CXL ENGINEERING CHANGE NOTICE

TITLE:	Link IDE Switch Passthrough
DATE:	Introduced date (05/23/2025) Updated date (03/11/2026)
AFFECTED DOCUMENT:	CXL Specification Revision 4.0 Version 1.0
SPONSOR:	Derek Rohde, NVIDIA

Part I

1. **Summary of Functional Changes**

CXL encryption is currently limited to Link IDE, where flits are encrypted and decrypted on each hop along a path, including at switch ports. There is no end-to-end encryption scheme in CXL, and achieving one may take significant time and effort. CXL TSP architecture does not allow for switches to be included in the TCB, as doing so is not in alignment with many confidential computing architectures.

Link IDE Switch Passthrough (LISP) provides a mechanism to bridge the current gap between the lack of support for switches in TSP topologies and the lack of E2E encryption. It does so in limited topologies (a single CXL host and device with peer PCIe devices) by enabling Link IDE on the CXL links and creating a passthrough path in the switch for CXL.cachemem flits. The CXL flits are not decrypted by the switch, thus the switch does not need to be included in the TCB.

The major changes required or recommended to implement the ECN are as follows:

- Creation of a direct port-to-port CXL.cachemem flit forwarding path through a switch, including LLCTRL messages, which bypasses the link and transaction layers in the CXL port
- An optional but strongly recommended crediting mechanism to limit the flow of CXL.cachemem flits to manage congestion in CXL switch ports. This requires dedicated receive buffering in the switch for CXL.cachemem flits.
- Optionally but strongly recommended, increase credited receive buffer sizes in the host and device to account for increased link latency in the LISP path in the switch

LISP is supported with the following constraints:

- Only one CXL Type 1, 2, or 3 device may be included in a LISP VH
- Only 256B flit mode is supported
- PBR is not supported
- MLDs are not supported

2. **Benefits as a Result of the Changes**

LISP enables more flexible topologies, especially for CXL accelerators, not currently supported in the TSP architecture.

3. **Assessment of the Impact**

To implement LISP, switches must support the flit passthrough mechanism.

Additionally, it is recommended that switches, Root Ports, and device ports which intend to optimally support LISP also support the new CXL.cachemem crediting mechanism for improved performance.

There is no impact to backwards compatibility nor to existing systems, as LISP is optional. It is expected that existing Root Ports and device ports can be used in a LISP system function properly.

4. **Analysis of the Hardware Implications**

Impact to switches

- Support the flit passthrough path and forwarding of entire flits between one CXL USP and one DSP
- Optionally support generation of new ALMP messages used to issue credits to link partners to allow transmission of CXL.cachemem flits

Impact to Root Ports and device ports

- Optionally support reception of new ALMP messages with CXL.cachemem flow control credits sent by LISP switch ports and associated logic to stop transmission of those flits when credit is exhausted. Strongly recommended to avoid exceeding the PCIe recommendations for flit to ACK latency and to avoid blocking CXL.io traffic due to CXL.cachemem congestion.
- Optionally increase message flow control buffering advertised through the CRD field to account for the increased round trip time added by the switch

5. **Analysis of the Software Implications**

Software boot flow must be modified to detect that LISP may be enabled in a VH and to enable it when supported and desired. This includes setting the appropriate configuration fields and involves invoking a link retrain.

6. **Analysis of the Compliance and Test Implications**

New tests will be required to verify compliance of the LISP capabilities in Root Ports, device ports, and switches that support LISP.

Part II
Detailed Description of the change

Change Table 1-1 in Section 1.2, as follows:

Link Layer Clock	CXL.cachemem link layer clock rate usually defined by the flit rate during normal operation.
LISP	Link IDE Switch Passthrough, a mode of operation that allows a single CXL device to be attached via a LISP-capable switch to a host with the entire path between them encrypted and the switch not included in the TCB.
LLC	Last Level Cache

Add subsection 4.3.9 to Chapter 4

4.3.9 CXL Link IDE Switch Passthrough

See section 11.3.9.1.3 for impacts to the Link layer when operating in Link IDE Switch Passthrough mode.

Update Table 5-5 with the indicated change

Table 5-5. ALMP Byte 1 Encoding

Byte 1 Bits	Description
7:0	**Message Encoding** • 0000 0001b = L0p Negotiation ALMP (for 256B Flit mode; reserved for 68B Flit mode) • 0000 1000b = vLSM ALMP is encoded in Bytes 2 and 3 • 0000 1001b = Link IDE Switch Passthrough (LISP) flit credit update. Credit count update is encoded in Byte 2. See section 11.3.9.1 for details regarding LISP operation. • All other encodings are reserved

Add the following table after Table 5-7

Table 5-8. ALMP Byte 2 Encodings for LISP Credit Update

Byte 2 Bits	Description
8:0	**LISP Flit Credit Update Count** Contains the number of CXL.cachemem flit credits to update. Maximum value is the smaller of FFh or the value programmed in **Max Initial LISP Flit Credits** in the **ARB/MUX LISP Status and Control Register**.

Add subsection 5.2.2 to Chapter 5

5.2.2 CXL Link IDE Switch Passthrough

See section 11.3.9.1.2 for impacts to the ARB/MUX when operating in Link IDE Switch Passthrough mode and for specification of ALMP LIPS Flit Credit operation.

insert subsection 8.2.4.17 to Chapter 8

8.2.4.17 CXL LISP Capability Header

Bit Location	Attribute	Description
15:00	RO	CXL_Capability_ID: This defines the nature and format of the CXL Capability register. For the CXL LISP Header register, this field shall be 0011h.
19:16	RO	CXL_Capability_Version: This defines the version number of CXL Capability structure present. For this version of the specification, this field must be 1h.
31:20:00	RO	CXL LISP Capability Pointer: This defines the offset of the CXL LISP Capability structure relative to the beginning of the CXL Capability Header register. Details in Section 8.2.4.32

Update table 8-74 with the following change

Table 8-74. CXL_Capability_ID Assignment (Sheet 2 of 2)

Capability	ID	Highest Version	Mandatory[1]	Not Permitted[1]	Optional[1]
CXL Cache ID Decoder Capability (Section 8.2.4.29)	0Eh	1		P, D1, D2, LD, FMLD, UP1, DP1, R, USP, vUSP, RC, SLD-B	R, DSP, vDSP
CXL Extended HDM Decoder Capability (Section 8.2.4.30)	0Fh	3		All others	RC, USP, vUSP
CXL Extended Metadata Capability (Section 8.2.4.31)	10h	1		All others	CXL.mem- capable LD, D2, or one SLD-B per BPD that supports 256B Flit mode
CXL LISP Capability (Section 8.2.4.32)	11h	1		All others	USP, DSP

Add the following subsection to chapter 8

8.2.4.32 CXL LISP Capability Structure

The CXL LISP Capability structure is used to configure switch ports for LISP.

See Section 11.3.9.1 for details regarding LISP operation.

Table 8-170. CXL LISP Capability Structure

Offset	Register Name
00h	CXL LISP Capability Register
04h	CXL LISP Control Register

8.2.4.32.1 CXL LISP Capability Register (Offset 00h)

Table 8-171. CXL LISP Capability Register (Offset 00h)

Bit Location	Attributes	Description
0	HwInit	**LISP Bypass Path Supported**: If set, indicates support for the LISP mode bypass path in the switch.
31:1	RsvdP	**Reserved**

8.2.4.32.2 CXL LISP Control Register (Offset 04h)

Table 8-172. CXL LISP Control Register (Offset 04h)

Bit Location	Attributes	Description
0	RWS	**LISP Bypass Path Enable:** When set, the port is configured to operate in LISP mode. Only applicable if 256B flit mode is negotiated during link training.
7:1	RsvdP	**Reserved**

15:8	RWS	**Bypass Path Port Partner**: Must be configured with the Port Number (see the PCIe Base Specification) of the USP or DSP within the switch this port is forwarding CXL.cachemem flits to on the LISP bypass path. Only applicable if 256B flit mode is negotiated during link training.
31:16	RsvdP	**Reserved**

Add the following register to section 8.2.5

8.2.5.6 ARB/MUX LISP Status and Control Register (Offset 200h)

Table 8-176. ARB/MUX LISP Status and Control Register (Offset 200h)

Bit	Attributes	Description
0	HwInit	**LISP Credit Update ALMP Format Supported**: If set, indicates support LISP Mode Credit Update ALMPs • Switch ports: the switch port supports transmission of LISP Credit Update ALMPs • Root Ports and device ports: the port supports reception of LISP Credit Update ALMPs
1	RWS	**LISP Credit Update ALMP Enable**: When set, the Arb/Mux behaves as follows: • Switch ports: The Arb/Mux shall transmit LISP Credit Update ALMP messages to its link partner to update CXL.cachemem credit • Root Ports and device ports: The Arb/Mux shall update a credit counter with the indicated credit when a LISP Credit Update ALMP message is received Only applicable if 256B flit mode is negotiated during link training.
3:2	RsvdP	**Reserved**
11:4	HwInit/RWS	**Max LISP Flit Credit** • For Root Ports or device ports which have the **LISP Credit Update ALMP Format Supported** bit set, this field indicates the maximum initial credits the switch may send to the attached port after link up.
23:16	RWS	**Max Initial LISP Flit Credits**: In switch ports which are enabled to transmit LISP Credit Update ALMPs, this field is programmed with the value read by software in the link partner's **Max LISP Flit Credits Supported** field.
31:24	RsvdP	**Reserved**

Add the following section to chapter 9

9.21 Link IDE Switch Passthrough

Link IDE Switch Passthrough, described in **Section 11.3.9.1**, requires modification to the software configuration flow. During normal initialization a Root Port and device port exchange flow control credits directly with their link partners. When operating in LISP mode, message flow control credits must be exchanged and updated between the Root Port and device port, rather than with the switch port link partners they are directly connected to. The modified flow facilitates the proper credit exchange by configuring the LISP path and then initiating a Secondary Bus Reset. The high-level modification to the configuration flow follows:

- After link training and enumeration, software configures the ports as usual
- Software determines the path between the Root Port and device port is LISP capable, and performs LISP configuration
  - Enable LISP bypass on the switch ports and configure the LISP bypass partner ports
  - If LISP flit credit update ALMP format is supported by link partners, enable the feature
  - Set the LL_Init_Stall and LL_Crd_Stall bits in the **CXL Link Layer Control and Status Register** in each port in the LISP path
- To reset the initialized credits, software retrains the links by writing the

Secondary Bus Reset bit of the Bridge Control Register in the switch DSP and the Root Port.

- After both links in the LISP path have retrained, software clears the LL_Init_Stall and LL_Crd_Stall bits in the Root Port and device port. Credit exchange between the Root Port and the device port commences.
    - Software may choose to set the LL_Init_Stall and LL_Crd_Stall bits again so that they are preconfigured for any subsequent reset events.

When Link IDE is configured by Software, it is only configured in the Root Port and the device ports as if the two are directly connected on a link. The switch ports are not configured for CXL.cachemem Link IDE as this functionality is bypassed in the switch.

Add the following subsection to chapter 11, section 11.3.9

11.3.9.1 Link IDE Switch Passthrough

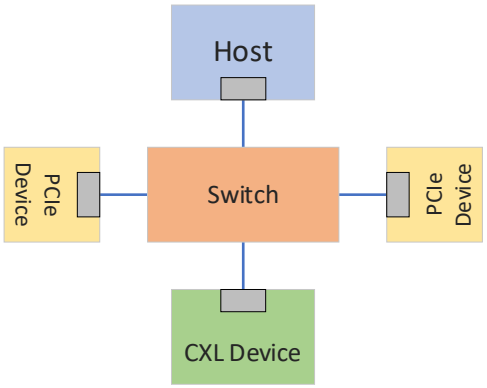
Many deployment security models do not allow for switches to be included in the Trusted Computing Boundary. As such, enabling Link IDE on upstream and downstream switch links, with the switch able to decrypt CXL.cachemem Link IDE flits and see their contents in the clear is an unacceptable threat in these cases. Without an end-to-end encryption mechanism for CXL.cachemem, this severely limits deployable topologies when encryption is required.

Link IDE Switch Passthrough (LISP) enables switches to be deployed in constrained topologies. The following limitations apply to deployment of LISP mode:

- Only one CXL Type 1, 2, or 3 device may be included in a VH
- Only 256B flit mode is supported
- PBR is not supported
- MLDs are not supported

LISP enables topologies where a single CXL.cachemem device is deployed in a VH, coupled with peer PCIe devices. LISP enables the CXL device to communicate securely with the host using CXL.cachemem and with the peer PCIe devices using CXL.io through the switch.

Figure 11-22. Example LISP mode topology



When a switch has been configured to operate in LISP mode, its behavior for handling CXL.cachemem flits is changed. Rather than decrypt received flits, unpack embedded messages, then generate new flits on the switch's target port, flits are forwarded through the switch intact (excluding the Flit Header, CRC, and FEC). When in LISP mode, the switch creates an internal direct connection between the USP connected to

the Host and the DSP connected to the CXL Device.

Figure 11-23. LISP Bypass path relative to CXL layers

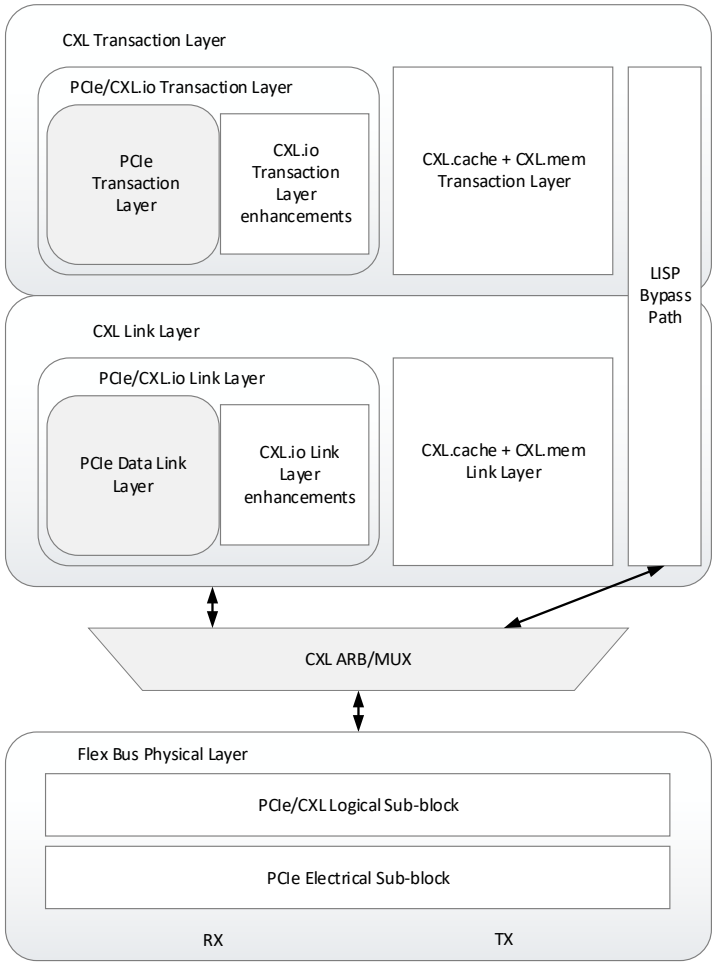
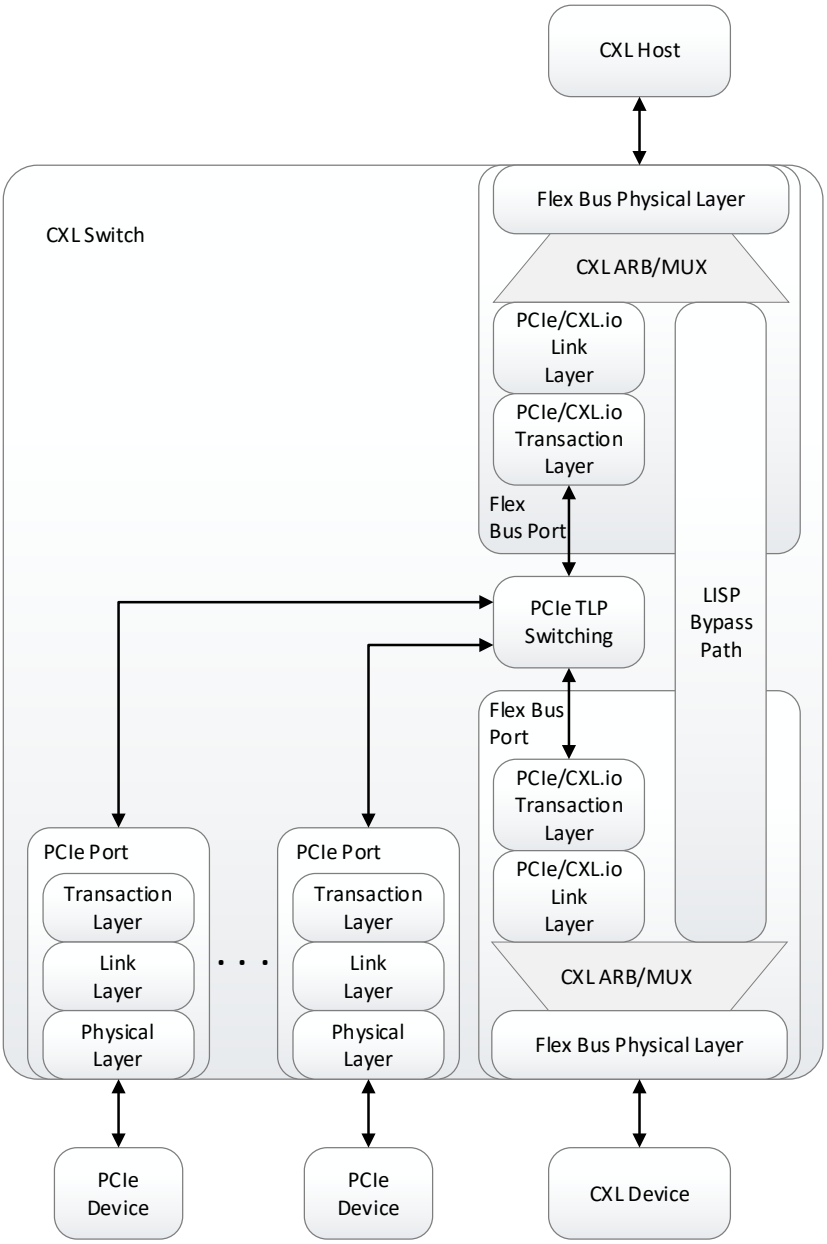


Figure 11-23 illustrates the Flit bypass path taken in a switch port which has been configured for LISP operation. CXL.io flits are detected by the ARB/MUX layer and passed to the PCIe/CXL.io Link and Transaction layers for normal processing. However, CXL.cachemem flits are not passed to the CXL.cache + CXL.mem Link and Transaction layers. Instead, these flits bypass these layers and are forwarded directly to their destination port within the switch.

Figure 11-24. CXL Switch in LISP Mode



Note: for sake of clarity, Figure 11-24 does not include the CXL.mem and CXL.cache Transaction and Link layer blocks (which have been bypassed) in the Flex Bus Ports.

When a switch has been configured to operate in LISP mode, as in Figure 11-24, its two CXL-enabled Flex Bus ports are bound to each other via the LISP bypass path for CXL.cachemem traffic. CXL.cachemem Flits received on either port are always forwarded to the other. Meanwhile, CXL.io flits are unpacked and TLPs forwarded to the switching logic as normal for forwarding to the appropriate egress port.

11.3.9.1.1 Physical Layer

There are no changes or impact to the Electrical sub-block by LISP. Likewise, the Logical sub-block continues to manage the Link and maintains responsibility for Flit headers, replay, etc. Even for CXL.cachemem Flits, these behaviors continue to be Link-based between the two physical ports of each link and are not extended over the bypass path. Creation and insertion of the Flit header, FEC, and CRC on a per-link

basis does not impact the Link IDE stream between the endpoints since those fields are not encrypted nor integrity protected.

IMPLEMENTATION NOTE: Replay Buffer backpressure

When a switch port has egress traffic (both CXL.io TLPs and CXL.cachemem flits) that exceeds its link bandwidth, congestion may result at the port. This may, in turn, result in congestion spreading to the other LISP port within the switch at the other end of the LISP bypass path. Eventually that port may not be able to receive any more flits on its link due to all receive buffering being filled. A switch port may handle this situation by delaying the flit ACK returned to the Root Port until the CXL.cachemem flit has been received in its egress buffer to the Device. This may cause the switch to exceed the PCIe recommendations for maximum flit-to-ACK latency. In this implementation, there are no required changes to a Root Port or Device port. However, congestion resulting from CXL.cachemem flits may result in blocking of CXL.io traffic when a Root Port or Device port has been backpressured.

Figure 11-24. CXL Switch in LISP Mode Requiring Link Backpressure

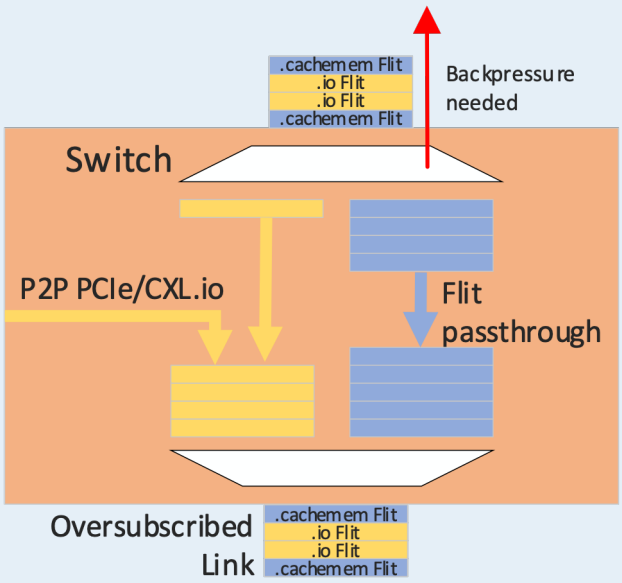


Figure 11-24 illustrates a switch port transmit path (bottom) which has become congested with P2P CXL.io traffic from a PCIe device and CXL.io + CXL.cachemem traffic from its LISP partner port (top). Backpressure is needed on the receiver port while the transmitting port drains.

11.3.9.1.2 ARB/MUX Layer

In order to mitigate blocking of CXL.io Flits at the Root Port or Device port when CXL.cachemem flits are congested, an optional but strongly recommended new feature is introduced. When enabled, a new type of ALMP message is used as a flow control mechanism for CXL.cachemem flits. These ALMP messages are sent from the switch to

the connected host or device, providing credit for the link partner to transmit the indicated number of CXL.cachemem flits. Transmission of the ALMP LISP credit message is an indication that the switch has sufficient buffering to receive the credited number of flits. Receipt of the ALMP LISP credit message by a host or device ARB/MUX increments a local counter by the number indicated in the message. Transmission of a CXL.cachemem flit decrements the counter. The counter is initialized to zero; the host or device may not begin transmitting CXL.cachemem flits until it receives an ALMP credit message. After the initial credit grant, the switch may only send additional credits equal to received CXL.cachemem flits as they are removed from the receive buffer. Note that this functionality is asymmetric on a link: only switch ports transmit ALPM LISP credit messages and only Root Ports and Device ports comprehend reception of the messages and react to them. This feature must only be enabled if both link partners support it, as indicated by the **LISP Credit Update ALMP Format Supported** bit in the **ARB/MUX LISP Status and Control** register.

11.3.9.1.3 Link Layer

There are no required changes to the Link Layer in LISP Root Port and device ports. The CXL.cachemem Link Layer is bypassed in Switch ports when configured for LISP. When operating in LISP mode, Switch ports must not unpack nor generate for transmission, messages in CXL.cachemem flit slots, whether encrypted or not, including LLCTRL messages and updates to CRD. In CXL hosts and devices, flits are unpacked and processed in the respective Link layers and messages passed to the Transaction layers. Exchange of CRD and LLCTRL messages, including INIT.param and IDE occurs between the Root Port and device port after software clears the LL_Init_Stall and LL_Crd_Stall bits in **CXL Link Layer Control and Status Registers**.

> **IMPLEMENTATION NOTE: Flow Control Buffer Sizing**
>
> LISP introduces additional latency between the two communicating link layers of the host and device: the switch ingress port, internal LISP Bypass path, switch egress port, and an additional physical link on the switch egress port. A CXL implementation normally chooses buffer sizing based on the anticipated latency of a single link and the directly attached port. Hosts and devices expecting to operate in LISP enabled deployments may choose to increase the message flow control buffering to account for the increased latency for optimal performance.

Update section 11.5.2 Scope with the following change

- This scope does not include the following:
  - CXL switches, except when the switch is configured for Link IDE Switch Passthrough; see **Section 11.3.9.1**.
    - Devices connected via a CXL switch, including MLDs, GFDs
    - Direct P2P using CXL.mem
    - Direct P2P using UIO over CXL.io

Add subsection 14.13.25 to Chapter 14

14.13.25 CXL LISP Capability Header

Test Steps:

1. Find this capability by reading all the elements within the Array_Size.
2. Read this element (1 DWORD).
3. Decode this into:

Bits	Variable
15:0	CXL_Capability_ID

19:16 CXL_Capability_Version
31:20 CXL LISP Capability Pointer

4. Save CXL LISP Capability Pointer, which is used in subsequent tests.
5. Verify:

Variable Value Condition

CXL_Capability_ID 0011h Always
CXL_Capability_Version 1h Always

Pass Criteria:

• Verify Conditions are met

Fail Conditions:

• Verify Conditions failed

Update existing test:

14.15.1 Sticky Register Test to add following registers in Sticky Register test

CXL LISP Control Register (Offset 04h)

Bits	Variable	Settings
0:0	LISP Bypass Path Enable	Set to 1h (if supported)

ARB/MUX LISP Status and Control Register (Offset 200h)

Bit	Variable	Settings
1:1	LISP Credit Update ALMP Enable	Set to 1h