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.

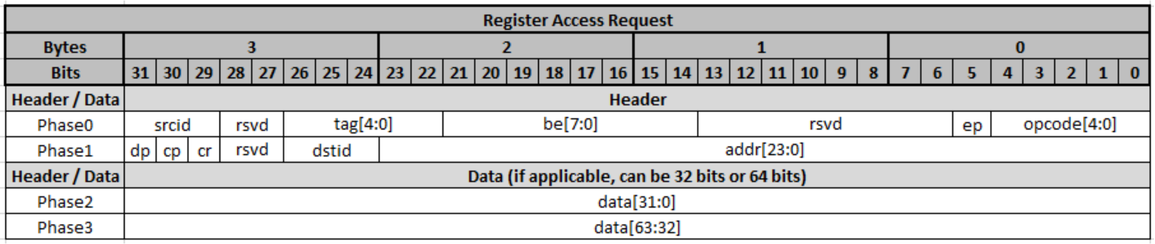
Table 7-5. Field descriptions for Register Access Requests

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	{RL[3:0], Offset[19:0]} Offset is the Byte Offset. RL[3:0] encodings are as follows: 0h: Register Locator 0 1h: Register Locator 1 2h: Register Locator 2 3h: Register Locator 3 Fh: Accesses for Protocol specific MMIO registers that are shadowed in the Adapter (e.g., ARB/MUX registers defined in the *CXL Specification*). The offsets for these registers are implementation specific, and the protocol layer must translate accesses to match the offsets implemented in the Adapter. Other encodings are reserved. For accesses to Reserved RL encodings, the completer must respond with a UR.
Configuration Reads/Writes	{RL[3:0], Rsvd[7:0], Byte Offset[11:0]}, where RL[3:0] encodings are as follows: 0h: UCIe Link DVSEC Fh: Accesses for Protocol specific configuration registers that are shadowed in the Adapter (e.g., ARB/MUX registers defined in the *CXL Specification*). The offsets for these registers are implementation specific, and the protocol layer must translate accesses to match the offsets implemented in the Adapter. Other encodings are reserved. For accesses to Reserved RL encodings, the completer must respond with a UR.
DMS Register Reads/Writes	These allow for accessing the DMS registers implemented in UCIe Spoke Type 0, 1, or 2. 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). Addr[23:22] encodings are as follows: 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.

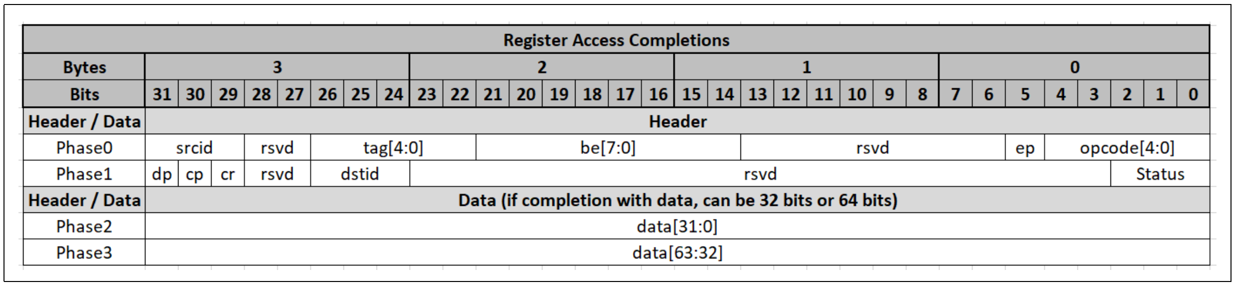
Figure 7-1. Format for Register Access Request



<table>
<tr><th colspan="33">Register Access Request</th></tr>
<tr><th>Bytes</th><th colspan="8">3</th><th colspan="8">2</th><th colspan="8">1</th><th colspan="8">0</th></tr>
<tr><th>Bits</th><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><th>Header / Data</th><th colspan="32">Header</th></tr>
<tr><td>Phase0</td><td colspan="3">srcid</td><td colspan="2">rsvd</td><td colspan="5">tag[4:0]</td><td colspan="8">be[7:0]</td><td colspan="8">rsvd</td><td>ep</td><td colspan="5">opcode[4:0]</td></tr>
<tr><td>Phase1</td><td>dp</td><td>cp</td><td>cr</td><td colspan="2">rsvd</td><td colspan="3">dstid</td><td colspan="24">addr[23:0]</td></tr>
<tr><th>Header / Data</th><th colspan="32">Data (if applicable, can be 32 bits or 64 bits)</th></tr>
<tr><td>Phase2</td><td colspan="32">data[31:0]</td></tr>
<tr><td>Phase3</td><td colspan="32">data[63:32]</td></tr>
</table>

Figure 7-2 gives the format for Register Access completions.

Figure 7-2. Format for Register Access Completions



<table>
<tr><th colspan="33">Register Access Completions</th></tr>
<tr><th>Bytes</th><th colspan="8">3</th><th colspan="8">2</th><th colspan="8">1</th><th colspan="8">0</th></tr>
<tr><th>Bits</th><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><th>Header / Data</th><th colspan="32">Header</th></tr>
<tr><td>Phase0</td><td colspan="3">srcid</td><td colspan="2">rsvd</td><td colspan="5">tag[4:0]</td><td colspan="8">be[7:0]</td><td colspan="8">rsvd</td><td>ep</td><td colspan="5">opcode[4:0]</td></tr>
<tr><td>Phase1</td><td>dp</td><td>cp</td><td>cr</td><td colspan="2">rsvd</td><td colspan="3">dstid</td><td colspan="21">rsvd</td><td colspan="3">Status</td></tr>
<tr><th>Header / Data</th><th colspan="32">Data (if completion with data, can be 32 bits or 64 bits)</th></tr>
<tr><td>Phase2</td><td colspan="32">data[31:0]</td></tr>
<tr><td>Phase3</td><td colspan="32">data[63:32]</td></tr>
</table>

Table 7-7 gives the field descriptions for a completion.

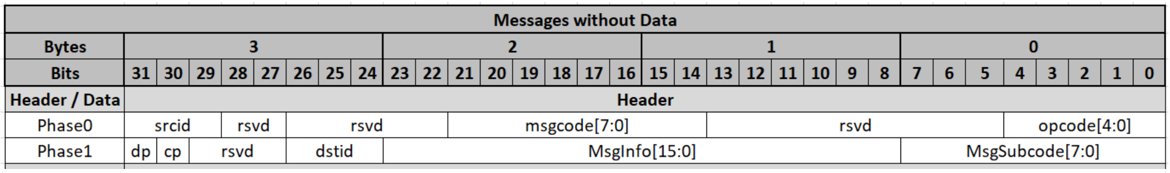
Table 7-7. Field Descriptions for a Completion

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



<table>
<tr><th colspan="33">Messages without Data</th></tr>
<tr><th>Bytes</th><th colspan="8">3</th><th colspan="8">2</th><th colspan="8">1</th><th colspan="8">0</th></tr>
<tr><th>Bits</th><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><th>Header / Data</th><th colspan="32">Header</th></tr>
<tr><td>Phase0</td><td colspan="3">srcid</td><td colspan="2">rsvd</td><td colspan="5">rsvd</td><td colspan="8">msgcode[7:0]</td><td colspan="9">rsvd</td><td colspan="5">opcode[4:0]</td></tr>
<tr><td>Phase1</td><td>dp</td><td>cp</td><td colspan="3">rsvd</td><td colspan="3">dstid</td><td colspan="16">MsgInfo[15:0]</td><td colspan="8">MsgSubcode[7:0]</td></tr>
</table>

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.Adapter1.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.Adapter1.Req.L1}		04h	Reserved	L1 Request for Stack 1 Adapter LSM.
{LinkMgmt.Adapter1.Req.L2}		08h		L2 Request for Stack 1 Adapter LSM.
{LinkMgmt.Adapter1.Req.LinkReset}		09h		LinkReset Request for Stack 1 Adapter LSM.
{LinkMgmt.Adapter1.Req.Disable}		0Ch		Disable Request for Stack 1 Adapter LSM.
{LinkMgmt.Adapter1.Rsp.Active}	06h	01h	0000h: Regular Response FFFFh: Stall Response	Active Response for Stack 1 Adapter LSM.
{LinkMgmt.Adapter1.Rsp.PMNAK}		02h		PMNAK Response for Stack 1 Adapter LSM
{LinkMgmt.Adapter1.Rsp.L1}		04h		L1 Response for Stack 1 Adapter LSM.
{LinkMgmt.Adapter1.Rsp.L2}		08h		L2 Response for Stack 1 Adapter LSM.
{LinkMgmt.Adapter1.Rsp.LinkReset}		09h		LinkReset Response for Stack 1 Adapter LSM.
{LinkMgmt.Adapter1.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 FFFFh: Stall Response	Parity Feature enable Ack.
{ParityFeature.Nak}		01h		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	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. 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.
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.TXSELFCAL Done req}	0000h	B5h	05h
{MBTRAIN.TXSELFCAL 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)

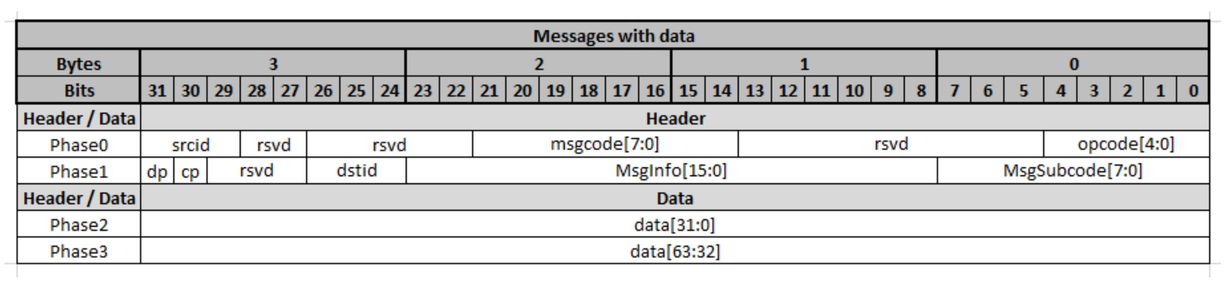
Message	MsgInfo[15:0]	MsgCode[7:0]	MsgSubcode[7:0]
{MBTRAIN.LINKSPEED start req}	0000h	B5h	15h
{MBTRAIN.LINKSPEED start resp}	0000h	BAh	15h
{MBTRAIN.LINKSPEED error req}	0000h	B5h	16h
{MBTRAIN.LINKSPEED error resp}	0000h	BAh	16h
{MBTRAIN.LINKSPEED exit to repair req}	0000h	B5h	17h
{MBTRAIN.LINKSPEED 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



<table>
<tr><th colspan="33">Messages with data</th></tr>
<tr><th>Bytes</th><th colspan="8">3</th><th colspan="8">2</th><th colspan="8">1</th><th colspan="8">0</th></tr>
<tr><th>Bits</th><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><th>Header / Data</th><th colspan="32">Header</th></tr>
<tr><td>Phase0</td><td colspan="3">srcid</td><td colspan="2">rsvd</td><td colspan="5">rsvd</td><td colspan="8">msgcode[7:0]</td><td colspan="9">rsvd</td><td colspan="5">opcode[4:0]</td></tr>
<tr><td>Phase1</td><td>dp</td><td>cp</td><td colspan="3">rsvd</td><td colspan="3">dstid</td><td colspan="16">MsgInfo[15:0]</td><td colspan="8">MsgSubcode[7:0]</td></tr>
<tr><th>Header / Data</th><th colspan="32">Data</th></tr>
<tr><td>Phase2</td><td colspan="32">data[31:0]</td></tr>
<tr><td>Phase3</td><td colspan="32">data[63:32]</td></tr>
</table>

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		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: • [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		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]: “Streaming Protocol” [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		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. 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.
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	[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)

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]:
	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	0Bh	[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]**}

{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]: UCIe-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 UCIe 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	[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. 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 [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. 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 [15:8]: **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: Reserved FFh: No Repair [7:0]: **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: Reserved FFh: No Repair

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	[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. 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 [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. 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 [15:8]: **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: Reserved FFh: No Repair [7:0]: **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: Reserved FFh: No Repair

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

<table>
<tr><td colspan="8">3</td><td colspan="8">2</td><td colspan="8">1</td><td colspan="8">0</td></tr>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="3">srcid=011b</td><td colspan="3">rsvd</td><td>resp</td><td colspan="3">vc</td><td colspan="8">msgcode</td><td colspan="8">length</td><td>rsvd</td><td colspan="5">opcode = 11000b</td></tr>
<tr><td>rsvd</td><td>cp</td><td colspan="3">rsvd</td><td colspan="3">dstid=111b</td><td colspan="16">msgcode-specific</td><td colspan="2">rsvd</td><td colspan="2">msgcode-specific</td><td colspan="2">rsvd</td><td colspan="2">rxqid</td></tr>
</table>

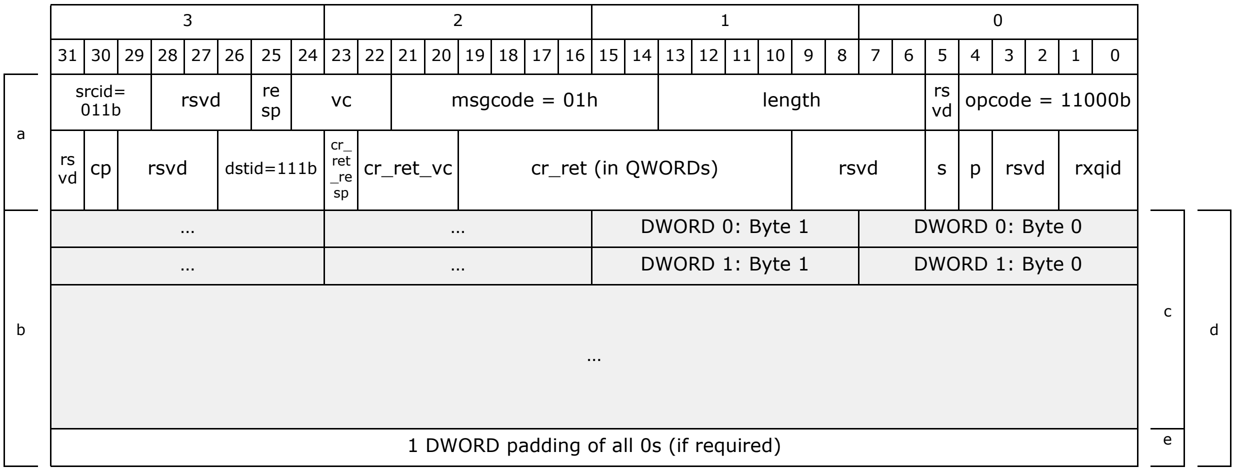
Table 7-13. Common Fields in MPM Header of all MPM with Data Messages on Sideband

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

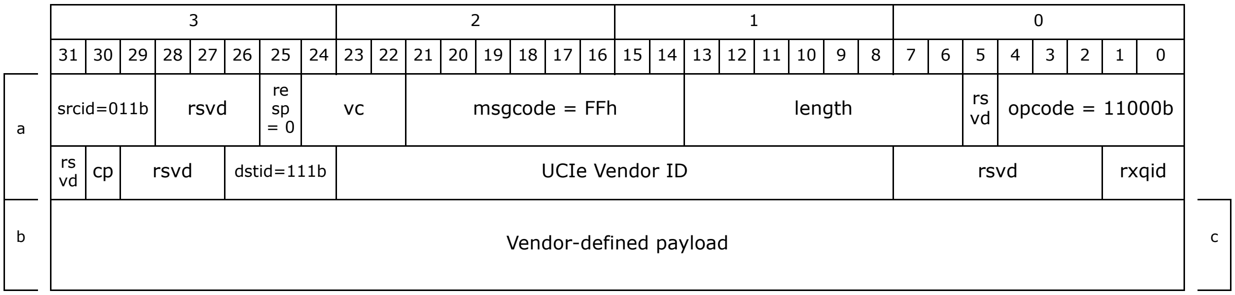
Location	Bit	Description
MPM Header[a]	s	Segmented MTP (see Section 8.2.4.2). The first and middle segments in a segmented MTP have this bit set to 1. The last segment in a segmented MTP will have this bit cleared to 0. An unsegmented MTP also has this bit cleared to 0.
	p	If this is set to 1, there 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

<table>
<tr><td colspan="8">3</td><td colspan="8">2</td><td colspan="8">1</td><td colspan="8">0</td></tr>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="3">srcid=011b</td><td colspan="7">rsvd</td><td colspan="8">msgcode</td><td colspan="8">msgcode-specific</td><td>rs vd</td><td colspan="5">opcode = 10111b</td></tr>
<tr><td>rs vd</td><td>cp</td><td colspan="3">rsvd</td><td colspan="3">dstid=111b</td><td colspan="20">msgcode-specific</td><td colspan="2">rsvd</td><td colspan="2">msgcode-specific</td></tr>
</table>

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

<table>
<tr><td></td><td colspan="8">3</td><td colspan="8">2</td><td colspan="8">1</td><td colspan="8">0</td></tr>
<tr><td></td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td rowspan="2">a</td><td colspan="3">srcid=011b</td><td colspan="7">rsvd</td><td colspan="8">msgcode = 01h</td><td colspan="3">NumVC</td><td colspan="6">rsvd</td><td colspan="5">opcode = 10111b</td></tr>
<tr><td>rs vd</td><td>cp</td><td colspan="3">rsvd</td><td colspan="3">dstid=111b</td><td colspan="16">Port ID[15:0]</td><td colspan="8">rsvd</td></tr>
</table>

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

<table>
<tr><td></td><td colspan="8">3</td><td colspan="8">2</td><td colspan="8">1</td><td colspan="8">0</td></tr>
<tr><td></td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td rowspan="2">a</td><td colspan="3">srcid=011b</td><td colspan="7">rsvd</td><td colspan="8">msgcode = 02h</td><td>cr_ret_resp_a</td><td colspan="3">cr_ret_vc_a</td><td>cr_ret_resp_b</td><td colspan="3">cr_ret_vc_b</td><td>rsvd</td><td colspan="5">opcode = 10111b</td></tr>
<tr><td>rsvd</td><td>cp</td><td colspan="2">rsvd</td><td colspan="4">dstid=111b</td><td colspan="10">cr_ret_a</td><td colspan="10">cr_ret_b</td><td colspan="2">rsvd</td><td colspan="2">rxqid</td></tr>
</table>

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
a	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

	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							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 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 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								rsvd	opcode = 10111b				
	rsvd	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.

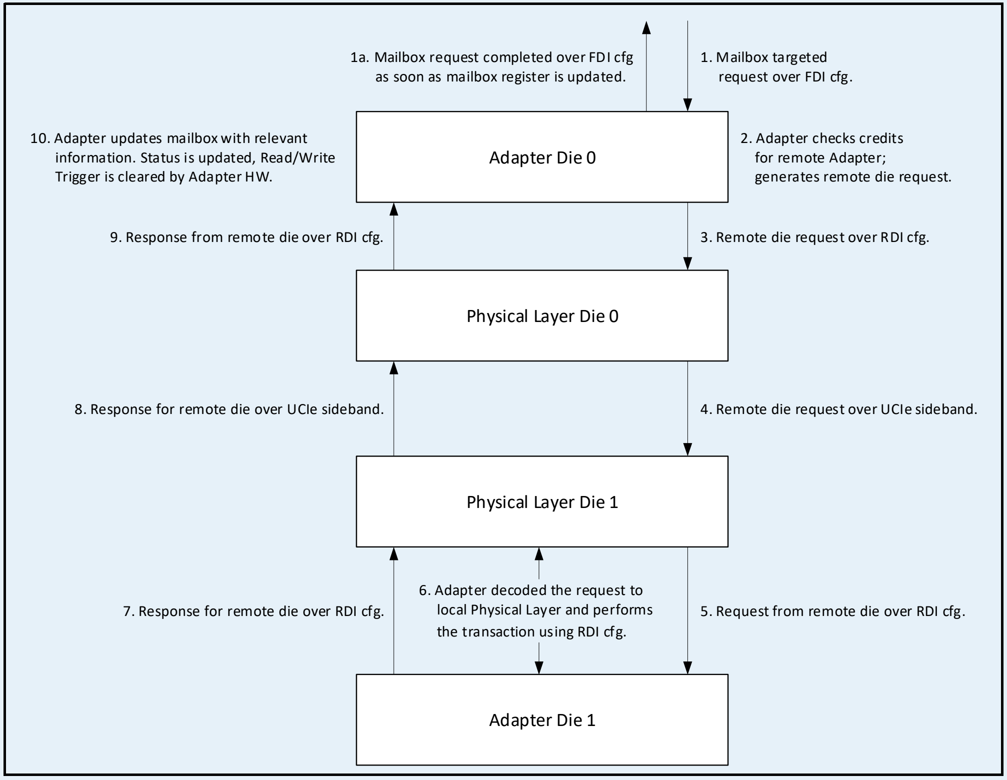
All other messages except Vendor Defined messages must always sink and make forward progress, and not block any messages on the sideband interface behind them. All Link Management message requests have an associated response, and the source of these messages must only have one outstanding request at 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)


1a. Mailbox request completed over FDI cfg as soon as mailbox register is updated.
1. Mailbox targeted request over FDI cfg.
10. Adapter updates mailbox with relevant information. Status is updated, Read/Write Trigger is cleared by Adapter HW.
Adapter Die 0
2. Adapter checks credits for remote Adapter; generates remote die request.
9. Response from remote die over RDI cfg.
3. Remote die request over RDI cfg.
Physical Layer Die 0
8. Response for remote die over UCIe sideband.
4. Remote die request over UCIe sideband.
Physical Layer Die 1
7. Response for remote die over RDI cfg.
6. Adapter decoded the request to local Physical Layer and performs the transaction using RDI cfg.
5. Request from remote die over RDI cfg.
Adapter Die 1

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.

§ §