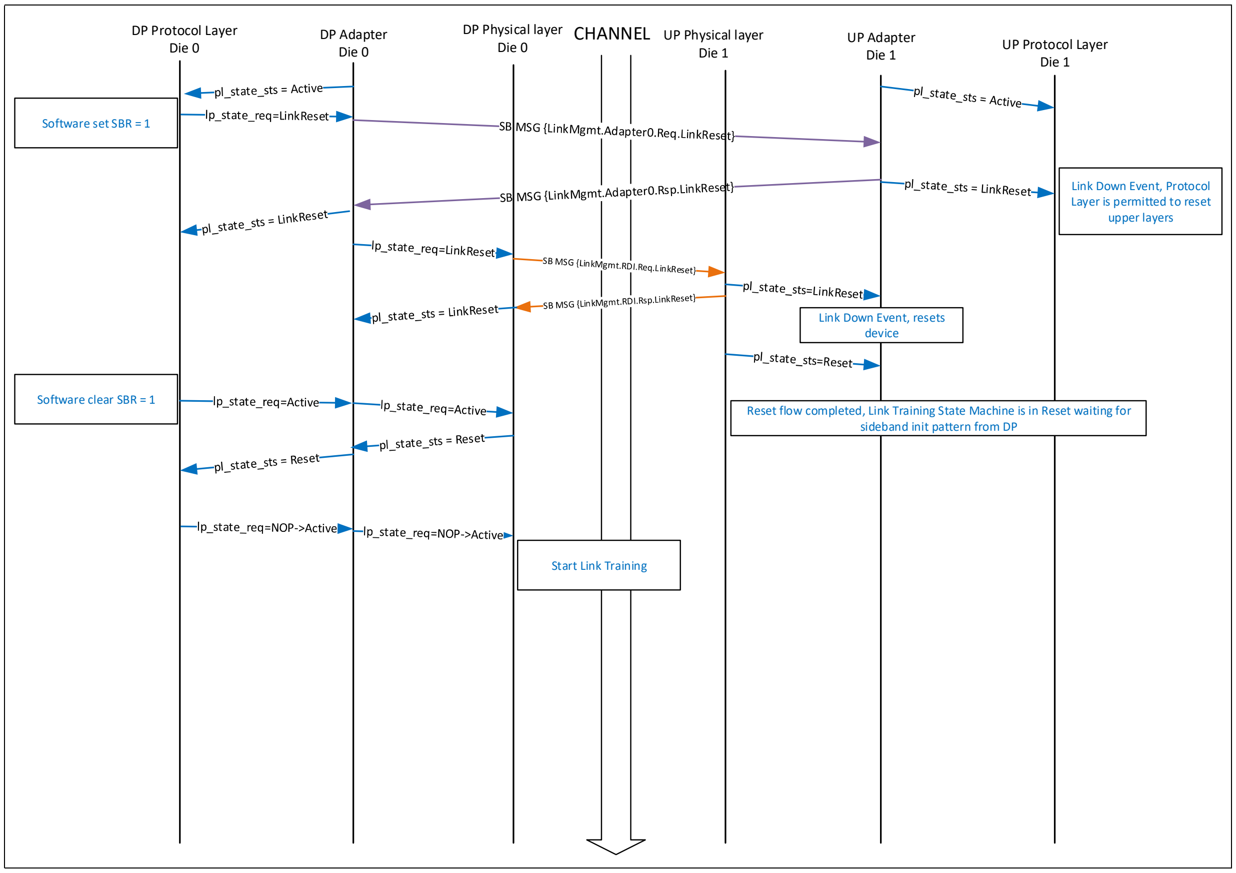
10.3.4 Example Flow Diagrams

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 LTSM is 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.

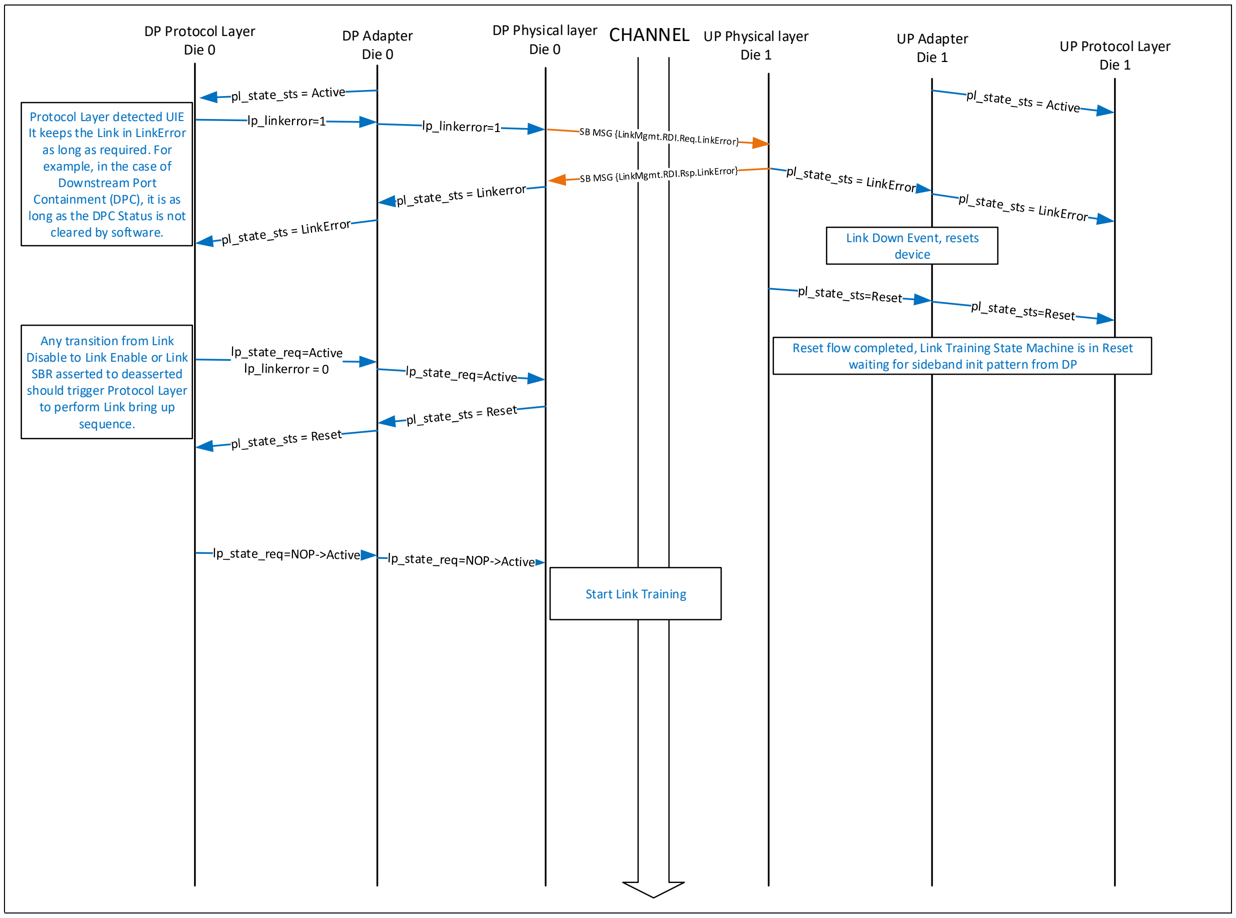
Figure 10-30. LinkReset Example



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.

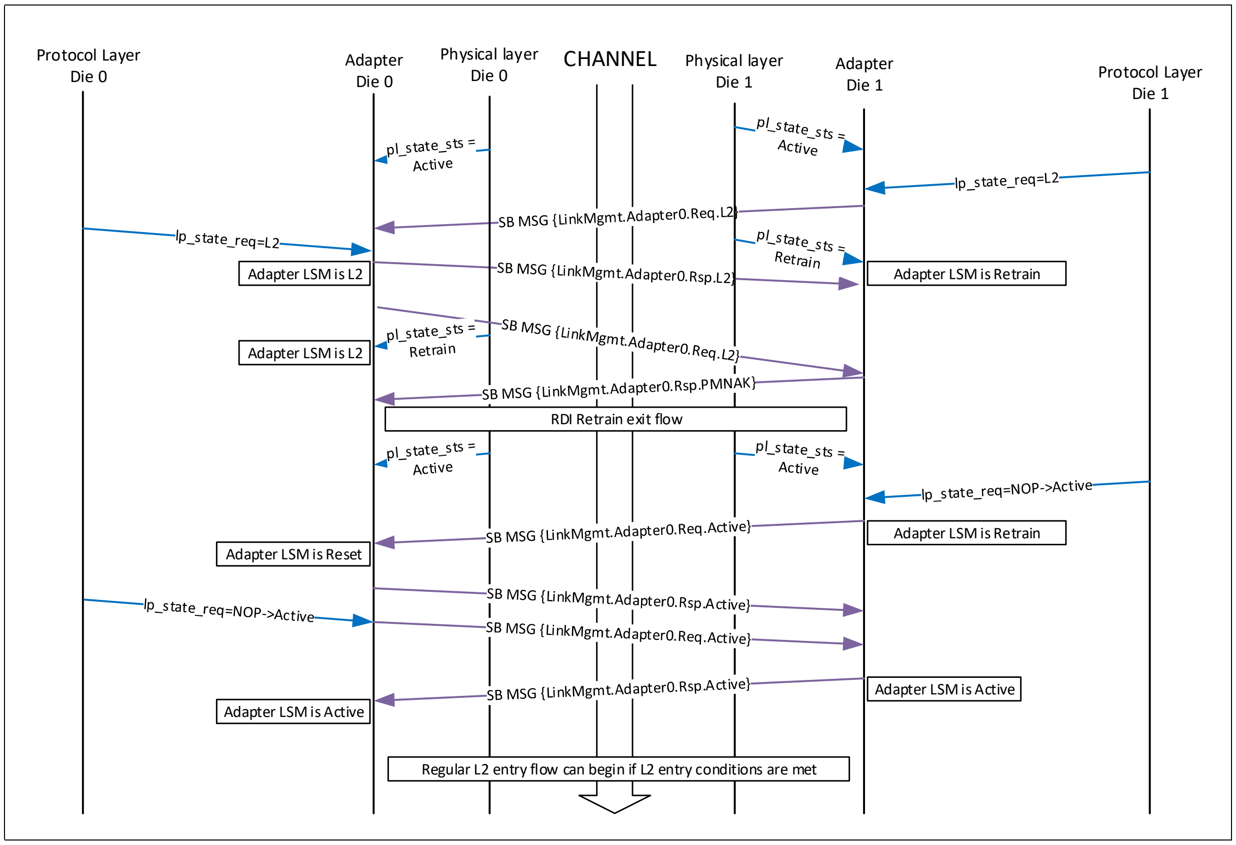
Figure 10-31. LinkError example


DP Protocol Layer Die 0
DP Adapter Die 0
DP Physical layer Die 0
CHANNEL
UP Physical layer Die 1
UP Adapter Die 1
UP Protocol Layer Die 1
pl_state_sts = Active
pl_state_sts = Active
Protocol Layer detected UIE It keeps the Link in LinkError as long as required. For example, in the case of Downstream Port Containment (DPC), it is as long as the DPC Status is not cleared by software.
lp_linkerror=1
lp_linkerror=1
SB MSG {LinkMgmt.RDI.Req.LinkError}
SB MSG {LinkMgmt.RDI.Rsp.LinkError}
pl_state_sts = LinkError
pl_state_sts = LinkError
pl_state_sts = Linkerror
pl_state_sts = LinkError
Link Down Event, resets device
pl_state_sts=Reset
pl_state_sts=Reset
Reset flow completed, Link Training State Machine is in Reset waiting for sideband init pattern from DP
Any transition from Link Disable to Link Enable or Link SBR asserted to deasserted should trigger Protocol Layer to perform Link bring up sequence.
lp_state_req=Active
lp_linkerror = 0
lp_state_req=Active
pl_state_sts = Reset
pl_state_sts = Reset
lp_state_req=NOP->Active
lp_state_req=NOP->Active
Start Link Training

10.3.4.3 Example of L2 Cross Product with Retrain on RDI

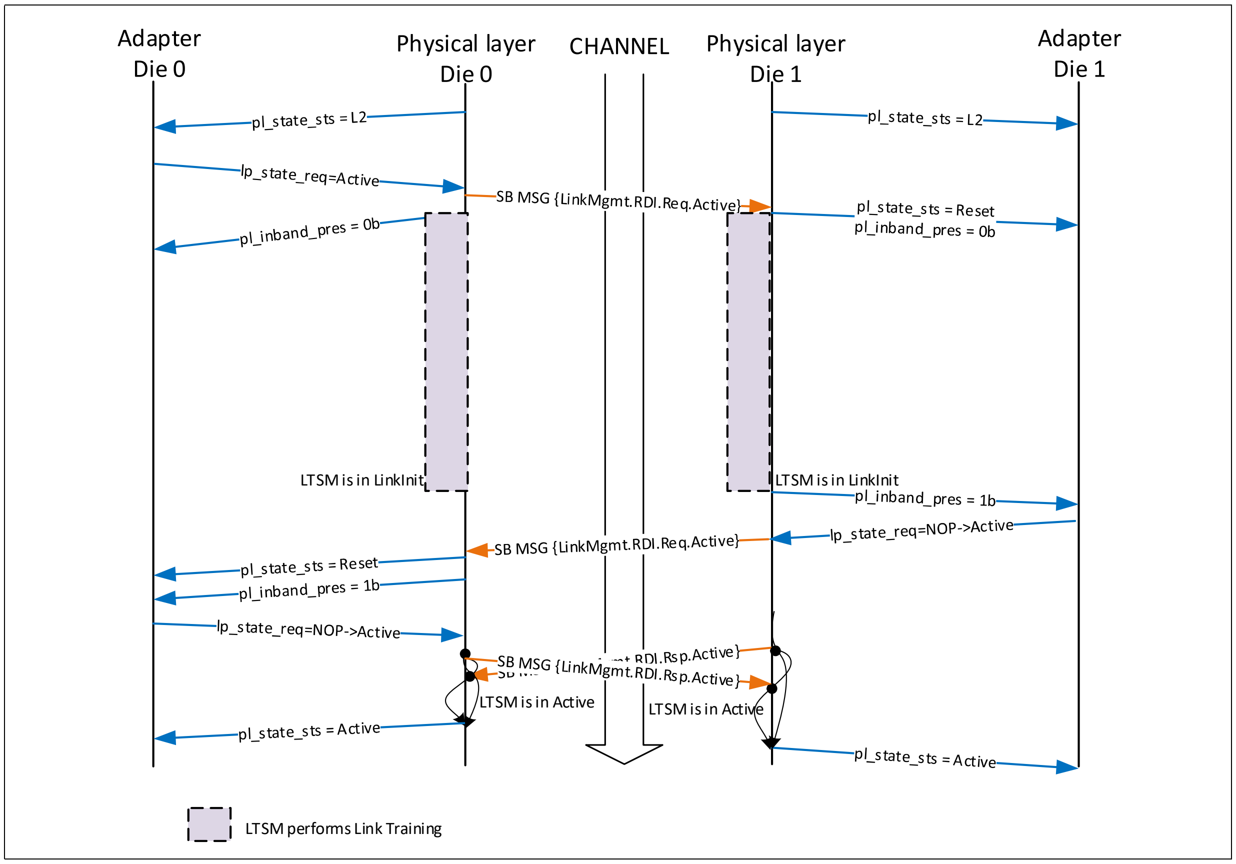
Figure 10-32 shows an example of L2 entry cross product with Retrain transition on RDI (i.e., both flows happen to overlap in a meaningful way) and the corresponding events to resolve the state on either side of the Link.

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.

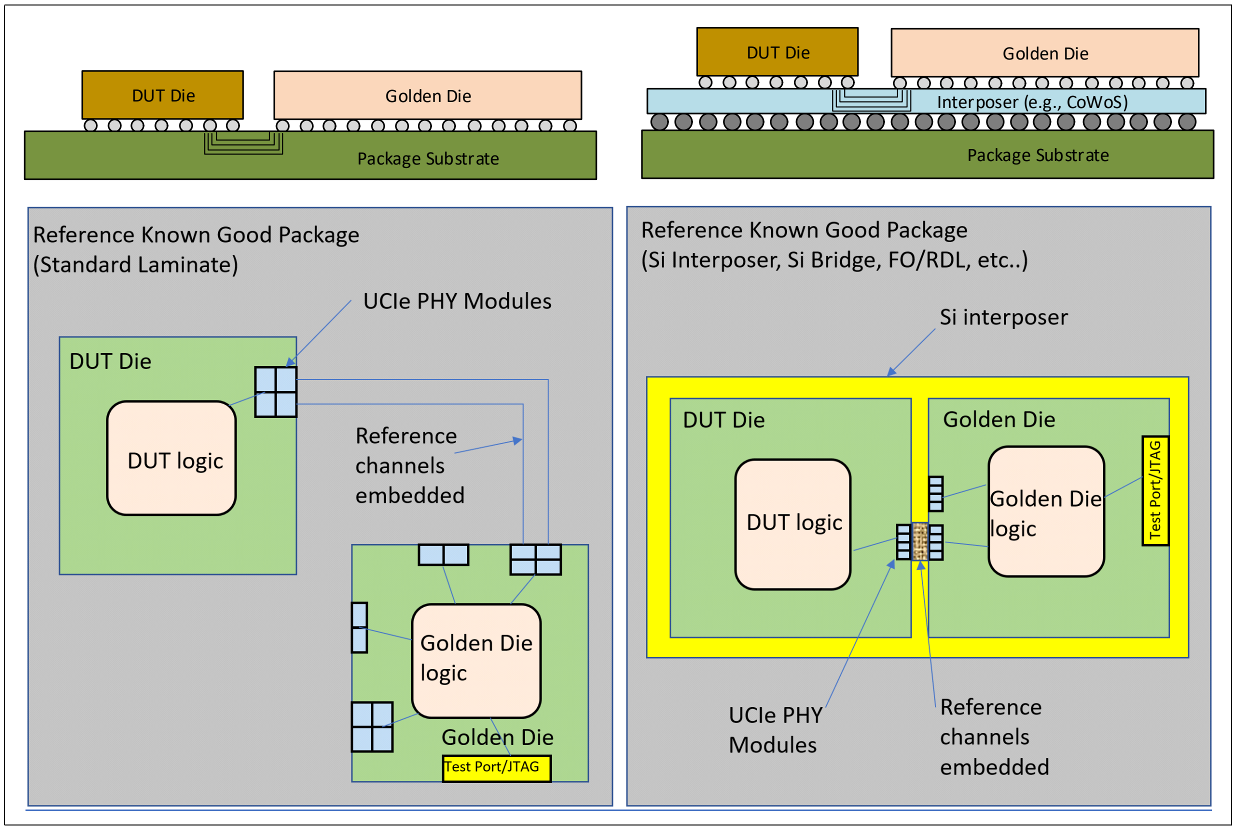
This specification defines the required hardware capabilities of the UCIe stack in the DUT. A separate document will be published later to describe the following:

- 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.

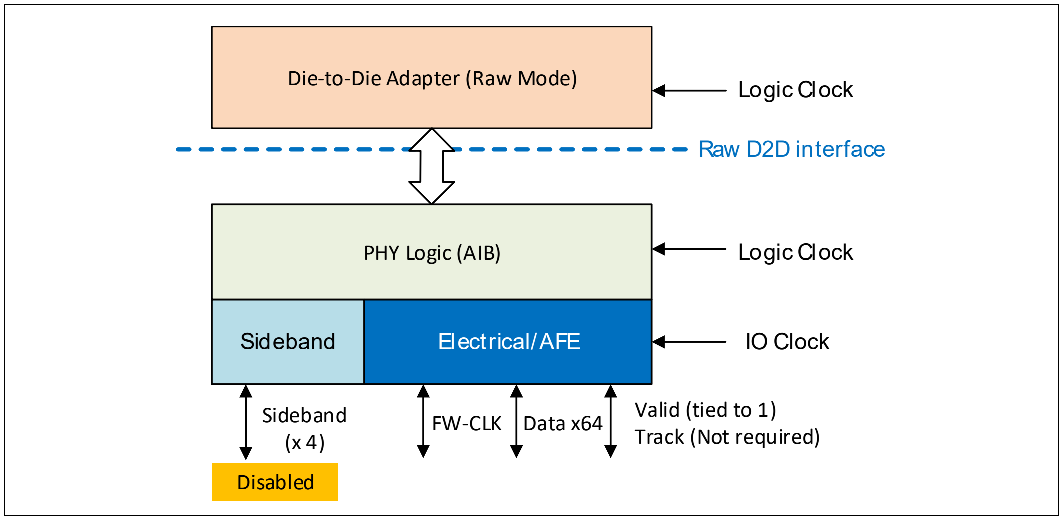
B.1.2 Always high Valid

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

§ §