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 7-15 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 7-15, Physical Layer Die 1 must ensure that it has dedicated space to sink the request in Step 6 independent of any ongoing remote register access request or completion from Die 1 to Die 0, or any other sideband message for state transition, etc. Similarly, Physical Layer Die 1 must have dedicated space for remote die register access completion in Step 7.

Dynamic coarse clock gating is permitted in Adapter or Physical Layer in a subset of the RDI states (see Chapter 10.0). Thus, when applicable, any sideband transfer over RDI or FDI must follow the clock gating exit handshake rules as defined in Chapter 10.0. It is recommended to always perform the clock exit gating handshakes for sideband transfers if implementations need to decouple dependencies between the interface status and sideband transfers.

Implementations of the Physical Layer and Adapter must ensure that there is no receiver buffer overflow for messages being sent over the UCIe sideband Link. This can be done by either ensuring that the time to exit clock gating is upper bounded and less than the time to transmit a sideband packet over the UCIe sideband Link, OR that the Physical Layer has sufficient storage to account for the worst-case backup of each sideband message function (i.e., remote register access requests, remote register access completions, Link state transition messages for Adapter LSM(s), Link state transition messages for RDI, and Link Training related messages). The latter offers more-general interoperability at the cost of buffers.

7.1.4 Operation on RDI and FDI

The same formats and rules of operation are followed on the RDI and FDI. The protocol is symmetric — requests, completions, and messages can be sent on **lp_cfg** as well as on **pl_cfg** signals. Implementations must ensure deadlock-free operation by allowing a sufficient quantity of sideband packets to sink and unblock the sideband bus for other packets. At the interface, these transactions are packetized into multiple phases depending on the configuration interface width (compile time parameter). Supported interface widths are 8, 16, or 32 bits. **lp_cfg_vld** and **pl_cfg_vld** are asserted independently for each phase. They must be asserted on consecutive clock cycles for transferring consecutive phases of the same packet. They may or may not assert on consecutive clock cycles when transferring phases of different packets. For packets with data, 64b of data is always transmitted over RDI or FDI; for 32b of valid payload, the most-significant 32b (Phase 4) of the packet are assigned to 0b before transfer.

§ §

8.0 System Architecture

8.1 UCIe Manageability

8.1.1 Overview

UCIe Manageability is optional and defines mechanisms to manage a UCIe-based SiP that is independent of UCIe mainband protocols. This accelerates the construction of a UCIe-based SiP by allowing a common manageability architecture and hardware/software infrastructure to be leveraged across implementations.

Examples of functions that may be performed using UCIe Manageability include the following:

- Discovery of chiplets that make up an SiP and their configuration,
- Initialization of chiplet structures, and parameters (i.e., serial EEPROM replacement),
- Firmware download,
- Power and thermal management,
- Error reporting,
- Performance monitoring and telemetry,
- Retrieval of log and crash dump information,
- Initiation and reporting of self-test status,
- Test and debug, and
- Various aspects of chiplet security.

UCIe manageability has been architected with the following goals:

- Support for management using mainband or sideband is mainband protocol agnostic allowing it to be used with chiplets that implement existing mainband protocols, mainband protocols that may be standardized in the future, and vendor-specific mainband protocols.
- The required core capabilities of UCIe manageability may be realized in hardware allowing simple chiplets to remain simple while providing advanced manageability capabilities (i.e., those that may require firmware implementation) to be implemented in chiplets that require these capabilities.
- UCIe Chiplets that support manageability may be used to realize products for a variety of markets. These markets may have vastly different manageability and security requirements. UCIe manageability defines a menu of optional management and security capabilities that build on top of the required core capabilities.
- UCIe manageability is intended to foster an open chiplet ecosystem where SiPs may be constructed from chiplets produced by different vendors. This means that common features and capabilities that are generally useful across market segments are standardized. Mechanisms are supplied for vendor-specific extensions.

- Manageability capabilities are discoverable and configurable, allowing a common firmware base to be rapidly used across SiPs.
- UCIe manageability builds on top of applicable industry standards.

8.1.2 Theory of Operation

A UCIe-based System-in-Package (SiP) that supports manageability contains one or more UCIe Chiplets that support UCIe manageability. Chiplets in the SiP that support UCIe manageability form a Management Domain. The SiP may also contain chiplets that do not support UCIe manageability and these chiplets are outside the Management Domain. If the Management Domain contains more than one chiplet, then the chiplets are interconnected through Management Ports using chiplet-to-chiplet management links to form a Management Network. The Management Network is fully connected meaning that there is a path from any chiplet in the Management Domain to any other chiplet in the Management Domain.

A UCIe Chiplet that supports manageability contains a Management Fabric and one or more Management Entities. A Management Entity is a Management Element, a Management Port, or a Management Bridge. An example UCIe chiplet that supports manageability is shown in Figure 8-1 and an example SiP that supports manageability consisting of four UCIe chiplets is shown in Figure 8-2.

Figure 8-1. Example UCIe Chiplet that Supports Manageability

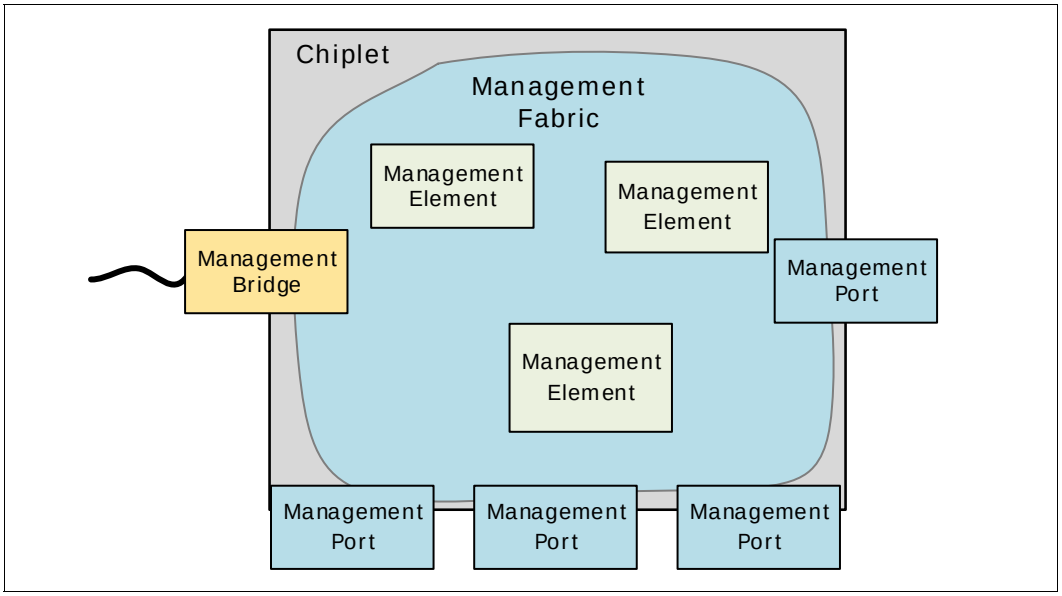
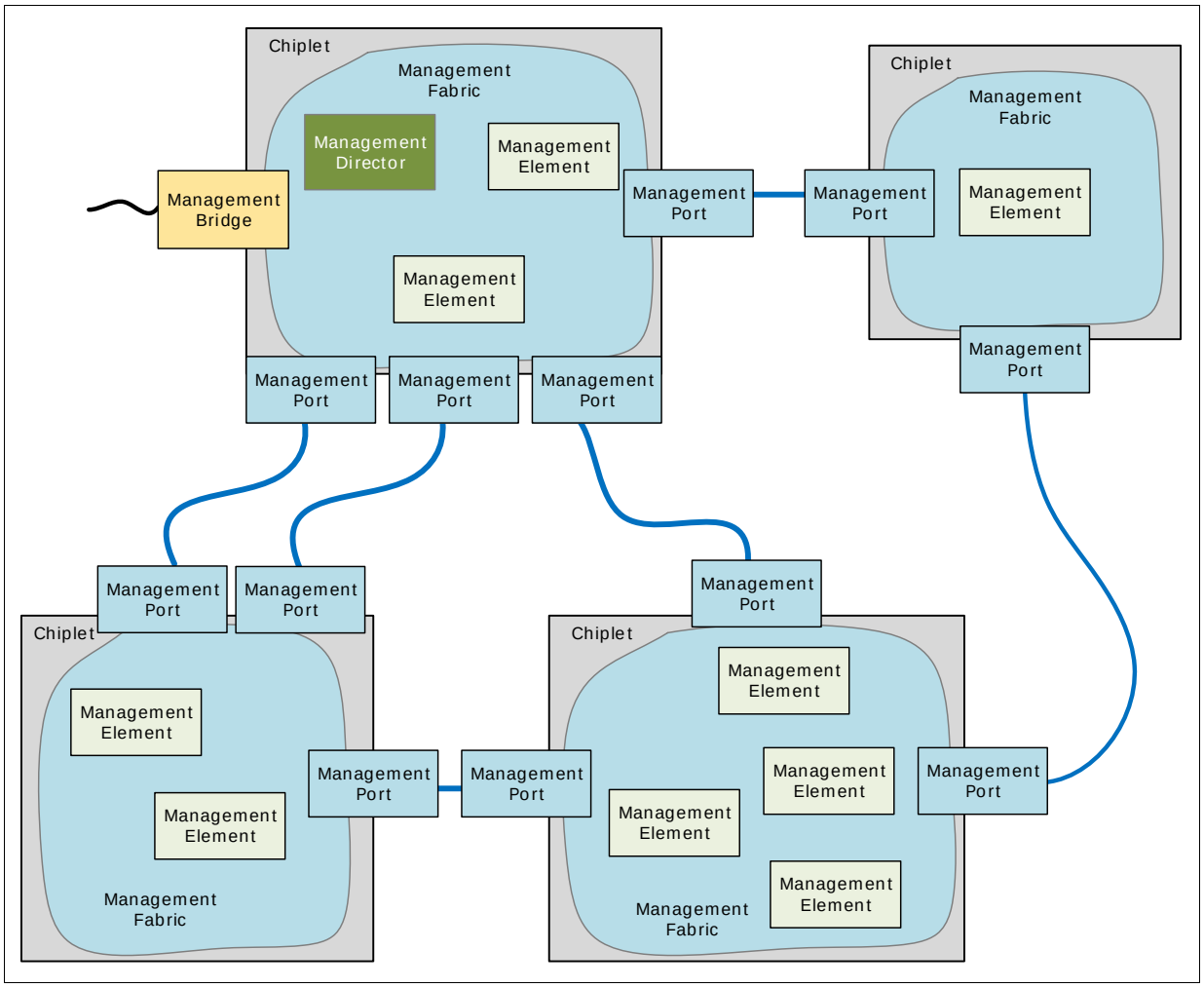
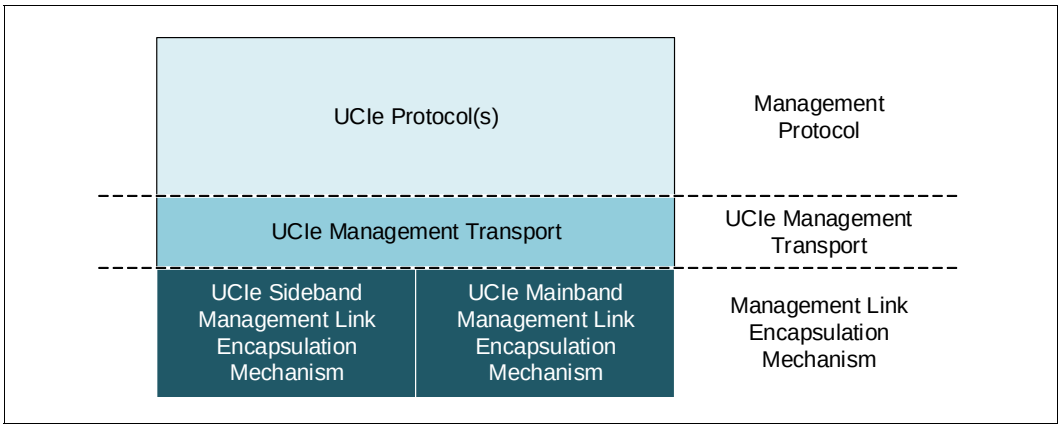


Figure 8-2. Example SiP that Supports Manageability



The UCIe Management Transport is an end-to-end media-independent protocol for management communication on the Management Network. This includes between Management Entities within a chiplet as well as between Management Entities in different chiplets. As shown in Figure 8-3, the Management Protocols above the UCIe Management Transport are used to implement management services. An example of a Management Protocol is the UCIe Memory Access Protocol.

Figure 8-3. UCIe Manageability Protocol Hierarchy



A Management Port is a Management Entity that acts as the interface between the Management Fabric within a chiplet and a point-to-point management link that interconnects two chiplets. As shown in Figure 8-3. UCIe Manageability Protocol Hierarchy, below the UCIe Management Transport is a Management Link Encapsulation Mechanism that defines how UCIe Management Transport packets are transferred across a point-to-point management link. Two Management Link Encapsulation Mechanisms are defined, one for the UCIe sideband and one for the UCIe mainband. See Section 8.2 for additional details of Management Link Encapsulation Mechanisms. Whether a specific UCIe sideband or mainband link in a chiplet may function as a point-to-point management link is implementation specific.

A chiplet that supports manageability should support at least one UCIe sideband Management Port. To enable broad interoperability, it is strongly recommended that a chiplet support enough UCIe sideband Management Ports to enable construction of an SiP with a single management domain using only UCIe sideband. If a chiplet supports management applications that require high bandwidth, such as test, debug, and telemetry, then it is strongly recommended that the chiplet support UCIe mainband Management Ports.

A Management Fabric within a UCIe chiplet facilitates communication between Management Entities inside the chiplet. A Management Entity is a Management Element, a Management Port, or a Management Bridge. The Management Fabric may be realized using one or more on-die fabrics and the implementation of the Management Fabric is beyond the scope of this specification.

A Management Element is a Management Entity that implements a management service. A Management Element must support the UCIe Management Transport protocol and one or more Management protocols.

A Management Bridge is a Management Entity that interconnects the Management Network to another interconnect, allowing agents on the Management Network and the other interconnect to communicate. The interconnect associated with a bridge may be internal to an SiP or external to the SiP.

One of the Management Elements within an SiP is designated as the Management Director. An SiP may contain multiple Management Elements that may act as a Management Director; however, there can only be one active Management Director at a time. How the Management Director is selected in such SiPs is beyond the scope of this specification. The roles of the Management Director include the following:

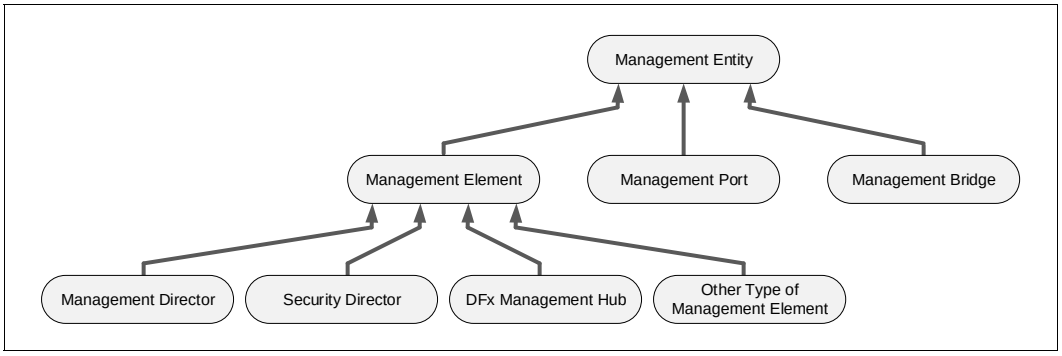
- Discovering chiplets and configuring Chiplet IDs,
- Discovering and configuring Management Elements,

- Discovering and configuring Management Ports,
- Discovering and configuring Management Bridges,
- Acting as the manageability Root of Trust (RoT), and
- Coordinating overall management of the SiP.

One or more of the Management Elements within an SiP may function as a Security Director. A Security Director is responsible for configuring security parameters.

The relationship between the various type of Management Entities is shown in Figure 8-4.

Figure 8-4. Relationship Between the Various Types of Management Entities



Unless otherwise specified, UCIe manageability, Management Entities, and all associated manageability structures in a chiplet (e.g., those in a Management Capability Structure) are reset on a Management Reset. A Management Reset occurs on initial application of power to a chiplet. Other conditions that cause a Management Reset are implementation specific.

8.1.3 UCIe Management Transport

The UCIe Management Transport is a protocol that facilitates communication between Management Entities on an SiP's Management Network. UCIe Management Transport packets are produced and consumed by Management Entities on the Management Network and the packets flow unmodified through the network.

8.1.3.1 UCIe Management Transport Packet

Figure 8-5 shows the fields that make up a UCIe Management Transport packet. The first two DWORDs contain the packet header. This is followed by a protocol specified field defined by the Management Protocol carried in the packet. Finally, a UCIe Management Transport packet may optionally contain a Packet Integrity value computed over the previous contents of the packet. If present, Packet Integrity is one DWORD in size.

Reserved fields in a UCIe Management Transport packet must be filled with 0s when the packet is formed. Reserved fields must be forwarded unmodified on the Management Network and ignored by receivers. An implementation that relies on the value of a reserved field in a packet is non-compliant.

Figure 8-5. UCIe Management Transport Packet

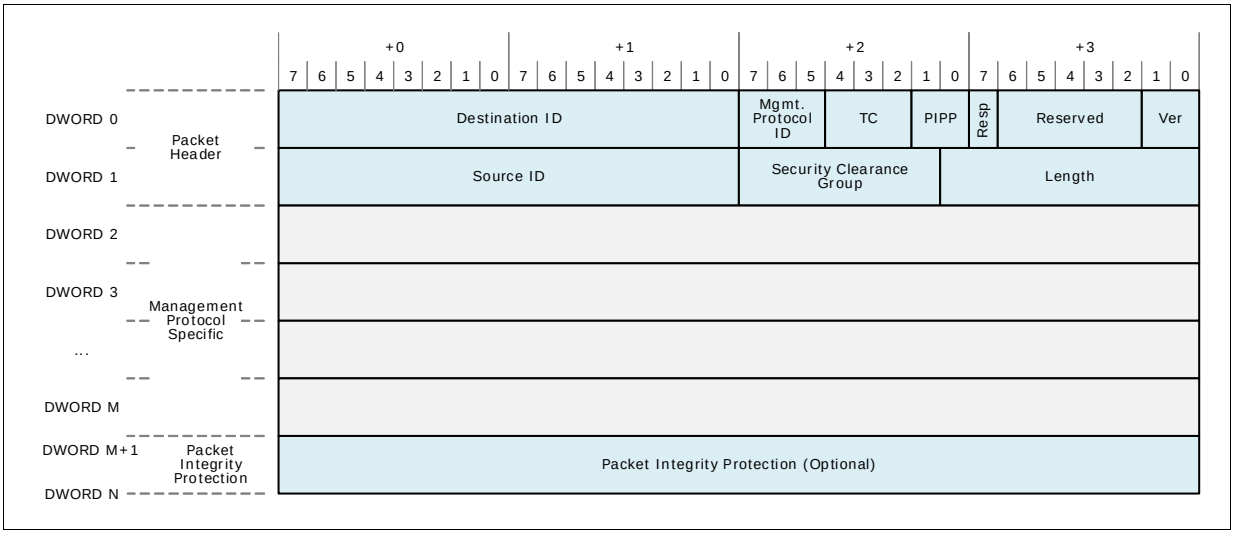


Table 8-1 defines the fields of a UCIe Management Transport packet. All fields in the table have little endian bit ordering (e.g., Destination ID bits 0 through 7 are in Byte 1 with bit 0 of the Destination ID in Byte 1 bit 0, and Destination ID bits 8 through 15 are in Byte 0 with Destination ID bit 8 in Byte 0 bit 0).

Table 8-1. UCIe Management Transport Packet Fields (Sheet 1 of 2)

Field Name	Field Size	Description
Destination ID	16 bits	**Destination ID** This field specifies the Management Network ID of the entity on the Management Network that is to receive the packet.
Mgmt. Protocol ID	3 bits	**Management Protocol ID** This field contains an ID that corresponds to a Management Protocol and specifies the type of payload contained in the Management Protocol Specific field. See Section 8.1.3.1.3 for ID values.
TC	3 bits	**Traffic Class** This field is a packet label used to enable different packet servicing policies. Each Traffic Class is a unique ordering domain with no ordering guarantees between packets in different Traffic Classes. See Section 8.1.3.1.1.
PIPP	2 bits	**Packet Integrity Protection Present (PIPP)** A nonzero value in this field indicates that the packet contains a packet integrity protection value in the Packet Integrity Protection field. The type of packet integrity is indicated by the nonzero value. See Section 8.1.3.4 for more information. 00b: Packet Integrity Protection field is not present in the packet 11b: CRC Integrity (one DWORD in size) Others: Reserved
Resp	1 bit	**Request or Response** This field indicates whether the packet is a request or a response. 0: Request packet 1: Response packet
Reserved	5 bits	**Reserved**
Ver	2 bits	**UCIe Management Transport Packet Version** This field indicates the version of the UCIe Management Transport packet. This field must be set to 00b in this version of the specification. If a Management Entity receives a packet with a UCIe Management Transport packet version that it does not support, then Management Entity must discard the packet.

Table 8-1. UCIe Management Transport Packet Fields (Sheet 2 of 2)

Field Name	Field Size	Description
Source ID	16 bits	**Source ID** This field indicates the Management Network ID of the entity on the Management Network that originated the packet.
Security Clearance Group	7 bits	**Security Clearance Group** This field is used by the UCIe Management Transport access control mechanism. In request packets, this field is set to the security clearance group value associated with the request. In response packets, this field must be cleared to 00h. ***Note:*** A response packet is handled at the same Security Clearance Group as its corresponding request packet.
Length	9 bits	**Length** This field indicates the length of the entire packet in DWORDs. This includes the UCIe Management Network Header, the Management Protocol Specific field, and the Packet Integrity Protection field if present. The length of the packet in DWORDs is equal to the value of this field plus 1 (e.g., a value of 000h in this field indicates a packet length of one DWORD, a value of 001h in this field indicates a packet length of two DWORDs, and so on).
Management Protocol Specific	Varies	**Management Protocol Specific** This field contains Management Protocol specific packet contents. The format of this field is indicated by the Management Protocol field. This field must be an integral number of DWORDs.
Packet Integrity Protection	Varies	**Packet Integrity Protection** If the Packet Integrity Protection Present (PIPP) field in the packet has a nonzero value, then this field is present in the packet and corresponding packet integrity value. See Section 8.1.3.4 for more information. This field must be an integral number of DWORDs. In this version of the specification only CRC integrity protection is supported which is always one DWORD.

A request packet is a packet with the Resp field in the UCIe Management Transport packet header assigned to 0. A requester is a Management Entity that first introduces a request packet into a Management Fabric. A responder is the Management Entity that performs the actions associated with a request that consists of one or more request packets and is the ultimate destination of these request packet(s). A response packet is a packet with the Resp field in the UCIe Management Transport packet header assigned to 1. As a result of performing the actions associated with a request, the responder may introduce one or more response packets into the Management Fabric destined to the requester.

A Management entity that receives a malformed packet must discard the packet.

- A packet with an incorrect length in the Length field is malformed.
  - An example of a malformed packet with an incorrect length in the Length field is one where the Length field in the UCIe Management Transport packet header indicates a length of 65 DWORDs but the actual length of the packet is 64 DWORDs.
- A packet whose size exceeds the Management Transport Packet size supported by a chiplet is malformed.
- A packet that violates a requirement outlined in this specification is malformed.
  - An example of a malformed packet due to a requirement violation is a response packet with a nonzero value in the Security Clearance Group field.

8.1.3.1.1 Traffic Class and Packet Ordering Requirements

Traffic classes (TCs) are used to enable different packet servicing policies. The UCIe Management Transport supports eight traffic classes, and the characteristics of each traffic class are described in

Table 8-2. The Management Director configures management fabric routes and may configure different routes for different traffic classes (e.g., TC0 traffic may be routed to UCIe sideband Management Port A and TC2 traffic may be routed to UCIe mainband Management Port B).

Table 8-2. Traffic Class Characteristics

Traffic Class	Description
0	**Default Ordered Lossless Traffic Class** Traffic class optimized for packets that require high reliability and availability. Example: Configuration and health monitoring packets
1	**Low Latency Ordered Lossless Traffic Class** This traffic class has the same characteristics as the Default Ordered Lossless Traffic Class but is intended to be lightly loaded and used for packets that require low latency and should bypass congested Default Ordered Lossless Traffic Class packets. Example: Debug cross trigger and low latency power management packets
2	**Bulk Lossless Ordered Performance Traffic Class** Traffic class optimized for packets associated with bulk traffic. In a properly functioning system without errors, packets in this traffic class are never dropped. Example: Firmware download packets
3	**Bulk Lossy Ordered Performance Traffic Class** Traffic class optimized for packets associated with bulk traffic that may be dropped in the presence of congestion. Example: Debug trace packets
4	**Unordered Lossless Traffic Class** Traffic class optimized for packets that require high performance. Request packets in this traffic class may be delivered out-of-order. Response packets in this traffic class may be delivered out-of-order. Example: High performance UCIe Memory Access protocol accesses.
5 to 6	**Reserved**
7	**Vendor-Specific Traffic Class**

A request packet and an associated response packet need not have the same traffic class. Which traffic classes are allowed and how they are mapped between a request and response are Management Protocol specific.

An implementation shall ensure forward progress on all traffic classes. Quality of service between traffic classes is implementation specific and beyond the scope of this specification.

Each traffic class is a unique ordering domain and there are no ordering guarantees for packets in different traffic classes and there are no ordering guarantees between request and response packets in the same traffic class. Regardless of the traffic class, a response packet associated with any traffic class must be able to bypass a blocked request packet associated with any traffic class.

Within an ordered traffic class, request packets are delivered in-order from a requester to a responder and response packets are delivered in-order from a responder to the requester. There are no ordering guarantees between requests to different responders and there are no ordering guarantees between responses from different responders to a requester.

Within an unordered traffic class there are no ordering guarantees between packets of any type and the chiplet's Management Fabric is free to arbitrarily reorder packets. While packets may be reordered on an unordered traffic class, there is no requirement that they be reordered (i.e., an implementation is free to maintain ordering as in an ordered traffic class).

Packets associated with a lossy traffic class may be dropped during normal operation. The policy used to determine when a lossy traffic class packet is dropped is vendor defined and beyond the scope of this specification (e.g., due to exceeding a vendor defined congestion threshold). Lossless traffic

classes are reliable, and packets associated with a lossless traffic class are not dropped during normal operation; however, packets associated with a lossless traffic class may be dropped due to an error condition. The detection of lost packets and recovery from lost packets is the responsibility of a Management Protocol.

To maintain forward compatibility, a UCIe Management Transport packet with a reserved traffic class value is treated in the same manner as a packet with a traffic class value of zero (i.e., Default Ordered Lossless Traffic Class).

To maintain interoperability, all implementations are required to support all traffic classes; however, an implementation is only required to maintain the ordered and lossless characteristics of a traffic class. All other traffic class characteristics may be ignored. For example, an implementation may treat all traffic classes in the same manner as the Default Ordered Lossless Traffic Class. Under no circumstances may packets in an ordered traffic class be reordered between a requester and a responder. Similarly, under no circumstances may a packet in a lossless traffic class be dropped in a properly functioning system without errors.

> **IMPLEMENTATION NOTE**
>
> **Chiplet Route Through**
>
> Chiplet route through occurs when a UCIe Management Transport packet flows through a chiplet (i.e., the packet enters a chiplet through a UCIe Management Port, is not destined to any Management Entity within the chiplet, and the packet matches a Route Entry that causes it to be routed out a UCIe Management Port). Due to chiplet route through it is desirable that chiplets implement the packet servicing policies associated with a TC even if none of the Management Entities within the chiplet utilize that TC. Chiplets are always required to support chiplet route through for all traffic classes.

8.1.3.1.2 Packet Length

The length of a Packet is indicated by the Length field, is an integral number of DWORDs, and consists of the entire length of the packet (i.e., the UCIe Management Network Header, the Management Protocol Specific field, and Packet Integrity Protection field if present.).

The maximum packet length architecturally supported by the UCIe Management Transport is 512 DWORDs (i.e., 2048B). A chiplet may support a maximum packet length that is less than the architectural limit. The Maximum Packet Size (MPS) field in the Chiplet Capability Structure indicates the maximum packet size supported by the chiplet. If a packet larger than that advertised by MPS is received on a Management Port or Management Bridge, then it is silently discarded and not emitted on the chiplet's Management Fabric.

The Configured Maximum Packet Size (CMPS) field in the Chiplet Capability Structure controls the maximum UCIe Management Transport packet size generated by a Management Entity within the chiplet. The initial value of this field corresponds to 8 DWORDs (i.e., 64B). The CMPS field does not affect UCIe Management Transport packets emitted by Management Ports and Management Bridges when forwarding packets into a chiplet's Management Fabric. This allows packets to be routed through the chiplet (e.g., between Management Ports) that are larger than the size of packets generated by Management Entities within the chiplet.

8.1.3.1.3 Management Protocol

The Management Protocol field in a packet contains a Management Protocol ID that indicates the format of the Management Protocol Specific field. Table 8-3 lists the Management Protocols supported by the UCIe Management Transport.

Table 8-3. Management Protocol IDs

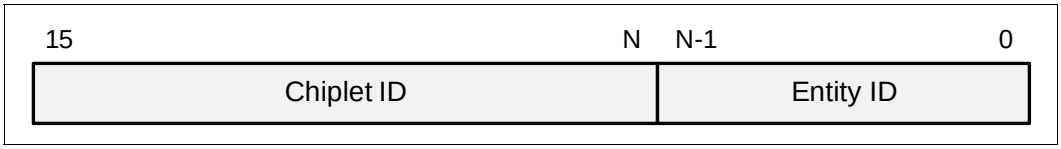
Management Protocol ID	Description
0	Reserved
1	UCIe Memory Access Protocol This protocol is used to access memory mapped structures and memory.
2	UCIe Test and Debug Protocol
3 to 6	Reserved
7	Vendor defined

8.1.3.2 Management Network ID

Management Network IDs are used to uniquely identify Management Entities in the Management Network and to route UCIe Management Transport packets between Management Entities.

As shown in Figure 8-6, a Management Network ID is a 16-bit field that consists of the concatenation of a Chiplet ID and an Entity ID. The Chiplet ID uniquely identifies a chiplet within an SiP and the Entity ID is a fixed value that uniquely identifies a Management Entity within a chiplet. Together the Chiplet ID and Entity ID uniquely identify each Management Entity in an SiP.

Figure 8-6. Management Network ID Format



The size of the Chiplet ID in bits is chiplet implementation specific and may be 2-bits to 15-bits in size. The size of the Entity ID in bits is also chiplet implementation specific and may be 1-bit to 14-bits in size. For each chiplet, the sum of the size of the Chiplet ID and the Entity ID must be 16-bits. The size of the Chiplet ID and Entity ID fields may be different in different chiplets in an SiP (i.e., it is not a requirement that all chiplets of an SiP have the same Chiplet ID field size). To facilitate interoperability an implementation should make the Chiplet ID field as large as possible (i.e., make the Entity ID field only as large as needed to address Management Entities in the chiplet).

The Management Director initializes the Chiplet ID value associated with each chiplet during SiP initialization. This is done by writing the Chiplet ID value to the Chiplet field in the Chiplet Capability Structure and setting the Chiplet ID Valid bit in that structure. The Management Director may determine the size of the Chiplet ID associated with a chiplet by examining the initial value of the Chiplet ID field in the Chiplet Capability Structure or by determining which bits are read-write in this field.

Each Management Entity within a chiplet has a fixed Entity ID. Entity ID zero within each chiplet must be a Management Element that is used to initialize and configure the chiplet. Entity IDs within a chiplet that map to Management Entities may be sparse. For example, a chiplet may contain Management Entities at Management Entity IDs zero, one, and three. The other Entity IDs are unused. A UCIe Management Packet that targets an unused Entity ID within a chiplet is silently discarded by the chiplet's Management Fabric.

IMPLEMENTATION NOTE

Configuring Routing in an SiP with Different Chiplet ID Sizes

A System-in-Package (SiP) may be composed of chiplets with varying sizes for the Chiplet ID portion of the Management Network ID. To establish routing within such an SiP, one approach is to identify the smallest Chiplet ID size among all chiplets in the SiP. Subsequently, Route Entries (as described in Section 8.1.3.6.2.2) are configured as if each chiplet possessed a Chiplet ID size equivalent to this minimum. For chiplets with a Chiplet ID size exceeding the minimum, any unused Chiplet ID bits are assigned 0 in the Base ID field of a Route Entry and set to 1 in the Limit ID field of a Route Entry.

8.1.3.3 Routing

UCIe Management Transport packets are routed based on the Destination ID field in the UCIe Management Network Header.

The routing of UCIe Management Transport packets differs depending on whether the Chiplet ID value is initialized or uninitialized. The Chiplet ID value is initialized when the Chiplet ID Valid bit is set to 1 in the Chiplet Capability Structure. The Chiplet ID value is uninitialized when the Chiplet ID Valid bit is cleared to 0 in the Chiplet Capability Structure.

A UCIe Management Transport request packet is one where the Resp field is cleared to 0. A UCIe Management Transport response packet is one where the Resp field is set to 1.

While the Chiplet ID and Entity ID size of chiplets may vary in an SiP, all packet routing associated with a chiplet is performed using the Chiplet ID size of the chiplet performing the routing.

The method used to configure UCIe Management Transport routing within an SiP is beyond the scope of this specification. The routing may be pre-determined during SiP design and this static configuration may be used by the Management Director to configure routing. Alternatively, the Management Director may discover the SiP configuration (i.e., chiplets, Management Network topology, and Chiplet ID size of each chiplet) and use this information to dynamically configure routing.

Because the Management Network may contain redundant management links between chiplets as well as links that form cycles, care must be exercised to ensure that the UCIe Management Transport routing is acyclic and deadlock free. In the absence of faults, request packets and response packets are delivered in order; however, it is possible to configure UCIe Management Transport routing such that the path used by a request packet from a requester to a responder is different from path used by a response packet from the responder back to the requester.

8.1.3.3.1 Routing of a Packet from a Management Entity within the Chiplet

This section describes routing of a UCIe Management Transport packet generated by a Management Entity within the chiplet.

- If the chiplet's Chiplet ID value is initialized, then the packet is routed as follows.
  - If the Chiplet ID portion of the packet's Destination ID matches the Chiplet ID value of the chiplet performing the routing, the packet is routed within the chiplet based on the Entity ID portion of the packet's Destination ID.
    - If the Entity ID portion of the packet's Destination ID matches that of a Management Entity within the chiplet, then the packet is routed to that Management Entity.

    - If the Entity ID portion of the packet's Destination ID does not match that of any Management Entity within the chiplet, then the packet is discarded.
  - If the Chiplet ID portion of the packet's Destination ID does not match the Chiplet ID value of the chiplet, then the packet is routed based on Management Port Route Entries.
    - If the packet's Destination ID matches a Route Entry associated with a Management Port, then the packet is routed out that Management Port. See Section 8.1.3.6.2.2 for Route Entry matching rules.
    - If the packet's Destination ID matches multiple Route Entries within the chiplet and the packet is associated with an ordered traffic class, then the packet is discarded.
    - If the packet's Destination ID matches multiple Route Entries within the chiplet and the packet is associated with an unordered traffic class, then the packet is routed out one of the Management Ports with a matching Route Entry. Which matching Route Entry is selected in the unordered traffic class case is vendor defined.
    - If the packet's Destination ID does not match any Route Entries within the chiplet, then the packet is discarded.
- If the chiplet's Chiplet ID value is uninitialized, then packet is routed as follows.
  - If the packet is a UCIe Management Transport Response packet and the corresponding UCIe Management Transport Request packet was received from a Management Port, then the packet is routed as follows.
    - If the link is up that is associated with Management Port on which the corresponding UCIe Management Transport Request packet was received, then the response packet is routed out that same Management Port on virtual channel zero (VC0). This allows a Management Entity outside the chiplet to configure the chiplet before the chiplet's Chiplet ID value is initialized.
    - If the link associated with Management Port on which the corresponding UCIe Management Transport Request packet was received is not up (a link may be down after receiving a UCIe Management Transport Request packet for reasons such as link instability, or a write to the Link Not Up field, etc.), then the response packet is discarded.
  - Otherwise, the packet is routed within the chiplet based on the Entity ID portion of the packet's Destination ID.
    - If the Entity ID portion of the packet's Destination ID matches that of a Management Entity within the chiplet, then the packet is routed to that Management Entity.
    - If the Entity ID portion of the packet's Destination ID does not match that of any Management Entity within the chiplet, then the packet is discarded.

IMPLEMENTATION NOTE

Routing for Uninitialized Chiplet IDs

When a chiplet's Chiplet ID value is uninitialized, then a packet received on a chiplet's Management Port is routed based on the Entity ID portion of the packet's Destination ID to a Management Entity within the chiplet, if one exists. When a response packet is emitted by this Management Entity, the packet is routed out the same Management Port on which the corresponding request was received. This allows an agent external to the chiplet to configure the chiplet before the Chiplet ID and Route Entries are initialized (e.g., following a Management Reset).

When a response packet is routed out the Management Port on which the corresponding request was received, it is routed out the Management Port on virtual channel zero. While there is no requirement that requests to uninitialized chiplets use virtual channel zero (VC0), it is strongly encouraged that virtual channel zero be used.

8.1.3.3.2 Routing of a Packet Received on a Management Port

This Section describes routing of a UCIe Management Transport packet received on a chiplet's Management Port.

- If the chiplet's Chiplet ID value is initialized, then the packet is routed in the same manner as a packet generated by a Management Entity within the chiplet. See Section 8.1.3.3.1 for how such a packet is routed.
- If the chiplet's Chiplet ID value is uninitialized, then the packet is routed within the chiplet based on the Entity ID portion of the packet's Destination ID.
  - If the Entity ID portion of the packet's Destination ID matches that of a Management Entity within the chiplet, then the packet is routed to that Management Entity.
  - If the Entity ID portion of the packet's Destination ID does not match that of any Management Entity within the chiplet, then the packet is discarded.

8.1.3.4 Packet Integrity Protection

A UCIe Management Transport packet may optionally contain a Packet Integrity Protection field that is used to protect the integrity of the packet. The presence and type of packet integrity are indicated by the Packet Integrity Protection Present (PIPP) field in the packet.

CRC protection, defined in Section 8.1.3.4.1, is the only packet integrity protection currently defined and is one DWORD in size.

8.1.3.4.1 CRC Integrity Protection

When the PIPP field in a packet is set to 11b, then the Packet Integrity Protection field in the packet is one DWORD in size and contains a 32-bit CRC computed over the previous contents of the packet (i.e., the UCIe Management Network Header and Management Protocol Specific field). Each bit of the Packet Integrity Protection field is set to the corresponding bit of the computed CRC (e.g., bit 31 of the computed CRC corresponds to bit 31 of the Packet Integrity Protection field).

The 32-bit CRC required by this specification is CRC-32C (Castagnoli) which uses the generator polynomial 1EDC6F41h. The CRC is calculated using the following Rocksoft™ Model CRC Algorithm parameters:

Name : "CRC-32C"
Width : 32
Poly : 1EDC6F41h
Init : FFFFFFFFh
RefIn : True
RefOut : True
XorOut : FFFFFFFFh
Check : E3069283h

When the PIPP field in a packet is set to 11b and the CRC contained in the Packet Integrity Protection field is incorrect, then the packet is discarded.

8.1.3.5 Access Control

The Management Network in an SiP may contain multiple Management Entities that issue requests and multiple Management Entities that expose assets (e.g., structures, keys, or memory). The UCIe Management Transport supports an access control mechanism that may be used by a Management Protocol to prevent unauthorized access to assets by Management Entities on the Management Network.

Some Management Protocols have their own security architecture, so the use of the access control mechanism is Management Protocol specific. Table 8-4 shows which Management Protocols use the access control mechanism.

Table 8-4. Management Protocol use of Access Control Mechanism

Management Protocol	Uses Access Control Mechanism
UCIe Memory Access Protocol	Yes
UCIe Test and Debug Protocol	Yes[a]
Vendor Defined	Vendor Defined

a. The UCIe Test and Debug Protocol uses the UCIe Memory Access Protocol for configuration and status field accesses and as a result uses the Access Control Mechanism.

The access control mechanism is based on a 7-bit security clearance group value contained in request packets. When a Management Entity emits a request packet on the Management Network, it populates the Security Clearance Group field in the packet with a value that corresponds to the security clearance group associated with the requester. When a Management Entity receives a request packet, it determines whether the asset(s) accessed by the request are allowed or prohibited by that security clearance group.

A Management Entity may emit packets with different security clearance group values. How the security clearance group values that a Management Entity emits are configured or selected is beyond the scope of this specification (see Section 8.1.3.6.4.1).

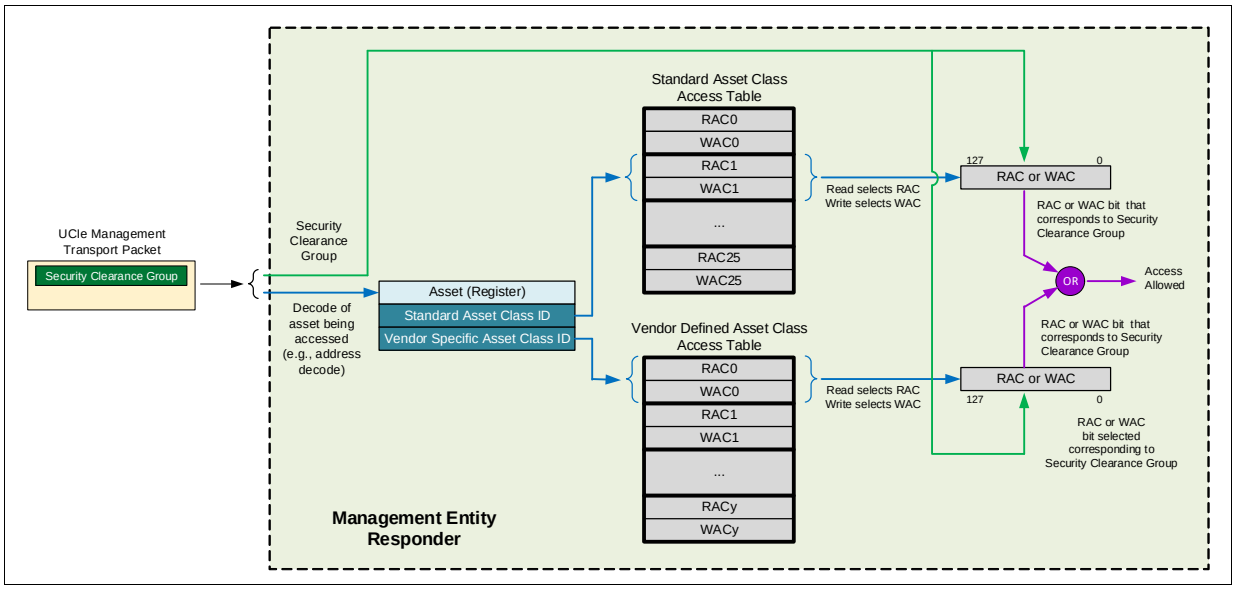
Although the security clearance group value is seven bits in size, an implementation may choose to restrict the number of security clearance groups. When an implementation restricts the number of security clearance groups to a value N, then security clearance group values from 0 to (N-1) are allowed, and security clearance group values from N to 127 are not allowed. Security Clearance Group 0 is dedicated for use by Security Directors (see Section 8.1.3.5.2).

The method a Management Entity uses to determine whether a request packet is allowed access to an asset is shown in Figure 8-7 and consists of the following steps.

1. The standard asset class ID or the vendor defined asset class ID of the asset being accessed by the request packet is determined.
   - UCIe defines standard asset classes (see Section 8.1.3.5.1) and supports vendor defined asset classes. Each asset must map to a standard asset class, a vendor defined asset class, or both.
   - Associated with each asset class is an asset class ID. The mapping associated with this step produces a standard asset class ID, a vendor defined asset class ID, or both.
   - The manner and granularity in which an implementation maps assets to asset class IDs are beyond the scope of this specification and may be done as part of address decoding, tagging of assets, or some other mechanism.
2. Each asset class ID determined in the previous step is mapped to a 256-bit access control structure.
   - Associated with each asset class ID is a 128-bit Read Access Control (RAC) structure and a 128-bit Write Access Control (WAC) structure. If an asset's state is being read, then the RAC structure is selected as the access control structure. If an asset's state is being written/ modified, then the WAC structure is selected as the access control structure.
     - RAC and WAC structures are contained in the Access Control Capability Structure (see Section 8.1.3.6.3).
     - If a Management Entity does not have any assets that correspond to an asset class ID, then the RAC and WAC structures associated with that asset class ID must be hardwired to 0 (i.e., all the bits on both the RAC and WAC structures are read-only with a value of 0) so no accesses would be allowed.
     - If an implementation restricts the number of security clearance groups, then RAC and WAC structure bits associated with security clearance groups that are not supported must be hardwired to 0. For example, if an implementation only supports Security Clearance Groups 0 through 3, then bits 4 through 127 must be read-only with a value of 0 in all RAC and WAC structures.
3. The bit corresponding to the security clearance group value in the request packet is examined in each access control structure determined in the previous step to determine whether access to the asset is allowed.
   - The 7-bit security clearance group value in the request packet is a value from 0 to 127 and this value maps to a corresponding bit in a 128-bit access structure (e.g., security clearance group value 27 maps to access control structure bit 27).
   - If a bit corresponding to the security clearance group value in an access control structure is set to one, then access to that asset is allowed by that security clearance group. If a bit corresponding to the security clearance group value in an access control structure is set to 0, then access to that asset is prohibited by that security clearance group.
4. If access to the asset is allowed by either the standard asset class or the vendor defined asset, then access to the asset is granted and the request is processed. If access to the asset class is prohibited by both the standard asset class and a vendor defined asset class, then access to the asset is prohibited and the request is not processed.
   - How a request that attempts to access a prohibited asset is handled is Management Protocol specific.

If a request packet requires access to multiple assets for the request to be serviced, then Step 1 through Step 3 are performed and the request is processed only if access is granted to all assets. If access is prohibited to any asset associated with the request, then no asset is accessed by the request.

Figure 8-7. Access Control Determination in a Responder Management Entity



8.1.3.5.1 Standard Asset Classes

The objective of the standard asset classes is to provide a consistent classification of assets across chiplet implementations and applications for access control. To achieve this, it must be possible for a chiplet implementer to map chiplet assets that are accessible over the Management Network into asset classes without understanding the underlying architecture, roles, or applications associated with an SiP that uses the chiplet.

Standard asset class 0 is for SiP security configuration (e.g., reading and writing the RAC and WAC structures). The remaining standard asset classes are constructed by taking the Cartesian product of a set of asset types and asset contexts and removing elements that are not applicable in practice. Asset types used to construct the standard asset class are shown in Table 8-5 and asset contexts used to construct the standard asset class are shown in Table 8-6. The standard asset classes are shown in Table 8-7.

In some cases, an asset may logically map to two or more standard asset classes. For example, a memory region may contain both SiP data and chiplet data. When this occurs, the asset should be mapped to the standard asset class with the lowest ID value.

In some cases, further access control granularity may be desired beyond what is offered by the standard asset classes. This granularity may be accomplished by separating the assets into different Management Elements within the chiplet. For example, a chiplet may contain two global secrets and the chiplet implementor may wish to allow one set of requesters access to the first global secret and a separate set of requesters access to the second global secret. By putting the two global assets into two different Management Elements, different security clearance groups may be given access to each global secret. In another example, a chiplet may contain an I/O controller and a memory controller and the chiplet implementor may wish to allow one security clearance group to configure the I/O controller and a different security clearance group to configure the memory controller. This granularity may be achieved by putting the I/O controller and memory controller in different Management Elements.

Table 8-5. Asset Types

Asset Type	Description
Persistent One-Time Secret	Data or configuration that is persistent, one-time programmable, and considered a secret or sensitive configuration. Example: one-time programmable device secret
Permanent Secret	Data or configuration that is permanent and remains unchanged for the lifetime of the device and considered a secret or sensitive configuration. Example: a device secret that is permanent for the lifetime of the device
Secret	A secret, data, or status that may compromise a secret, or configuration that may control the exposure of a secret. Example: security key
Permanent Denial of Service (PDOS)	Data or configuration that could potentially cause permanent denial of service. Example: thermal, power, or voltage controls
Sensitive	Volatile or persistent data that should have limited exposure, status that could expose information that should have limited exposure, or configuration that may control exposure or modification of sensitive information. Examples: error injection capabilities, sensitive state machines, private memory space
Permanent	Data or configuration that is one-time programmable and is not sensitive or a secret. Example: general fuses
Data	General data or user data that is not sensitive or a secret. Example: application space
Configuration	General configuration that cannot be used to expose user, sensitive, or secret information.
Status	General status that cannot be used to expose user, sensitive, or secret information. Example: boot status

Table 8-6. Asset Contexts

Asset Context	Description
Global	Asset associated with or which affects chiplets or SiPs produced by a manufacturer. The definition of the manufacturer is beyond the scope of the specification and may be the SiP integrator, the SiP designer, or an IP provider. All that matters is that the asset is the same in SiPs of that type (i.e., a global key).
SiP	Asset associated with or which affects a specific SiP.
Chiplet	Asset associated with or which affects a specific chiplet.
Partition	Asset associated with or which affects a partition. The definition of a partition is vendor defined but is broadly defined as a collection of hardware resources that act as an independent unit.

Table 8-7. Standard Security Asset Classes (Sheet 1 of 2)

Standard Security Asset Class ID	Asset Context	Asset Type
0	SiP	SiP Security Configuration
1	Global	Secret
2	SiP	Persistent One-Time Secret
3	SiP	Secret
4	SiP	Permanent Denial of Service (PDOS)
5	SiP	Sensitive

Table 8-7. Standard Security Asset Classes (Sheet 2 of 2)

Standard Security Asset Class ID	Asset Context	Asset Type
6	SiP	Permanent
7	SiP	Data
8	SiP	Configuration
9	SiP	Status
10	Chiplet	Permanent Secret
11	Chiplet	Secret
12	Chiplet	Permanent Denial of Service (PDOS)
13	Chiplet	Sensitive
14	Chiplet	Permanent
15	Chiplet	Data
16	Chiplet	Configuration
17	Chiplet	Status
18	Partition	Permanent Secret
19	Partition	Secret
20	Partition	Permanent Denial of Service (PDOS)
21	Partition	Sensitive
22	Partition	Permanent
23	Partition	Data
24	Partition	Configuration
25	Partition	Status

8.1.3.5.2 Security Director

A Management Element within an SiP that may configure security parameters is designated as a Security Director. An SiP may contain multiple Security Directors. When an SiP contains multiple Security Directors, coordination between security directors is beyond the scope of this specification.

The Security Clearance Group value of 0 is reserved for Security Directors and must not be used for any other purpose.

The Management Director may also be a Security Director. While it is not recommended that the Management Director operate using the Security Clearance Group value reserved for Security Directors (i.e., 0) during normal operation, it is required to operate with this value during initial configuration. When and how a Management Director changes the Security Clearance Group used for transactions is beyond the scope of this specification.

8.1.3.6 Initialization and Configuration

UCIe Management Transport initialization and configuration are performed through read and write operations using the UCIe Memory Access protocol to Management Entity fields. Management Entity fields are grouped by function into Management Capability Structures.

Unless otherwise specified, Management Capability Structures and any sub-structures must be read or written as single DWORD quantities (i.e., the Length field in the UCIe Memory Access Request must be 0h which represents a data length of one DWORD). All fields in a Management Capability Structure and any sub-structures have little endian bit ordering.

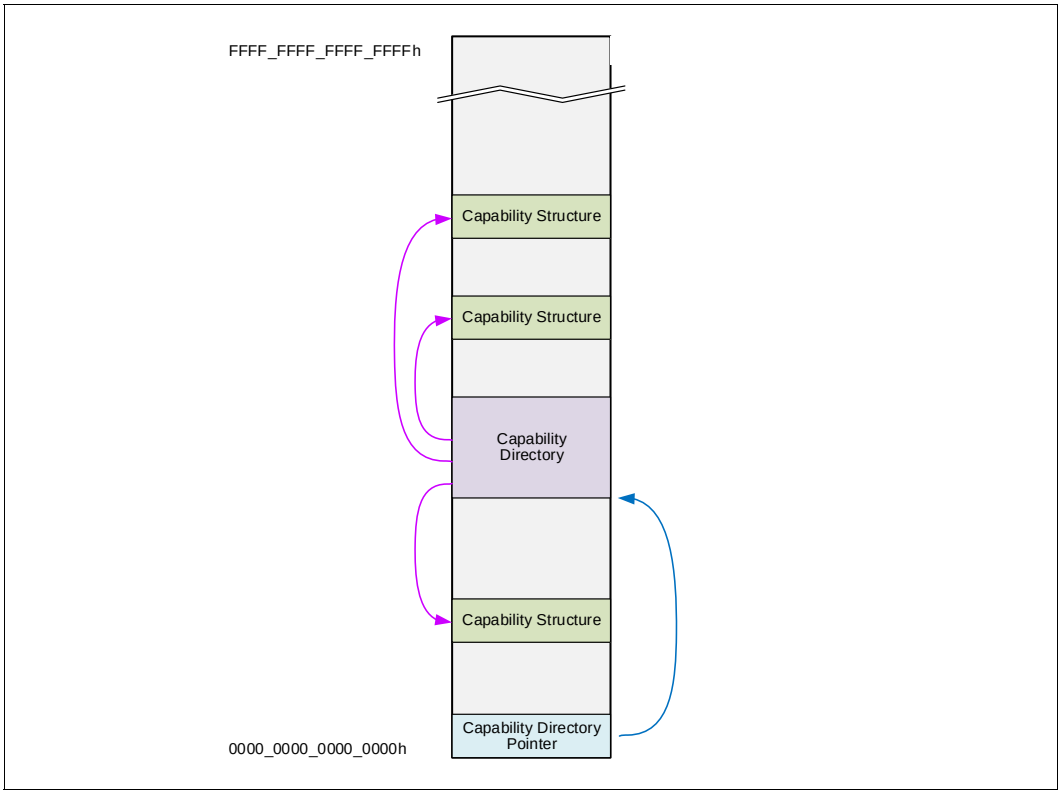
A Management Entity may support the UCIe Memory Access protocol (see Section 8.1.4) which exposes a 64-bit address space associated with the Management Entity containing fields that allow configuration by another Management Entity, such as the Management Director.

Each chiplet must support Management Element 0. Chiplet initialization and configuration are performed through Management Element 0 using the Chiplet Capability Structure and as a result Management Element 0 must support the UCIe Memory Access protocol. A chiplet may contain other Management Entities and the number of such Management Entities is implementation specific. These additional Management Entities may support the UCIe Memory Access protocol.

Figure 8-8 shows the UCIe Memory Access protocol memory map associated with a Management Entity that support the UCIe Memory Access Protocol. The contents and organization of the memory map are implementation specific except for a 64-bit Capability Directory Pointer value located at address 0. If the Management Entity implements any Management Capability Structures, then the Management Capability Directory Pointer contains the address of a Management Capability Directory. If the Management Entity does not implement any Management Capability Structures, then the Management Capability Directory Pointer contains a value of 0.

The Management Capability Directory, described in Section 8.1.3.6.1, contains a list of pointers (i.e., 64-bit UCIe Memory Access protocol addresses) to Management Capability Structures supported by the Management Entity and contains a pointer (i.e., the Element ID) of the next Management Entity in the chiplet if one exists.

Figure 8-8. Memory Map for Management Entities



The organization that all Management Capability Structures follow is shown in Figure 8-9. A Management Capability Structure is at least two DWORDs in size and may be larger. The size of a Management Capability Structure is Management Capability Structure specific. Associated with each Management Capability Structure is a Management Capability ID that identifies the capability. The list of Management Capability IDs defined by UCIe are listed in Table 8-8.

Figure 8-9. Management Capability Structure Organization

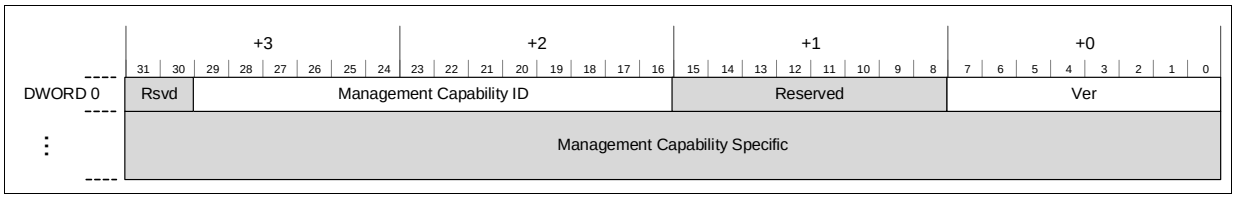
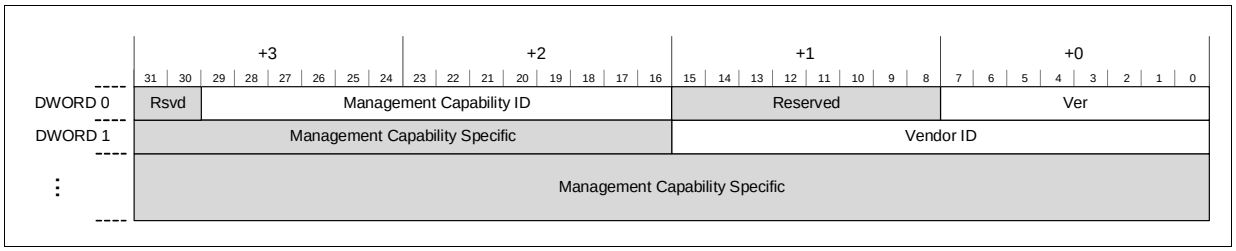


Table 8-8. UCIe-defined Management Capability IDs

Management Capability ID	Management Capability Structure Name	Description
0	Chiplet	See Section 8.1.3.6.2
1	Access Control	See Section 8.1.3.6.3
2	UCIe Memory Access Protocol	See Section 8.1.4.2
3	DFx Management Hub	See Section 8.3.1.1
4	Security Clearance Group	See Section 8.1.3.6.4
5	Open Drain Detection	See Section 8.1.6
6	Early Firmware Download	Section 8.4.1.2
7	Fast Throttle Trigger	See Section 8.4.2.1.2.1
8	Fast Throttle Response	See Section 8.4.2.1.2.2
9	Fast Throttle Logging	See Section 8.4.2.1.2.3
10	Emergency Shutdown Trigger	See Section 8.4.2.2.2.1
11	Emergency Shutdown Response	See Section 8.4.2.2.2.2
12	Emergency Shutdown Logging	See Section 8.4.2.2.2.3
13 to 12,287	Reserved	
12,288 to 16,383	Vendor defined	See Figure 8-10

The top 4096 Management Capability IDs are available for vendor-defined use. The organization of a Vendor Defined Management Capability Structure is shown in Figure 8-10. Bits 0 through 15 of DWORD 1 contain the UCIe-assigned identifier of the vendor that specified the Management Capability Structure.

Figure 8-10. Vendor Defined Management Capability Structure Organization



8.1.3.6.1 Management Capability Directory

The Management Capability Directory provides a method for discovery of Management Capability Structures associated with a Management Entity. The structure of the Management Capability Directory is shown in Figure 8-11 and described in Table 8-9.

Figure 8-11. Management Capability Directory

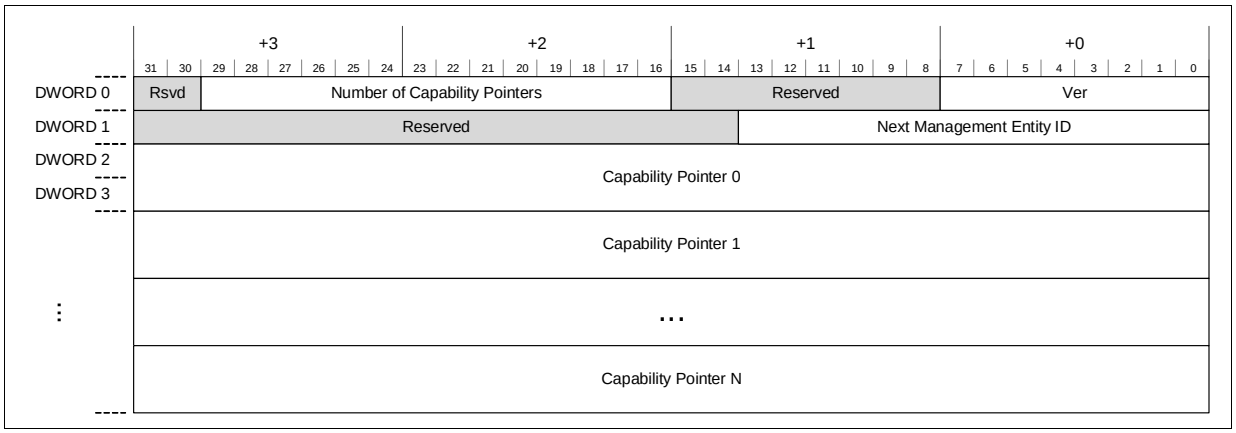


Table 8-9. Management Capability Directory Fields

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
Ver	0 [7:0]	17	RO	**Capability Directory Version** This field is a UCIe-defined version number that indicates the version of the capability directory. This field must be 00h in this version of the specification
Number of Capability Pointers	0 [29:16]	17	RO	**Number of Capability Pointers** This field indicates the number of Capability Pointers in the Capability Directory. Since the UCIe Memory Access Protocol must be supported in order to access the Management Capability Directory, this field cannot be zero.
Next Management Entity ID	1 [13:0]	17	RO	**Next Management Entity ID** Each chiplet contains a list of Management Entities that support the UCIe Memory Access Protocol starting with Management Element 0. This field contains the Management Entity ID of the next Management Entity in the chiplet that supports the UCIe Memory Access Protocol. If this is the last Management Entity in the chiplet that supports the UCIe Memory Access Protocol, then this field has a value of 0000h. Management Entity IDs in this list must be ordered from lowest to highest and may sparse (i.e., there may be gaps). The Management Entity list may be viewed as a list of Management Network IDs for Management Entities that support the UCIe Memory Access Protocol contained within the chiplet with the Chiplet ID field set to 0.

a. See Table 8-7 for a description of Standard Security Asset Class IDs.

Figure 8-12. Capability Pointer

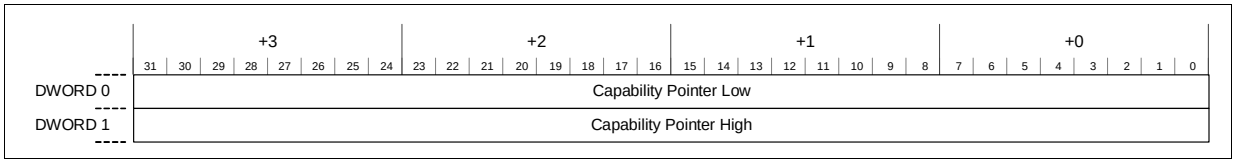


Table 8-10. Capability Pointer Fields

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
Capability Pointer Low	0 [31:0]	17	RO	**Capability Pointer Low** Bits 0 to 31 of the 64-bit address of the first byte of the Capability Structure associated with the Capability pointed to by this Capability Pointer. A value of all 0s in both the Capability Pointer Low and High fields indicates that this is a Null Capability Pointer and should be skipped. Because capability structures must be DWORD-aligned, bits 0 and 1 must be 00b.
Capability Pointer High	1 [31:0]	17	RO	**Capability Pointer High** Bits 32 to 63 of the 64-bit address of the first byte of the Capability Structure associated with the Capability pointed to by this Capability Pointer.

a. See Table 8-7 for a description of Standard Security Asset Class IDs.

8.1.3.6.2 Chiplet Capability Structure

The Chiplet Capability Structure must be implemented in Management Element 0 of each chiplet and must not be implemented in any other Management Entity in a chiplet.

Figure 8-13 shows the organization of the Chiplet Capability Structure. The Chiplet Capability Structure describes the basic characteristics of the chiplet. It points to a list of Management Port Structures that describe the characteristics of chiplet Management Ports. Each Management Port Structure contains one or more Route Entries that control the routing of UCIe Management Transport packets from the Management Fabric of the chiplet out the corresponding Management Port to an adjacent chiplet in the SiP.

Figure 8-13. Chiplet Capability Structure Organization

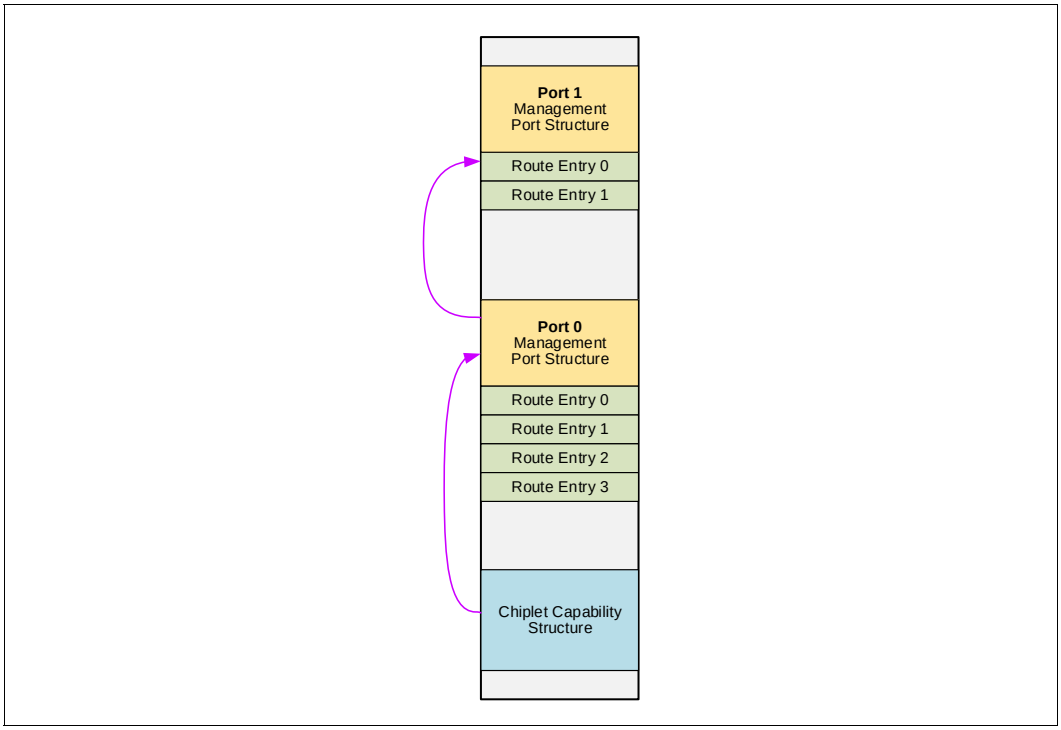


Figure 8-14. Chiplet Capability Structure

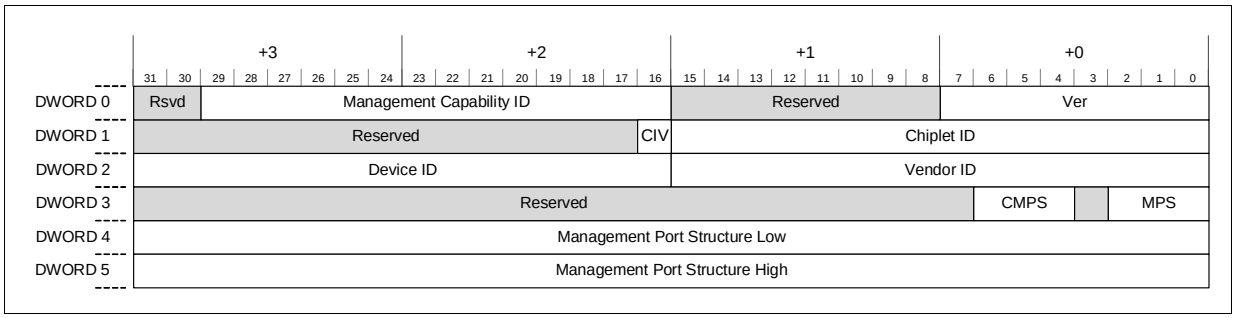


Table 8-11. Chiplet Capability Structure Fields (Sheet 1 of 2)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
Ver	0 [7:0]	17	RO	**Capability Structure Version** This field indicates the version of this capability structure. This field has a value of 00h in this specification.
Management Capability ID	0 [29:16]	17	RO	**Management Capability ID** This field specifies the Capability ID of this Management Capability structure. The Chiplet Capability structure has a Management Capability ID of 000h.
Chiplet ID	1 [15:0]	8	RW/RO	**Chiplet ID** This field is used to configure the Chiplet ID portion of the 16-bit Management Network ID associated with Management Element zero in the chiplet. A Management Network ID is partitioned into a Chiplet ID field in the upper bits and an Entity ID field in the lower bits (see Section 8.1.3.2). The lower bits of this field associated with the Entity ID portion of the Management Network ID are hardwired to 0 (i.e., RO). Since bits 0 and 1 are only associated with an Entity ID, they are always hardwired to zero. The upper bits of this field associated with the Chiplet ID portion of the Management Network ID may be read and written (i.e., RW). These upper bits must be initialized with the Chiplet ID value associated with the chiplet. The initial value of these upper bits is all ones (i.e., 1). The value of the Chiplet ID portion of the Management Network ID is used for UCIe Management Transport packet routing only when the Chiplet ID Valid (CIV) field is set to 1.
CIV	1 [16]	8	RW	**Chiplet ID Valid** When this bit is set to 1, the Chiplet ID value in the Chiplet ID field is used for UCIe Management Transport packet routing. When this bit is cleared to 0, the Chiplet ID value in the Chiplet ID field is uninitialized; see Section 8.1.3.3 for details on routing a UCIe Management Transport Packet with uninitialized Chiplet ID.
Vendor ID	2 [15:0]	17	RO	**Vendor ID** UCIe assigned identifier of the vendor that produced the chiplet.
Device ID	2 [31:16]	17	RO	**Device ID** Vendor assigned identifier that identifies the type of chiplet produced by that vendor. The tuple (Vendor ID, Device ID) uniquely identifies a type of chiplet.
MPS	3 [2:0]	17	RO	**Maximum Packet Size** This field indicates the maximum UCIe Management Transport packet size supported by the chiplet (see Section 8.1.3.1.2). 000b: 4 DWORDs 001b: 8 DWORDs 010b: 16 DWORDs 011b: 32 DWORDs 100b: 64 DWORDs 101b: 128 DWORDs 110b: 256 DWORDs 111b: 512 DWORDs

Table 8-11. Chiplet Capability Structure Fields (Sheet 2 of 2)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
CMPS	3 [6:4]	16	RW	**Configured Maximum Packet Size** This field indicates the configured maximum UCIe Management Transport packet size of the chiplet (see Section 8.1.3.1.2). The Configured Packet Size must be less than or equal to the Maximum Packet Size. Setting the Configured Packet Size to a value greater than the Maximum Packet Size blocks all Management Entities in the chiplet from emitting packets. Management Entities in the chiplet never generate UCIe Management Transport packets that are larger than the Configured Packet Size. This field has no effect on how a Management Entity handles receipt of a packet or transfer of packets on the Management Fabric. These behaviors are only affected by the Maximum Packet Size. The initial value of this field is 001b. 000b: 4 DWORDs 001b: 8 DWORDs 010b: 16 DWORDs 011b: 32 DWORDs 100b: 64 DWORDs 101b: 128 DWORDs 110b: 256 DWORDs 111b: 512 DWORDs
Management Port Structure Low	4 [31:0]	17	RO	**Management Port Structure** Bits 0 to 31 of the 64-bit address of the first Management Port Structure. Because the Management Port Structure must be DWORD-aligned, bits 0 and 1 must be 00b and are ignored. If the chiplet implements zero Management Ports, then this field must be 0.
Management Port Structure High	5 [31:0]	17	RO	**Management Port Structure** Bits 32 to 63 of the 64-bit address of the first Management Port Structure. If the chiplet implements zero Management Ports, then this field must be 0.

a. See Table 8-7 for a description of Standard Security Asset Class IDs.

8.1.3.6.2.1 Management Port Structure

The Management Port Structure provides a mechanism to discover and configure the characteristics of a chiplet Management Port. The structure contains Route Entries associated with the port and points to the next Management Port if one exists.

Figure 8-15. Management Port Structure

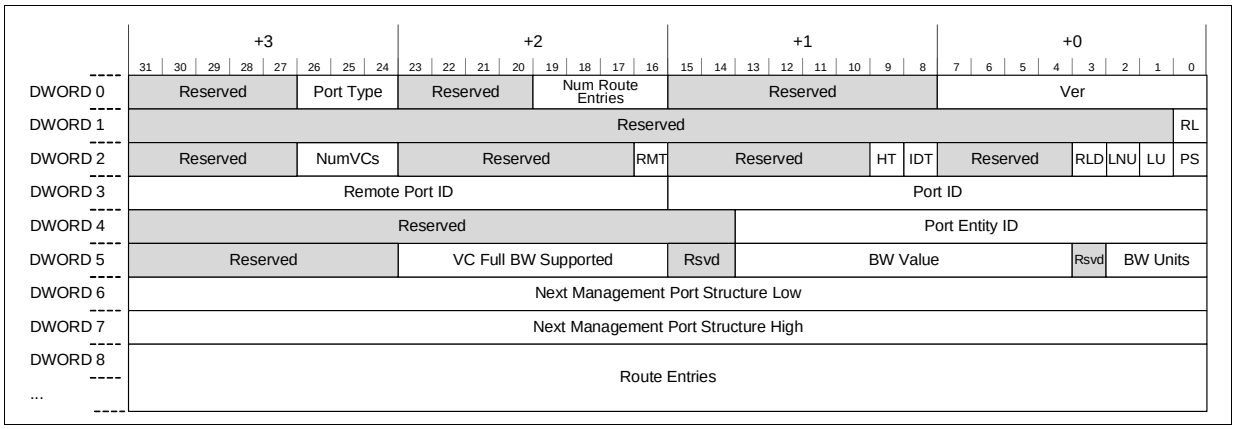


Table 8-12. Management Port Structure Fields (Sheet 1 of 5)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
Ver	0 [7:0]	17	RO	**Management Port Structure Version** This field indicates the version of the Management Port Structure. Must be 00h for this version of the specification.
Num Route Entries	0 [19:16]	17	RO	**Number of Route Entries** This field indicates the number of Route Entries associated with this Management Port. The number of Route Entries associated with the Management Port is equal to the value in this field plus 1. A Management Port must have at least one Route Entry associated with it. A value of 0h in this field indicates one Route Entry.
Port Type	0 [26:24]	17	RO	**Management Port Type** This field indicates the management port type. 000b: Not Implemented (skip) 001b: UCIe Sideband 010b: UCIe Mainband 111b: Vendor Defined Others: Reserved A value of 000b indicates that the management port is not implemented and should be skipped.

Table 8-12. Management Port Structure Fields (Sheet 2 of 5)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
RL	1 [0]	8	RW	**Retrain Link** Writing a 1 to this bit initiates retraining of the link associated with the Management Port. Because the UCIe Management Transport may be multiplexed with other protocols on the link, retraining a port link may affect SiP operation. Retraining a UCIe Sideband link will also retrain the corresponding UCIe Mainband link if one exists. Retraining a link may take time to complete after this bit is written. Status in this structure does not reflect the result of a link retraining until the operation completes. The Retrain Link Done (RLD) field may be used to determine when the operation has completed. Writing a 0 to this bit has no effect on the link. Reads of this field always return a value of 0 and have no effect on the link.
PS	2 [0]	17	RO	**Port Status** This field indicates the current Management Port Status. 0: Link Not Up 1: Link Up
LU	2 [1]	17	RW1C	**Link Up** This field indicates whether the link has transitioned to Link Up state since the last time this bit was cleared. The initial value of this field is 0. This bit is set to 1 when the link transitions from a link not up to a link up state. When this bit transitions to 1, this bit remains set to 1 until a 1 is written to this bit. Writing a 1 to this bit clears this bit to 0. Writing a 0 to this field has no effect on this field. Writing to this field has no effect on the link.
LNU	2 [2]	17	RW1C	**Link Not Up** This field indicates whether the link has transitioned to Link Not Up state since the last time this bit was cleared. The initial value of this field is 0. This bit is set to 1 when the link transitions from a link up to a link not up state. When this bit transitions to 1, this bit remains set to 1 until a 1 is written to this bit. Writing a 1 to this bit clears this bit to 0. Writing a 0 to this field has no effect on this field. Writing to this field has no effect on the link.
RLD	2 [3]	17	RW1C	**Retrain Link Done** This field indicates whether the link has completed the Link retraining since the last time this bit was cleared. The initial value of this field is 0. This bit is set to 1 when a 1 is written to the Retrain Link (RL) field and the corresponding retrain operation has completed. When this bit transitions to 1, this bit remains set to 1 until a 1 is written to this bit. Writing a 1 to this bit clears this bit to 0. Writing a 0 to this field has no effect on this field.

Table 8-12. Management Port Structure Fields (Sheet 3 of 5)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
IDT	2[8]	17	RW1C	**Init Done Timeout** This field indicates whether an Init Done Timeout was detected since the last time this bit was cleared. The initial value of this field is 0. This bit is set to 1 when an Init Done timeout is detected (see Section 8.2.4.4 for details). When this bit transitions to 1, this bit remains set to 1 until a 1 is written to this bit. Writing a 1 to this bit clears this bit to 0. Writing a 0 to this field has no effect on this field.
HT	2[9]	17	RW1C	**Heartbeat Timeout** This field indicates whether a Heartbeat Timeout was detected since the last time this bit was cleared. The initial value of this field is 0. This bit is set to 1 when a Heartbeat Timeout is detected (see Section 8.2.5.1.3 for details). When this bit transitions to 1, this bit remains set to 1 until a 1 is written to this bit. Writing a 1 to this bit clears this bit to 0. Writing a 0 to this field has no effect on this field. Heartbeat Timeout is implemented only on the UCIe sideband.
RMT	2[16]	17	RW1C	**Remote Management Transport** This field indicates whether the remote chiplet has advertised support for management transport on the associated Management Port since the last time this bit was cleared. The initial value of this field is 0. This bit is set to 1 when management transport support is advertised by the remote chiplet associated with this Management Port (see Section 4.5.3.3.1.1). When this bit transitions to 1, this bit remains set to 1 until a 1 is written to this bit. Writing a 1 to this bit clears this bit to 0. Writing a 0 to this field has no effect on this field.
Num VCs	2 [26:24]	17	RO	**Number of Virtual Channels** If the Port Status field indicates that the link is up, then the value of this field indicates the number of virtual channels available on the Management Port minus 1 (i.e., a value of 0 means one VC, a value of 1 means two VCs, and so on). Because implemented virtual channels must always start at 0 and increase sequentially. A value of N in this field indicates that Virtual Channels 0 through N are available. If the Port Status field indicates that the link is not up, then this field has a value of 000b.

Table 8-12. Management Port Structure Fields (Sheet 4 of 5)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
Port ID	3 [15:0]	17	RO	**Port Identifier** This field indicates the chiplet's unique 16-bit identifier associated with the corresponding Management Port. Port identifiers are statically assigned by the chiplet manufacturer, never change, and need not be assigned sequentially (i.e., their assignment may be sparse) except as outlined below. UCIe mainband and sideband ports associated with the same physical connection share all port ID bits in common except bit 0. Bit 0 has a value of 0 in the mainband port identifier and a value of 1 in the corresponding sideband port identifier. For example, if a UCIe mainband port has a port identifier of N, then N is even and the UCIe sideband port associated with that mainband port is odd and has a port identifier if (N+1). The port identifier FFFFh is reserved.
Remote Port ID	3 [31:16]	17	RO	**Remote Port Identifier** This field indicates the remote chiplet unique 16-bit port identifier associated with the remote Management Port (i.e., the Port ID of the adjacent chiplet). The value of this field is only valid when the Port Status field indicates that the link is up; otherwise, the value of this field is FFFFh. The value of this field is obtained from the remote chiplet during management transport path negotiation (see Section 4.5.3.3.1.1).
Port Entity ID	4 [13:0]	17	RO	**Port Entity ID** This field indicates the Entity ID associated with the Management Port (see Section 8.1.3.2).
BW Units	5 [2:0]	17	RO	**Port Bandwidth Units** If the Port Status field indicates that the link is up, then this field indicates the units associated with the BW Value field. Support for port bandwidth reporting is optional. 000b: Port bandwidth not reported 001b: KB/s 010b: MB/s 011b: GB/s 100b: TB/s Others: Reserved If the port Status field indicates that the link is not up or port bandwidth reporting is not supported, then this field has a value of 000b.
BW Value	5 [13:4]	17	RO	**Port Bandwidth Value** If the Port Status field indicates that the link is up, then this field indicates the maximum port bandwidth value. The units associated with the value are specified by the BW Units field. If the port bandwidth may change (e.g., because a chiplet port supports multiple link speeds), then this field reflects the current port bandwidth. Support for port bandwidth reporting is optional. If the port Status field indicates that the link is not up or port bandwidth reporting is not supported, then this field has a value of 000h.

Table 8-12. Management Port Structure Fields (Sheet 5 of 5)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
VC Full BW Supported	5 [23:16]	17	RO	**Virtual Channel Full Bandwidth Supported** Each bit in this field corresponds to a virtual channel. If the Port Status field indicates that the link is up and NUM VCs field indicates that a virtual channel is available, then a value of 1 in the bit corresponding to the virtual channel indicates that the virtual channel supports full link bandwidth reported by the BW Value field from the adjacent chiplet to this chiplet. A value of 0 in the bit corresponding to the virtual channel indicates that the virtual channel does not support full link bandwidth from the adjacent chiplet to this chiplet. Whether full link bandwidth is supported from this chiplet to the adjacent chiplet may be determined from the Management Port Structure in the adjacent chiplet. If the Port Status field indicates that the link is not up, then none of the virtual channels associated with the port are available. If a virtual channel is not available, then the value of the bit corresponding to the virtual channel has a value of 0.
Next Management Port Structure Low	6 [31:0]	17	RO	**Next Management Port Structure Low** Bits 0 to 31 of the 64-bit address of the first byte of the next Management Port Structure. Because Management Port Structures must be DWORD-aligned, bits 0 and 1 must be 00b. A value of all 0s in both the Management Port Structure Low and High fields indicates that there are no more Management Port Structures.
Next Management Port Structure High	7 [31:0]	17	RO	**Next Management Port Structure High** Bits 32 to 63 of the 64-bit address of the first byte of the Management Port Structure. A value of all 0s in both the Management Port Structure Low and High fields indicates that there are no more Management Port Structures.

a. See Table 8-7 for a description of Standard Security Asset Class IDs.

8.1.3.6.2.2 Route Entry

A Route Entry is used to specify a route from the Management Fabric within a chiplet out the Management Port associated with the Route Entry.

The TC Select field selects traffic classes that are filtered out from matching a Route Entry.

A Route Entry may specify a normal route or a default route. The type of route is determined by the RT field.

While the Chiplet ID and Entity ID size of chiplets may vary in an SiP, all Route Entry matching associated with a chiplet is performed using the Chiplet ID size of that chiplet.

Packet Route Entry matching is performed as follows.

- If a Route Entry has the Route Type field set to Normal Route, then a packet matches the Route Entry when all the following are true:
  - — The link is up,
  - — The packet is associated with a traffic class that has the corresponding bit of the TC Select field in the Route Entry set to 1,
  - — The Chiplet ID portion (using the Chiplet ID width for this chiplet) of the packet's Destination ID field is greater than or equal to the value in the Base ID field, and

  - The Chiplet ID portion (using the Chiplet ID width for this chiplet) of the packet's Destination ID field is less than or equal to the value in the Limit ID field.
- If a Route Entry has the Route Type field set to Default Route, then a packet matches the route when all the following are true:
  - The link is up,
  - The packet is associated with a traffic class that has the corresponding bit of the TC Select field in the Route Entry set to 1, and
  - The packet does not match any other Route Entry within the chiplet.

If a packet on the Management Fabric of a chiplet matches the route specified by the Route Entry, then the packet is transmitted out the Management Port associated with the Route Entry. The virtual channel used by the packet is specified by the VC ID field in the matching Route Entry.

Route Entries associated with the Management Ports on either side of a point-to-point link that interconnects two chiplets may be configured differently. This means that the TC-to-VC mapping in each direction on the link may be different.

Figure 8-16. Route Entry

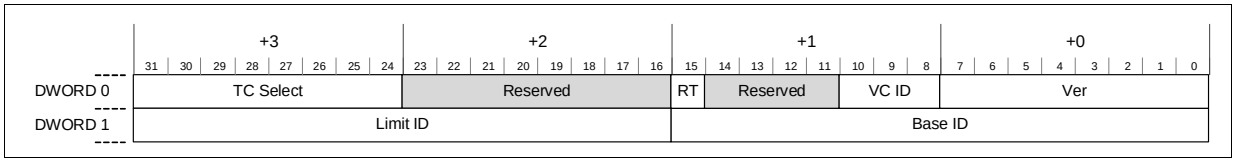


Table 8-13. Route Entry Fields (Sheet 1 of 2)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
Ver	0 [7:0]	8	RO	**Route Entry Version** This field indicates the version of the Route Entry. Must be 0h in this version of the specification
VC ID	0 [10:8]	8	RW	**Virtual Channel ID** This field selects the Virtual Channel (VC) used by packets that match this Route Entry. The default value of this field is 0 which maps all selected traffic classes onto VC0.
RT	0 [15]	8	RW	**Route Type** This field selects the routing type of this Route Entry. 0: Normal Route 1: Default Route The default value of this field is 0.

Table 8-13. Route Entry Fields (Sheet 2 of 2)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
TC Select	0 [31:24]	8	RW	**Traffic Class Select** This field selects which traffic classes may match this Route Entry. A packet's traffic class is specified Traffic Classes (TC) in the packet header. Each bit in this field corresponds to a traffic class (i.e., bit 0 corresponds to TC0, bit 1 to TC1, and so on). If a bit in this field is set to 0, then packets with the associated traffic class are filtered out from matching the route specified by the Route Entry. If a bit in this field is set to 1, then packets with the associated traffic class are considered for matching the route specified by the Route Entry. The default value of this field is 0 which filters out all traffic classes.
Base ID	1 [15:0]	8	RW/RO	**Base ID** This field contains the Base ID value of the Chiplet ID associated with this Route Entry. This field contains a 16-bit Management Network ID. The Management Network ID is partitioned into a Chiplet ID field in the upper bits and an Entity ID field in the lower bits (see Section 8.1.3.2). The lower bits of this field associated with the Entity ID portion of the Management Network ID are hardwired to 0 (i.e., RO). Since bits 0 and 1 are only associated with an Entity ID, they are always hardwired to zero. The upper bits of this field associated with the Chiplet ID portion of the Management Network ID may be read and written (i.e., RW). These upper bits must be initialized with the Chiplet ID value associated with the Base ID. The initial value of these upper bits is all ones (i.e., 1).
Limit ID	1 [31:16]	8	RW/RO	**Limit ID** This field contains the Limit ID value of the Chiplet ID associated with this Route Entry. This field contains a 16-bit Management Network ID. The Management Network ID is partitioned into a Chiplet ID field in the upper bits and an Entity ID field in the lower bits (see Section 8.1.3.2). The lower bits of this field associated with the Entity ID portion of the Management Network ID are hardwired to 0 (i.e., RO). The upper bits of this field associated with the Chiplet ID portion of the Management Network ID may be read and written (i.e., RW). These upper bits must be initialized with the Chiplet ID value associated with the Base ID. The initial value of these upper bits is all 0s. If the Base ID is greater than the Limit ID, then the Route Entry is disabled.

a. See Table 8-7 for a description of Standard Security Asset Class IDs.

8.1.3.6.3 Access Control Capability Structure

A Management Entity must support the access control mechanism outlined in Section 8.1.3.5 and must implement the Access Control Capability Structure described in this section. The Access Control Capability Structure provides access to the Read Access Control (RAC) and Write Access Control (WAC) structures associated with asset classes contained in the Management Entity.

The organization of the Access Control Capability Structure is shown in Figure 8-17. It consists of a 10-DWORD header that contains a pointer to the standard asset class access table and the vendor defined asset class access table.

The standard and vendor defined asset access class tables consists of a sequence of 128-bit (4-DWORD) RAC and 128-bit (4-DWORD) WAC structure pairs. The number of RAC/WAC structure pairs in the standard asset class access table is equal to 26 (i.e., the number of standard asset class IDs) and the number of RAC/WAC structure pairs in the vendor defined asset class access table corresponds to the number of vendor defined asset classes. The fields in the 4-DWORD RAC and WAC structures are described in Table 8-15 and Table 8-16, respectively.

Figure 8-17. Access Control Capability Structure

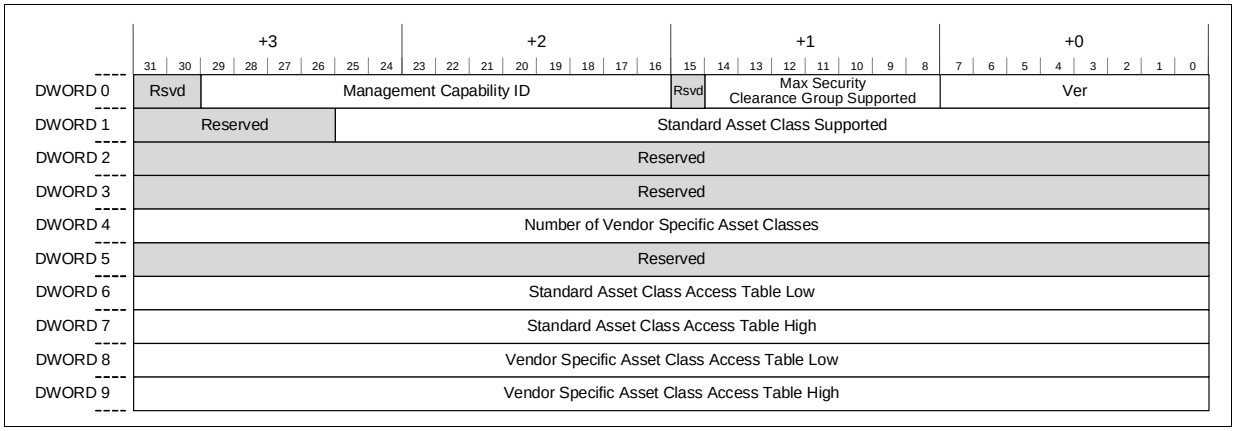


Table 8-14. Access Control Capability Structure Fields (Sheet 1 of 2)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
Ver	0 [7:0]	17	RO	**Capability Structure Version** This field indicates the version of this capability structure. This field has a value of 00h in this specification.
Max Security Clearance Group Supported	0 [14:8]	17	RO	**Max Security Clearance Group Supported** This field specifies the maximum security clearance group value supported. A value of N in this field indicates that security clearance group values 0 through N are supported. If this capability structure is implemented, then security clearance group 0 must be supported. If the number of security clearance groups is not restricted, then N is equal to 127.
Management Capability ID	0 [29:16]	17	RO	**Management Capability ID** This field specifies the Capability ID of this Management Capability structure. The Access Control Capability structure has a Management Capability ID of 001h.
Standard Asset Class Supported	1 [25:0]	17	RO	**Standard Asset Class Supported** Each bit in this field represents an asset class ID (see Table 8-5 and Table 8-7). If a bit in this field is set to 1, then the asset class corresponding to the asset class ID represented by the bit is supported. If a bit in this field is set to 0, then the asset class corresponding to the asset class ID represented by the bit is not supported.

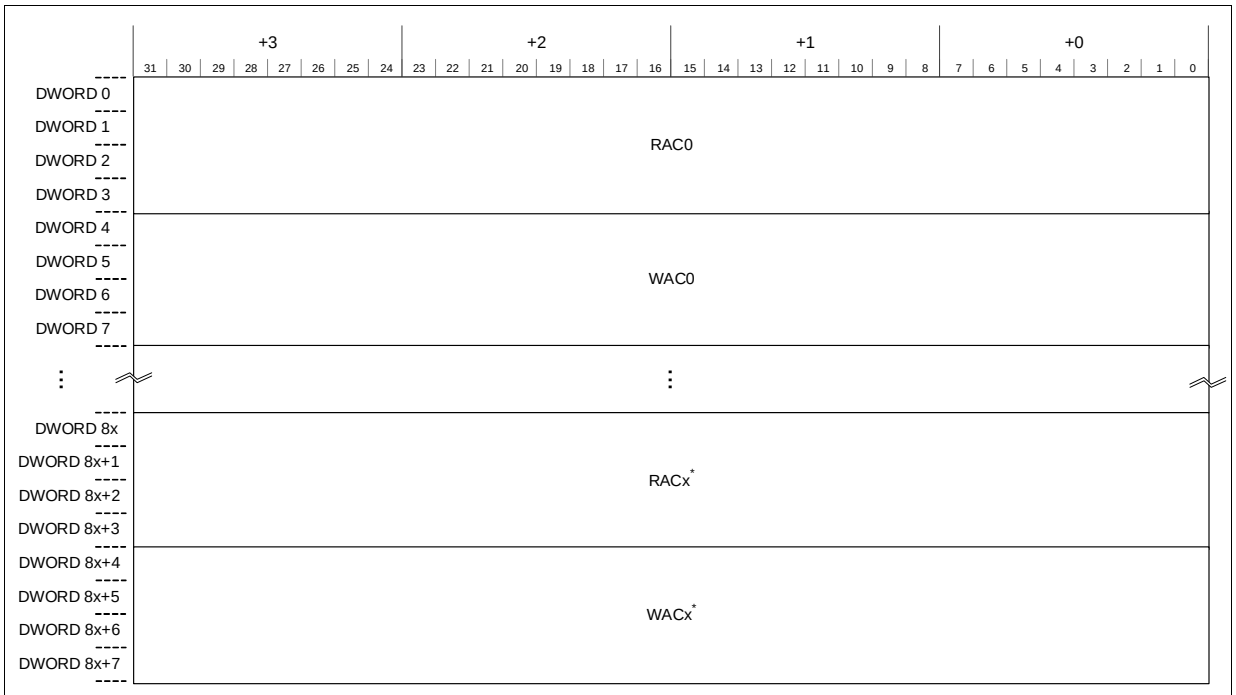
Table 8-14. Access Control Capability Structure Fields (Sheet 2 of 2)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
Number of Vendor Defined Asset Classes	4 [31:0]	17	RO	**Number of Vendor Defined Asset Classes** This field indicates the number of vendor defined asset classes. A value of all 0s in this field indicates that no vendor defined asset classes are supported.
Standard Asset Class Access Table Low	6 [31:0]	17	RO	**Standard Asset Class Access Table Low** Bits 0 to 31 of the 64-bit address of the base address of the Standard Asset Class Table. Because the Standard Asset Class Access Table must be DWORD-aligned, bits 0 and 1 must be 00b.
Standard Asset Class Access Table High	7 [31:0]	17	RO	**Standard Asset Class Access Table High** Bits 32 to 63 of the 64-bit address of the base address of the Standard Asset Class Table.
Vendor Defined Asset Class Access Table Low	8 [31:0]	17	RO	**Vendor Defined Asset Class Access Table Low** Bits 0 to 31 of the 64-bit address of the base address of the Vendor Defined Asset Class Table. Because the Vendor Defined Asset Class Access Table must be DWORD-aligned, bits 0 and 1 must be 00b. A value of zero in the Vendor Defined Asset Class Access Table Low and High fields indicates that there is no Vendor Defined Asset Class Access Table.
Vendor Defined Asset Class Access Table High	9 [31:0]	17	RO	**Vendor Defined Asset Class Access Table High** Bits 32 to 63 of the 64-bit address of the base address of the Vendor Defined Asset Class Table. A value of zero in the Vendor Defined Asset Class Access Table Low and High fields indicates that there is no Vendor Defined Asset Class Access Table.

a. See Table 8-7 for a description of Standard Security Asset Class IDs.

Figure 8-18. Standard Asset Class Access Table

Figure 8-19. Vendor Defined Asset Class Access Table



* The x in RACx/WACx corresponds to the number of vendor specific asset classes minus one (i.e., the first vendor specific asset class corresponds to RAC0/WAC0, the second vendor specific asset class corresponds to RAC1/WAC1, and so on).

Table 8-15. Read Access Control (RAC) Structure Field Description

Field Name	DWORD[a] & Bit Location	Standard Security Asset Class ID[b]	Attribute	Description
RACx_SD	8x [0]	0	RW/RO	**Read Access Control Security Director** This bit provides access control for the Security Director. If this bit is 1, then Security Director read accesses to assets in the Management Entity of the type associated with this RAC structure are allowed. If this bit is 0, then Security Director read accesses to assets in the Management Entity of the type associated with this RAC structure are not allowed. The initial value of this bit is 1. If the Management Entity contains no assets associated with the access class corresponding to this RAC structure, then this field is hardwired to 0.
RACx_LL	8x [31:1]	0	RW/RO	**Read Access Control Lower Lower** This field provides access control for Security Clearance Groups 1 through 31. Bit x corresponds to Security Clearance Group x. If a bit is 1, then read accesses with the corresponding clearance group to assets in the Management Entity of the type associated with this RAC structure are allowed. If this bit is 0, then read accesses with the corresponding clearance group to assets in the Management Entity of the type associated with this RAC structure are not allowed. The initial value of each bit in this field is 0. If the Management Entity contains no assets associated with the access class corresponding to this RAC structure, then this field is hardwired to 0. If the Management Entity does not support the security clearance group associated with a bit in this field, then the bit is hardwired to 0.
RACx_LM	8x+1 [31:0]	0	RW/RO	**Read Access Control Lower Middle** This field provides access control for Security Clearance Groups 32 through 63. Bit x corresponds to Security Clearance Group (x + 32). See RACx_LL for a description of this field. The initial value of each bit in this field is 0.
RACx_UM	8x+2 [31:0]	0	RW/RO	**Read Access Control Upper Middle** This field provides access control for Security Clearance Groups 64 through 95. Bit x corresponds to Security Clearance Group (x + 64). See RACx_LL for a description of this field. The initial value of each bit in this field is 0.
RACx_UU	8x+3 [31:0]	0	RW/RO	**Read Access Control Upper Upper** This field provides access control for Security Clearance Groups 96 through 127. Bit x corresponds to Security Clearance Group (x + 96). See RACx_LL for a description of this field. The initial value of each bit in this field is 0.

a. DWORD in this table refers to the DWORD offset into asset class access table for RACx. For example, the 128-bit RAC2 structure is at DWORD offsets 16, 17, 18, and 19.
b. See Table 8-7 for a description of Standard Security Asset Class IDs.

Table 8-16. Write Access Control (WAC) Structure Field Description

Field Name	DWORD[a] & Bit Location	Standard Security Asset Class ID[b]	Attribute	Description
WACx_SD	8x+4 [0]	0	RW	**Write Access Control Security Director** This bit provides access control for the Security Director. If this bit is 1, then Security Director write accesses to assets in the Management Entity of the type associated with this WAC structure are allowed. If this bit is 0, then Security Director write accesses to assets in the Management Entity of the type associated with this WAC structure are not allowed. The initial value of this bit is 1. If the Management Entity contains no assets associated with the access class corresponding to this WAC structure, then this field is hardwired to 0.
WACx_LL	8x+4 [31:1]	0	RW	**Write Access Control Lower Lower** This field provides access control for Security Clearance Groups 1 through 31. Bit x corresponds to Security Clearance Group x. If a bit is 1, then write accesses with the corresponding clearance group to assets in the Management Entity of the type associated with this WAC structure are allowed. If this bit is 0, then write accesses with the corresponding clearance group to assets in the Management Entity of the type associated with this WAC structure are not allowed. The initial value of each bit in this field is 0. If the Management Entity contains no assets associated with the access class corresponding to this WAC structure, then this field is hardwired to 0. If the Management Entity does not support the security clearance group associated with a bit in this field, then the bit is hardwired to 0.
WACx_LM	8x+5 [31:0]	0	RW	**Write Access Control Lower Middle** This field provides access control for Security Clearance Groups 32 through 63. Bit x corresponds to Security Clearance Group (x + 32). See WACx_LL for a description of this field. The initial value of each bit in this field is 0.
WACx_UM	8x+6 [31:0]	0	RW	**Write Access Control Upper Middle** This field provides access control for Security Clearance Groups 64 through 95. Bit x corresponds to Security Clearance Group (x + 64). See WACx_LL for a description of this field. The initial value of each bit in this field is 0.
WACx_UU	8x+7 [31:0]	0	RW	**Write Access Control Upper Upper** This field provides access control for Security Clearance Groups 96 through 127. Bit x corresponds to Security Clearance Group (x + 96). See WACx_LL for a description of this field. The initial value of each bit in this field is 0.

a. DWORD in this table refers to the DWORD offset into asset class access table for WACx. For example, the 128-bit WAC2 structure is at DWORD offsets 20, 21, 22, and 23.
b. See Table 8-7 for a description of Standard Security Asset Class IDs.

8.1.3.6.4 Security Clearance Group Capability Structure

The Security Clearance Group Capability Structure allows a Security Director to configure the Security Clearance Group value used by a Management Entity when issuing Management Transport requests. This capability structure must be implemented by a Management Entity that is the ultimate source of UCIe Management Transport request packets (i.e., emits request packets) and is not required to be implemented by any other Management Entity.

In some cases, a Management Entity may need to issue requests with different Security Clearance Group values when operating in different contexts. The Security Clearance Group Capability Structure supports multiple Security Clearance Group Contexts to allow a Security Director to configure a Security Clearance Group value for each context. How a Security Director determines the manageability functions provided by these contexts and what Security Clearance Group value should be used is beyond the scope of this specification.

Figure 8-20. Security Clearance Group Capability Structure

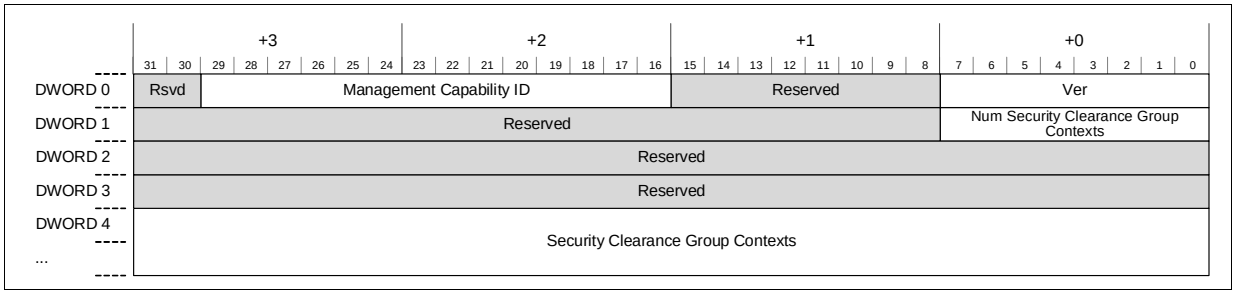


Table 8-17. Security Clearance Group Capability Structure Fields

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
Ver	0 [7:0]	17	RO	**Capability Structure Version** This field indicates the version of this capability structure. This field has a value of 00h in this specification.
Management Capability ID	0 [29:16]	17	RO	**Management Capability ID** This field specifies the Capability ID of this Management Capability structure. The Security Clearance Group Capability structure has a Management Capability ID of 004h.
Num Security Clearance Group Contexts	1 [7:0]	17	RO	**Number of Security Clearance Group Contexts** This field indicates the number of Security Clearance Group Contexts associated with this Management Entity. The number of Security Clearance Groups Contexts associated with this Management Entity is equal to the value in this field plus 1. A Management Entity that implements this capability structure must have at least one Security Clearance Group Context.

a. See Table 8-7 for a description of Standard Security Asset Class IDs.

8.1.3.6.4.1 Security Clearance Group Context

Figure 8-21. Security Clearance Group Context

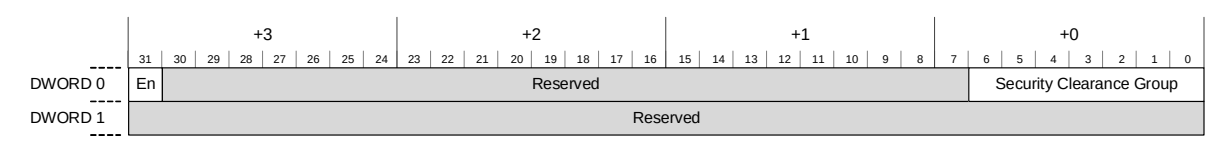


Table 8-18. Security Clearance Group Context Fields

Field Name	DWORD& Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
Security Clearance Group	0 [6:0]	0	RW	**Security Clearance Group** This field is configured by the Security Director with the Security Clearance Group value used by the Management Entity when issuing a request. The initial value of this field is 7Fh if this context is not associated with a Security Director. The initial value of this field is 00h if this context is associated with a Security Director.
En	0 [31]	0	RW	**Request Enable** When this field is set to 1 the Management Entity may issue requests associated with this security clearance group context. When this field is set to 0 the Management Entity must not issue requests associated with this security clearance group context. The initial value of this field is 0 if this context is not associated with a Security Director. The initial value of this field is 1 if this context is associated with a Security Director.

a. See Table 8-7 for a description of Standard Security Asset Class IDs.

8.1.4 UCIe Memory Access Protocol

The UCIe Memory Access Protocol provides read and write access to memory mapped structures and memory associated with a Management Entity that supports the UCIe Memory Access Protocol. A Management Entity exposes a 64-bit address space. The relationship of this address space to a system or I/O address map is beyond the scope of this specification.

The address space associated with a Management Entity may be local to that Management Entity or shared across one or more Management Entities in a chiplet. For example, the same address in two Management Entities may reference the same memory location or different memory locations (e.g., a memory location associated with each Management Entity). A Management Entity may have some addresses that are local and some that are shared. For shared addresses, how concurrent accesses, security, and mutual exclusion are handled is beyond the scope of this specification.

The UCIe Memory Access Protocol utilizes the UCIe Management Transport access control mechanism (see Section 8.1.3.5).

8.1.4.1 UCIe Memory Access Protocol Packets

This section describes UCIe Memory Access Protocol packets. These packets are carried by the UCIe management transport.

8.1.4.1.1 UCIe Memory Request Packet

Memory request packets are issued by a Management Entity to read or write memory mapped structures or memory in another Management entity. The Opcode field indicates the type of operation. When a UCIe Management Transport packet carries a UCIe Memory Request, the Resp field is set to 0 corresponding to a request packet.

UCIe Memory Request packet operations are non-posted. If a UCIe Management Transport packet that carries a UCIe Memory Request packet is not discarded, then a UCIe Memory Response packet is sent in response.

A Management Entity may issue requests on an ordered or unordered traffic class when the Unordered Traffic Class Enable (UE) bit is set to 1 in the UCIe Memory Access Protocol capability structure. When the UE bit is cleared to 0, then the Management Entity may only issue requests on an ordered traffic class and must not issue requests on an unordered traffic class. Whether a Management Entity utilizes an unordered traffic class is implementation specific.

The Tag field in a UCIe Memory Request packet is an 8-bit field populated by the requester, carried in a request packet, and returned by the responder in the corresponding response packet if one is generated. A requester may have multiple outstanding requests with the same Tag field value to the same or different responders. The responder must not assume that Tag field values are unique and must not in any way interpret the Tag field value. The use of the Tag field is requester implementation specific and may be used for applications such as mapping responses to previously issued requests; determining the responder associated with a response packet; and detecting lost, dropped, or discarded packets.

The maximum number of requests that a requester may have outstanding is requester implementation specific.

Figure 8-22 shows the fields of a UCIe Memory Access Request packet. Reserved fields (i.e., ones labeled as Rsvd) must be filled with all 0s when the packet is formed. Reserved fields must be forwarded unmodified on the Management Network and ignored by receivers. An implementation that relies on the value of a reserved field in a packet is non-compliant.

Figure 8-22. UCIe Memory Access Request Packet Format

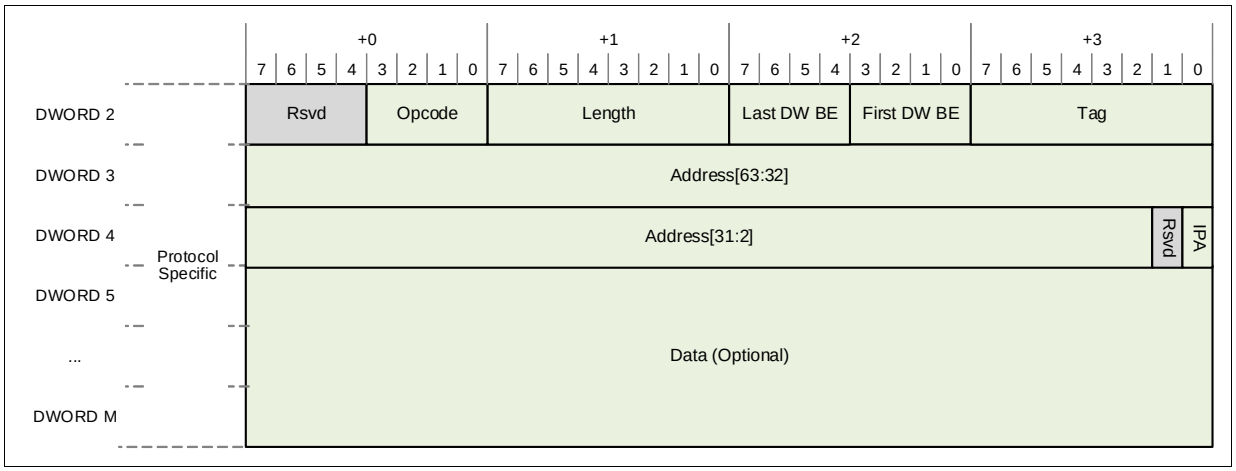


Table 8-19 defines the fields of a UCIe Memory Access Request packet. The packet starts at DWORD 2 because DWORDs 0 and 1 contain the UCIe Management Transport packet header. All fields in the table have little endian bit ordering, similar to Figure 8-5 (e.g., Address bits 32 through 39 are in Byte 3 bits 7 through 0 of DWORD 3, and Address bits 40 through 47 are in Byte 2 bits 7 through 0 of DWORD 2 and so on).

Table 8-19. UCIe Memory Access Request Packet Fields

Field Name	Field Size	Description
Tag	8 bits	**Tag** This field contains the value of the tag field of the corresponding memory access request.
First DW BE	4 bits	**First Data DWORD Byte Enable** This field contains byte enables for the first (or only) DWORD referenced. A value of 0 indicates that the corresponding byte is not enabled and a value of 1 indicates that the corresponding byte is enabled. Bit 0 corresponds to Byte 0. Bit 1 corresponds to Byte 1. Bit 2 corresponds to Byte 2. Bit 3 corresponds to Byte 3.
Last DW BE	4 bits	**Last Data DWORD Byte Enable** This field contains byte enables for the last DWORD referenced. If the LENGTH field has a value of 0, then this field must be 0000b. A value of 0 indicates that the corresponding byte is not enabled and a value of 1 indicates that the corresponding byte is enabled. Bit 0 corresponds to Byte 0. Bit 1 corresponds to Byte 1. Bit 2 corresponds to Byte 2. Bit 3 corresponds to Byte 3.
Length	8 bits	**Data Length** This field indicates the length of data referenced in DWORDs. The length of the packet in DWORDs is equal to the value of this field plus 1 (e.g., a value of 00h in this field indicates a packet length of one DWORD, a value of 01h in this field indicates a packet length of two DWORDs, and so on).
Opcode	4 bits	**Opcode** This field indicates the memory access request operation. 0000b: Reserved (used for responses) 0001b: Memory Read (MemRd) 0010b: Memory Write (MemWr) Others: Reserved
Address	62 bits	**Address** This field contains the DWORD address being referenced in the Management Entity.
IPA	1 bit	**Ignore Prohibited Access** This bit indicates whether accesses to prohibited assets should be ignored. When access to a prohibited asset is ignored, the asset is not accessed but the request completes successfully. The value of this bit is determined by the requester and should not be set to 1 during normal operation because doing so would cause access violations to not be reported in the Response Status. 0: Do not ignore access to prohibited assets 1: Ignore accesses to prohibited assets
Data	Varies	**Data** This field is present in Memory Write requests and contains the data being written. This field is not present in Memory Read requests.

8.1.4.1.2 UCIe Memory Access Response Packet

A UCIe Memory Access Response packet is generated by a Management Entity when the processing associated with a UCIe Memory Access Request packet completes. When a UCIe Management Transport packet carries a UCIe Memory Response, the Resp field is set to 1 corresponding to a response packet.

A UCIe Memory Access Protocol responder must always support UCIe Memory Request packets on all traffic classes (TC). The traffic class of a UCIe Memory Access Response packet is the same as the traffic class used in the corresponding UCIe Memory Access Request packet.

As described in Section 8.1.3.1.1, each traffic class is a unique ordering domain. There are no ordering guarantees for UCIe Memory Request packets in different traffic classes.

Within an ordered traffic class, UCIe Memory Request packets are delivered in-order between a requester and a responder and UCIe Memory Response packets are delivered in-order between a responder and the requester. There are no ordering guarantees between requests to different responders and there are no ordering guarantees between responses from different responders to a requester. Within an unordered traffic class there are no packet ordering guarantees and the packets may be delivered in any order.

A Management Entity may process received UCIe Memory Request packets sequentially (i.e., one at a time) or concurrently (i.e., two or more at a time). There are no ordering requirements between requests in different traffic classes; however, the result of processing these requests must be equivalent to some sequential processing of requests performed in an atomic manner.

Regardless of whether UCIe Memory Request packets are associated with an ordered or an unordered traffic class, a responder may send UCIe Memory Access Response Packets out-of-order (i.e., a responder is not required to send response packets in the same order that the corresponding request packets were received by the responder). This means that responses may be received by a requester in an order different from the order in which the requests were sent by the requester.

> **IMPLEMENTATION NOTE**
>
> **Responses in an Ordered Traffic Class**
>
> Because responders are free to send response packets for the UCIe Memory Access protocol in any order, an implementation may reorder these responses when carried in an ordered traffic class. While this conflicts with the ordered traffic class requirements, a requester cannot tell whether this reordering occurred at the responder or in the ordered traffic class of the Management Network.

The Status field in a UCIe Memory Access Response packet indicates the status associated with processing the corresponding UCIe Memory Access Request packet. If a UCIe Memory Access Request packet is processed successfully, then a UCIe Memory Access Response packet is generated with status Success. If the request requires response data, then all the data associated with the successful response is contained in a single response packet.

If a Management Entity receives a well formed UCIe Management Transport packet, but the UCIe Memory Access Request packet is malformed, then no processing of the request occurs and a response with no data and status Packet Error is returned.

- Examples of a malformed UCIe Memory Access Request packet:
  - — Receipt of a UCIe Memory Access Request packet with a reserved value in the Opcode field.
  - — Receipt of a UCIe Memory Access Request packet with the Length field set to zero and the Last DW BE field set to a nonzero value.

If a request violates the programming model of a Management Entity, then the request is not performed and a response with no data and status Programming Model Violation is returned.

- Examples of programming model violations:
  - — Unless otherwise specified all UCIe defined structures must be accessed as DWORDs.

If a Management Entity receives a request and is not capable of processing the request, but will be able to process the request at some point in the future, then a response with no data and status Retry Request is returned. The Retry Request status should not be used during normal operation and implementations are strongly encouraged to only use the Retry Status when absolutely necessary. How long a requester waits after receiving a response with status Retry Request before reissuing the request is implementation specific. The Max Retry Time Units and Max Retry Time Value fields in the UCIe Memory Access Protocol Capability Structure report the maximum duration of time during which a Management Entity may return a response with status Retry Request. A requester may use this time duration to determine how long to poll a responder before declaring that the responder has malfunctioned.

If the Management Entity can process a request, the request does not contain an error, and the request attempts to access an asset that is prohibited, then the asset is not accessed, and no processing associated with the request occurs.

- If the Ignore Prohibited Access (IPA) bit in the request is cleared to 0, then a response with no data and a status of Access Denied is returned.
- If the Ignore Prohibited Access (IPA) bit in the request is set to 1, then the required response data with all values set to zero and status Success is returned. The purpose of this is to allow an address range to be probed without returning errors.

The read of a byte whose corresponding byte enable is 0 in the First DW BE or Last DW BE field should return a value of FFh.

> **IMPLEMENTATION NOTE**
>
> **Read Value Returned on Unused Byte Lanes**
>
> If a Management Entity receives a UCIe Memory Access Request packet with a byte enable value of 0 in the First DW BE or Last DW BE field and does not return a value of FFh for the byte in the corresponding response, then care must be exercised to ensure that the data returned in unused bytes does not create a security issue. Implementations are strongly encouraged to align secure information on DWORD or larger boundaries.

Figure 8-23 shows the fields of a UCIe Memory Access Response packet. Reserved fields (i.e., ones labeled as Rsvd) must be filled with 0s when the packet is formed. Reserved fields must be forwarded unmodified on the Management Network and ignored by receivers. An implementation that relies on the value of a reserved field in a packet is non-compliant.

Figure 8-23. UCIe Memory Access Response Packet

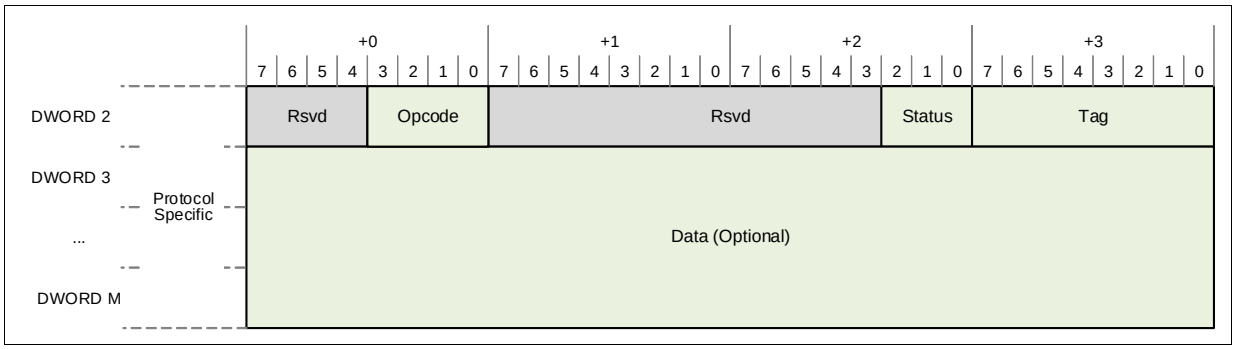


Table 8-20 defines the fields of a UCIe Memory Access Response packet. The packet starts at DWORD 2 because DWORDs 0 and 1 contain the UCIe Management Transport packet header. All fields in the table have little endian bit ordering (e.g., Tag bit 0 is in DWORD 3 bit 0, and Tag bit 7 is in DWORD 3 bit 7).

Table 8-20. UCIe Memory Access Response Packet Fields

Field Name	Field Size	Description
Opcode	4 bits	**Opcode** This field must be set to 0000b.
Status	3 bits	**Response Status** This field indicates the memory access response status. 000b: Success (SUCCESS) 001b: Programming Model Violation (PMV) 010b: Retry Request (RR) 011b: Access Denied (AD) 100b: Packet Error (PERR) Others: Reserved
Tag	8 bits	**Tag** This field contains the value of the tag field of the corresponding memory access request.
Data	Varies	**Data** If the memory access request was a Memory Read that was processed successfully (i.e., the Response Status field contains Success), then this field contains the data read. This field is not present in Memory Write completions.

8.1.4.2 UCIe Memory Access Protocol Capability Structure

A Management Entity that implements the UCIe Memory Access Protocol must implement the UCIe Memory Access Protocol Capability Structure described in this section.

The Max Buffered Requests field reports the maximum number of requests that the Management Entity is guaranteed to buffer. Issuing more outstanding requests to the Management Entity than this maximum may result in head-of-line blocking in the chiplet Management Fabric and/or a VC associated with a Management Port between chiplets.

The Request Buffer Size field reports the sum of the size of the requests that the Management Entity is guaranteed to buffer. Issuing more outstanding requests to the Management Entity than will fit in this buffer may result in head-of-line blocking in the chiplet Management Fabric and/or a VC associated with a Management Port between chiplets.

The Max Response Time Units and Max Response Time Value fields report the expected maximum time that the Management Entity requires to process a request. This is the expected maximum time with no other outstanding requests from receipt of a UCIe Memory Request packet at the Management Entity to the Management Entity emitting a corresponding UCIe Memory Response packet.

The UCIe Management Access Protocol does not define an architected completion timeout mechanism to detect lost packets or hardware failures; however, a requester may use the time reported in Max Response Time Units and Max Response Time Value fields in this capability structure to implement a vendor defined completion timeout mechanism. When a completion timeout mechanism is implemented, the requester must not declare a completion timeout sooner than the expected maximum response time reported by Response Time Units and Response Time Value fields.

Figure 8-24. UCIe Memory Access Protocol Capability Structure

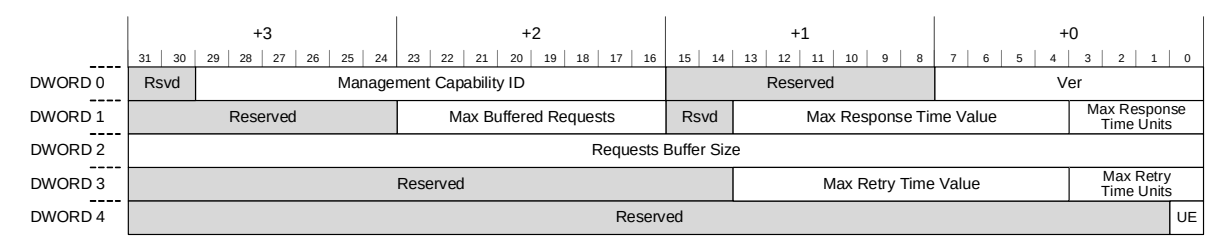


Table 8-21. UCIe Memory Access Protocol Capability Structure Fields (Sheet 1 of 2)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
Ver	0 [7:0]	17	RO	**Capability Structure Version** This field indicates the version of this capability structure. This field has a value of 00h in this specification.
Management Capability ID	0 [29:16]	17	RO	**Management Capability ID** This field specifies the Capability ID of this Management Capability structure. The UCIe Memory Access Protocol Capability structure has a Management Capability ID of 002h.
Max Response Time Units	1 [3:0]	17	RO	**Maximum Response Time Units** This field indicates the units associated with the Max Response Time Value field. 0000b: Reserved 0001b: nanoseconds (ns) 0010b: microseconds (us) 0011b: milliseconds (ms) 0100b: seconds (s) Others: Reserved

Table 8-21. UCIe Memory Access Protocol Capability Structure Fields (Sheet 2 of 2)

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
Max Response Time Value	1 [13:4]	17	RO	**Maximum Response Time Value** This field indicates the expected maximum response time value. The units associated with this value are specified by the Max Response Time Units field.
Maximum Buffered Requests	1 [23:16]	17	RO	**Maximum Number of Buffered Requests** This field reports the maximum number of requests that the Management Entity can buffer. Requests that the Management Entity are currently processing are considered buffered request. A value of zero in this field indicates that the maximum number of buffered requests is not reported.
Request Buffer Size	2 [31:0]	17	RO	**Request Buffer Size** This field reports the size of the Management Entity request buffer in DWORDs. The number of DWORDs consumed by a request packet in this buffer is equal to the value specified by the Length field in the UCIe Management Transport packet header. A request packet that the Management Entity is currently processing consumes space in this buffer. A value of all 0s in this field indicates that the size of the request buffer is not reported.
Max Retry Time Units	3 [3:0]	17	RO	**Maximum Retry Time Units** This field indicates the units associated with the Max Retry Time Value field. 0000b: Reserved 0001b: nanoseconds (ns) 0010b: microseconds (us) 0011b: milliseconds (ms) 0100b: seconds (s) Others: Reserved
Max Retry Time Value	3 [13:4]	17	RO	**Maximum Retry Time Value** This field indicates the maximum duration of time during which a Management Entity may return a response with status Retry Request (i.e., maximum retry time). A value of 000h in this field indicates that the maximum retry time value is not reported.
UE	4 [0]	16	RW	**Unordered Traffic Class Enable** When this field is set to 1 the Management Entity may issue UCIe Memory Access protocol requests on an ordered or unordered traffic class. When this field is set to 0 the Management Entity may only issue requests on an ordered traffic class and must not issue requests on an unordered traffic class. The initial value of this field is 0 in all Management Entities except the Management Director. The initial value of this field is 1 in the Management Director.

a. See Table 8-7 for a description of Standard Security Asset Class IDs.

8.1.5 Common Data Structures

All data structures described in this section are common and can be instantiated in any capability structure exposed through the UCIe memory access protocol (see Section 8.4). A capability structure that instantiates one of the common data structures must follow the memory layout and size for each field defined as part of the common data structure. A capability structure must support all the mandatory features for the common data structure(s) it is using.

In the following sections, a UMAP requester is an entity which accesses the common data structure through the UMAP protocol. The restrictions and rules for each common data structure defined in Section 8.1.5.1 apply to UMAP requesters. Access or manipulation of the common data structure locally by the chiplet/endpoint is beyond the scope of this specification.

8.1.5.1 Circular Buffer

A Circular Buffer structure is a common data structure that can be instantiated by a capability structure provided it respects the layout and behavior outlined in this section. A capability structure can instantiate multiple distinct instances of the Circular Buffer. Section 8.1.5.1.11 describes how to use the Circular Buffer structure.

This section defines both sink and source type of Circular Buffers that both follow similar memory layout and behavior. As a sink the UMAP requester writes to the Circular Buffer. As a source the UMAP requester reads from the Circular Buffer. The sink and source Circular Buffers share the same layout and semantics.

8.1.5.1.1 Circular Buffer Theory of Operation

A Circular Buffer is a common structure that a producer writes to at a write offset and consumer reads from at a read offset. Both offsets range from 0 to the Circular Buffer size minus 1. Both offsets wrap-around to 0 when they reach the Circular Buffer size value. The Circular Buffer is empty if the write offset and read offset are equal as shown in Figure 8-25. The Circular Buffer is full if the write offset incremented by 1 would be equal to the read offset as shown in Figure 8-26 or Figure 8-27.

Figure 8-25. Empty Circular Buffer

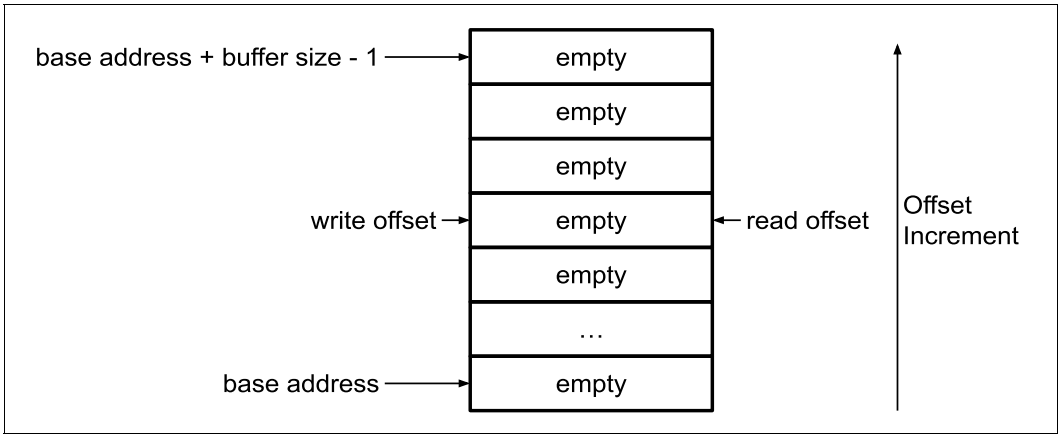


Figure 8-26. Full Circular Buffer with Write Offset below Read Offset

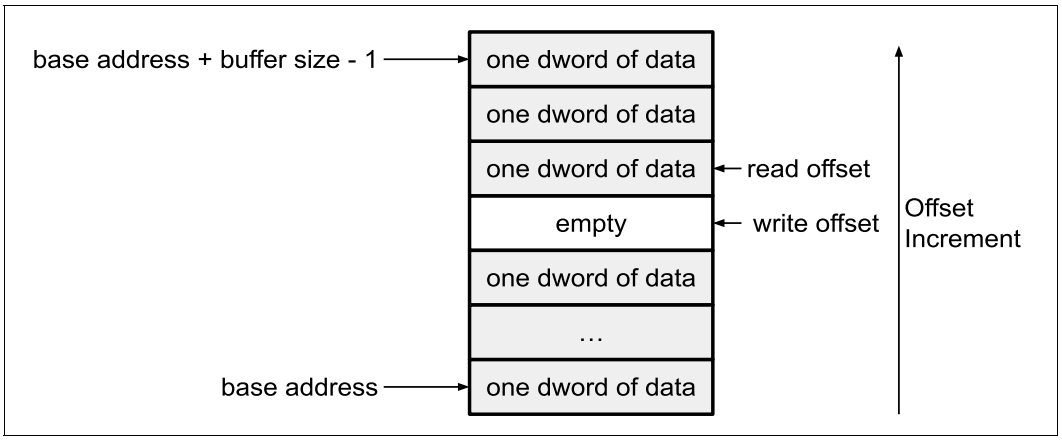
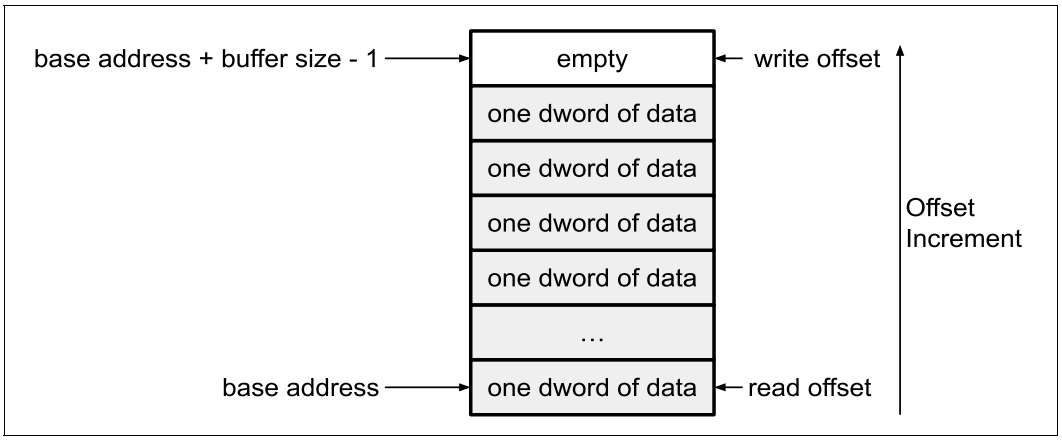


Figure 8-27. Full Circular Buffer with Write Offset above Read Offset



A Circular Buffer access has a maximum burst size which is the maximum value for data length of one UMAP packet. Maximum burst size is reported by the CB_MAX_BURST_SIZE field (see Table 8-28). If a UMAP packet that reads or writes from or to the Circular Buffer has a larger data length, a UMAP programming model violation error is reported (no error is reported in the Circular Buffer structure).

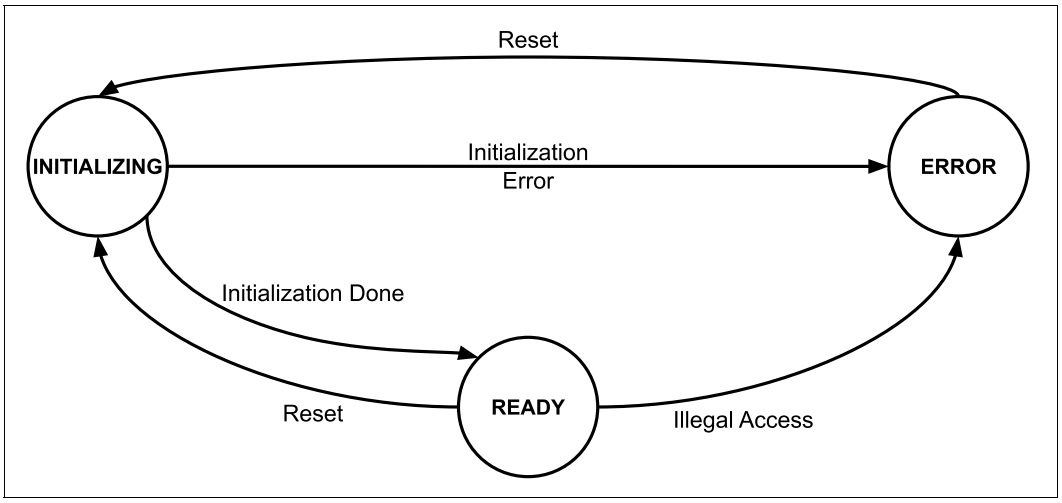
Note that the UCIe management fabric may impose further restrictions that impact the maximum burst size, such as the Maximum Packet Size (MPS) of each chiplet in the path. It is the responsibility of the requester to use a burst size that meets all the restrictions.

8.1.5.1.2 Circular Buffer State Machine

Figure 8-28 shows the possible states of a Circular Buffer and how it transitions from one state to another. A Circular Buffer must automatically transition from the INITIALIZING state to the READY state unless the management capability which defines the Circular Buffer also defines special actions that need to be taken to finalize the initialization of the Circular Buffer.

Note that the transition from INITIALIZING state to the READY state can take an implementation dependent amount of time. A Circular Buffer can transition from the INITIALIZING state to the ERROR state for implementation dependent reasons.

Figure 8-28. Circular Buffer State Machine



The following registers/fields can be accessed while the Circular Buffer is in any state:

- CB_VERSION
- CB_CAPABILITIES
- CB_STATE
- CB_ERROR_BITS
- CB_ERROR

If Reset Supported is 1 in CB_CAPABILITIES (see Table 8-22), then it is legal to write 1 to the Reset bit of CB_CONTROL while a Circular Buffer is in READY or ERROR states. Writing 1 to the Reset bit will reset the Circular Buffer, which shall transition to the INITIALIZING state. Additionally, the read offset and write offset will be reinitialized to an empty circular buffer (i.e., both values equal to each other). Note that read and write offset do not necessarily reset to 0.

It is illegal to write 1 to the Reset bit while the Circular Buffer is in the INITIALIZING state. The Circular Buffer shall ignore such reset requests and continue with its initialization.

It is also illegal to write 1 to the Reset bit if the reset operation is not supported (reported by CB_CAPABILITIES (see Table 8-22)). If 1 is written to the Reset bit when reset is not supported, then the Circular Buffer shall ignore such request.

When a Circular Buffer is in READY state, all Circular Buffer data structures can be accessed as long as it is allowed by access control and follows the write and read rules specified in Section 8.1.5.1.8 for a sink Circular Buffer or Section 8.1.5.1.9 for a source Circular Buffer.

In INITIALIZING or ERROR states, it is illegal for a requester to read (source Circular Buffer) or write (sink Circular Buffer) to a Circular Buffer with UMAP protocol.

Illegal accesses will be ignored and trigger a programming model violation (empty reply with programming model violation bit set). A Circular Buffer implementation may also report the error using Circular Buffer error reporting mechanisms (see Section 8.1.5.1.5).

8.1.5.1.3 Circular Buffer Initialization

Before using a Circular Buffer (sink or source) the UMAP requester must confirm it is initialized and ready. Circular Buffer initialization is defined by the management capability which exposes the Circular Buffer. Initialization, if necessary, should generally only need to happen once after management element initialization.

Circular Buffer initialization may also be needed to recover from some error conditions. See Table 8-23 for a list of errors which need Circular Buffer initialization for recovery.

When a UMAP read of CB_STATE reports READY state, initialization was successful. If an error occurred during initialization then CB_STATE will report ERROR state (see Section 8.1.5.1.5).

The Management Capability Structure may define how to handle multiple initialization attempt failures. For example, the Early Firmware Download capability (see Section 8.4.1 and more specifically Section 8.4.1.8) uses a sink Circular Buffer and defines that initialization should be retried at most one time if it fails after reset.

8.1.5.1.4 Circular Buffer Features

The Circular Buffer Capability register allows requesters to discover which optional features are supported by a particular Circular Buffer. See the capabilities register definition in Table 8-22 for the capabilities of the Circular Buffer that can be reported.

Table 8-22. CB_CAPABILITIES Circular Buffer Capabilities

Bit	Attribute	Description
0	RO	**Reset Supported** If 1, then reset is supported (i.e., initiator can reset the Circular Buffer using reset bit of CB_CONTROL). See Section 8.1.5.1.6 for details. If 0, then reset is not supported and if an error occurs, then the initiator must perform a full chiplet reset to recover from error. Note that this bit may change value and if it can change value then the UCIe management capability which uses Circular Buffer is responsible for defining when and why this bit can change.
1	RO	**Overflow/Underflow** Set for source Circular Buffers that detect and report overflow. Set for sink Circular Buffers that detect and report underflow.
31:2	RO	**Reserved**

8.1.5.1.5 Circular Buffer State and Errors

The CB_STATE field (see the field's definition in Table 8-28) can only take one of the values defined in Table 8-23. Each value corresponds to one state in the state machine of the Circular Buffer state machine described in Section 8.1.5.1.2.

Table 8-23. CB_STATE Field Value and Definitions

Field Values	Description
0	**READY** Circular Buffer ready either for write (sink Circular Buffer) or for read (source Circular Buffer). The requester can initiate write or read following the sequences described in Section 8.1.5.1.8 or Section 8.1.5.1.9.
1	**INITIALIZING** Circular Buffer is initializing. This is a transient state so it will not remain in this state indefinitely. When initialization successfully completes, CB_STATE changes to READY. The INITIALIZING state may timeout and proceed to ERROR state (see Section 8.1.5.1.3 for details).
2	**ERROR** See Table 8-24 and Table 8-25 for information on the types of errors that may occur.
Other Values	**Reserved**

When an error occurs, the error code reported in CB_ERROR field shall reflect the first error condition observed by the hardware. Every error that occurs accumulates in CB_ERROR_BITS where bits are set for every error type detected and remain set until a Circular Buffer or chiplet reset. See Table 8-24 for the error vector bit definitions. The CB_ERROR_BITS shall reset to 0 (no error) after a Circular Buffer reset. Note that if Circular Buffer reset is not supported (see Table 8-22), then only a full chiplet reset can clear CB_ERROR_BITS and CB_ERROR (see Section 8.1.5.1.2 for details).

Table 8-24. CB_ERROR_BITS — Error Vector

Bit	Attribute	Description
0	RO	**Internal** See CB_ERROR definition in Table 8-25 for details.
1	RO	**Initialization Failure** See CB_ERROR definition in Table 8-25 for details.
2	RO	**Access while Not Ready** See CB_ERROR definition in Table 8-25 for details.
3	RO	**Overflow** See CB_ERROR definition in Table 8-25 for details.
4	RO	**Underflow** See CB_ERROR definition in Table 8-25 for details.
31:5	RO	**Reserved**

The CB_ERROR field can only take one of the values defined in Table 8-25.

Table 8-25. CB_ERROR Values Definitions (Sheet 1 of 2)

Field Values	Description
0	**No Error**
1	**Internal** Internal error is for vendor defined errors.
2	**Initialization Failure** The Circular Buffer initialization failed. The capability structure which makes use of the Circular Buffer structure can further define what might be the cause of failures through error reporting mechanisms unique to each capability structure. If the capability structure does not further define the reasons then this should be considered as a vendor defined failure.

Table 8-25. CB_ERROR Values Definitions (Sheet 2 of 2)

Field Values	Description
3	**Access while Not Ready** Access to a register (other than the always legal registers) of the Circular Buffer structure or Circular Buffer when the state is not READY. Note that this error is not necessarily detected by a Circular Buffer and thus requesters should not depend on a Circular Buffer implementation to report such illegal access.
4	**Overflow** Circular Buffer overflow (i.e., either the requester wrote DWORDs between the read offset and the write offset, or the requester updated the write offset past the read offset). Value is 0 for source Circular Buffer. If the Circular Buffer implementation can detect overflow access, the implementation shall discard the overflowing write. Overflow detection is optional for a Circular Buffer.
5	**Underflow** Circular Buffer underflow (i.e., either the requester read DWORDs between the write offset and the read offset, or the requester updated the read offset past the write offset). Circular Buffer underflow detection is optional. This field is always 0 for a sink Circular Buffer.
Other values	**Reserved**

When the CB_ERROR field is set to a value other than “No Error”, the field will not be changed until the Circular Buffer or chiplet is reset.

The CB_VENDOR_DEFINED_ERROR_STATUS field defined in Table 8-26 is for vendor defined error status. It is for providing additional details related to the error reported in CB_ERROR (i.e., a user of the circular buffer does not need to use the CB_VENDOR_DEFINED_ERROR_STATUS when handling errors).

If there is no vendor defined status when the Circular Buffer is in ERROR state, then this field shall return zero (see Table 8-26).

Note that if CB_VENDOR_DEFINED_ERROR_STATUS returns 0, it does not mean that there are no errors. When an error occurs, CB_ERROR is set to a value other than No Error (see Table 8-25) and CB_STATE reports ERROR state.

Table 8-26. CB_VENDOR_DEFINED_ERROR_STATUS Value Definitions

Field Values	Description
0	No vendor defined error status. The reported error, if any, does not require any vendor defined error status.
Other values	Reserved for vendors. Vendors are free to define meaning for each of those values.

8.1.5.1.6 Control Register

Table 8-27 defines each bit of the Circular Buffer Control register (CB_CONTROL).

Table 8-27. CB_CONTROL — Control Register Bit Definition

Bit	Attribute	Description
0	RW	**Reset** Writing 1 to this bit will cause the Circular Buffer to become empty by setting the read and write offsets to the same value. The read and write offset values are not required to reset to 0. It is illegal to write 1 to the reset bit if CB_CAPABILITIES reports reset as not supported (see Section 8.1.5.1.2 and Section 8.1.5.1.4 for details). The Circular Buffer shall ignore such reset requests (i.e., when reset is not supported as reported in CB_CAPABILITIES). Setting this bit, when it is legal to do so, shall clear error conditions.
31:1	RW	**Reserved**

Read of the control register (CB_CONTROL) shall return 0 because this is a write only register.

8.1.5.1.7 Circular Buffer Common Fields

Table 8-28 defines registers and fields that are common between sink and source Circular Buffer.

Table 8-28. Sink and Source Circular Buffer Structure Fields (Sheet 1 of 2)

Register/Field Name	DWORD & Bit Location	Attribute	Description
CB_VERSION	0 [7:0]	RO	Version of the Circular Buffer control structure. Must be 0.
CB_CAPABILITIES	1 [31:0]	RO	Bit vector of supported capabilities (see Table 8-22 for details)
CB_MAX_BURST_SIZE	2 [7:0]	RO	Specifies the maximum data length for a UMAP operation to the Circular Buffer. The maximum number of DWORDs is equal to this field plus 1 (e.g., a value of 00h in this field indicates a data length of one DWORD, a value of 01h in this field indicates a data length of two DWORDs, and so on).
CB_BUFFER_SIZE	3 [31:0]	RO	The size of the Circular Buffer in DWORDs. A Circular Buffer must be at least one DWORD in size and the value 0 is reserved.
CB_ADDRESS_LOW	4 [31:0]	RO	Bits [31:0] of the 64-bit base address of the Circular Buffer. Because capability structures must be DWORD-aligned, bits [1:0] must be 00b.
CB_ADDRESS_HIGH	5 [31:0]	RO	Bits [63:32] of the 64-bit base address of the Circular Buffer.
CB_CONTROL	6 [31:0]	RW	Control vector (see Table 8-27 for details).
-	7 [31:0]	-	See Table 8-29 for a sink Circular Buffer. See Table 8-30 for a source Circular Buffer.
-	8 [31:0]	-	See Table 8-29 for a sink Circular Buffer. See Table 8-30 for a source Circular Buffer.
CB_STATE	9 [1:0]	RO	See Table 8-23 for details.
CB_ERROR_BITS	10 [31:0]	RO	See Table 8-24 for details.
CB_ERROR	11 [3:0]	RO	See Table 8-25 for details.
CB_VENDOR_DEFINED_ERROR_STATUS	11 [31:16]	RO	See Table 8-26 for details.
Reserved	12 [31:0]	RO	Reserved

Table 8-28. Sink and Source Circular Buffer Structure Fields (Sheet 2 of 2)

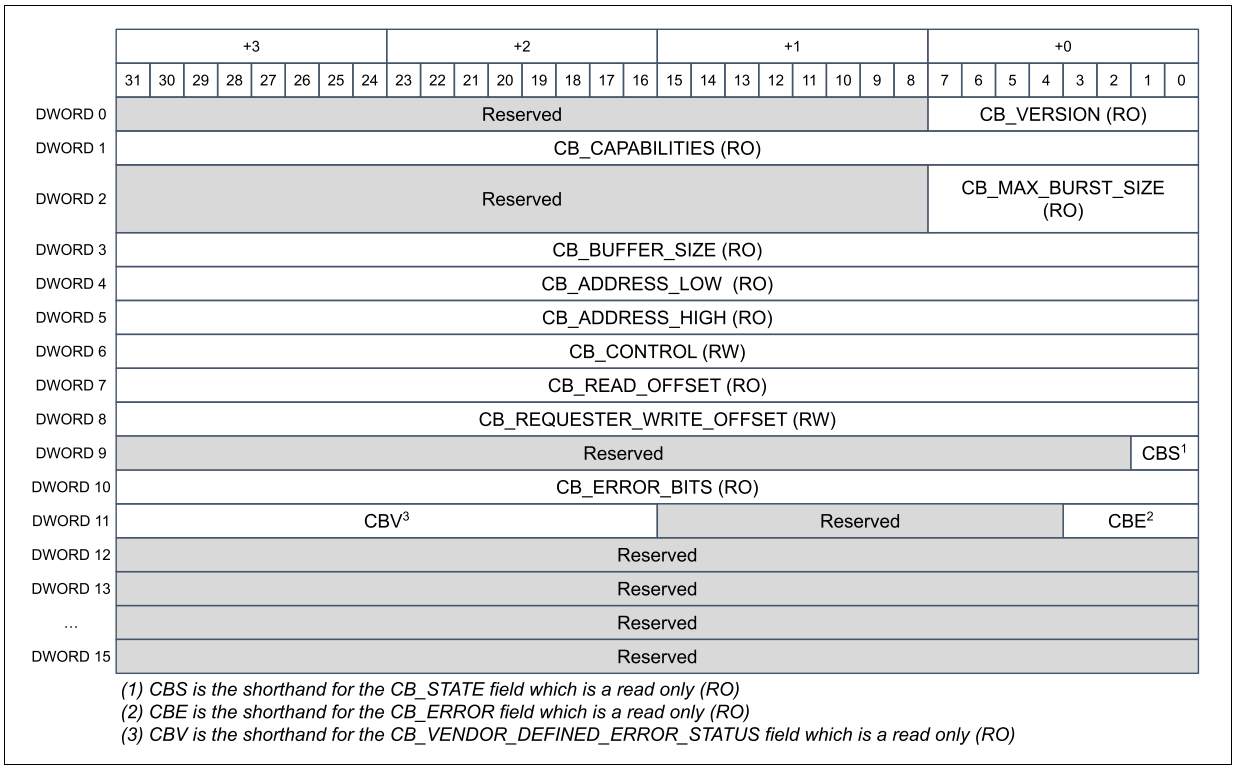
Register/Field Name	DWORD & Bit Location	Attribute	Description
Reserved	13 [31:0]	RO	Reserved
Reserved	14 [31:0]	RO	Reserved
Reserved	15 [31:0]	RO	Reserved

8.1.5.1.8 Sink Circular Buffer Structure

Figure 8-29 shows the sink Circular Buffer structure. A sink Circular Buffer is written to by a UMAP requester using the CB_REQUESTER_WRITE_OFFSET added to the 64-bit Circular Buffer base address for the start address of a UMAP write. The 64-bit circular base address is formed by concatenating CB_ADDRESS_HI with CB_ADDRESS_LOW.

The management entity that is reading from the Circular Buffer uses CB_READ_OFFSET added to the 64-bit Circular Buffer base address for the read's start address (see Figure 8-29 and Table 8-29).

Figure 8-29. Sink Circular Buffer, UMAP Requester Write to and Management Entity Read from



(1) CBS is the shorthand for the CB_STATE field which is a read only (RO)
(2) CBE is the shorthand for the CB_ERROR field which is a read only (RO)
(3) CBV is the shorthand for the CB_VENDOR_DEFINED_ERROR_STATUS field which is a read only (RO)

Table 8-29 defines the fields specific to a sink Circular Buffer. See Section 8.1.5.1.7 for the definition of the registers that are common between sink and source Circular Buffer.

Table 8-29. Sink Circular Buffer Structure Fields

Register/Field Name	DWORD & Bit Location	Attribute	Description
CB_READ_OFFSET	7 [31:0]	RO	The offset at which the management entity will read next from the Circular Buffer. Updated by the management entity.
CB_REQUESTER_WRITE_OFFSET	8 [31:0]	RW	The offset at which the UMAP requester shall write next to the Circular Buffer. Updated by the UMAP requester.

A Circular Buffer must be initialized and in READY state before a UMAP requester can write to it.

After the sink Circular Buffer is in READY state, the requester shall use the following steps to write to the Circular Buffer:

1. Read the CB_BUFFER_SIZE register *(cb size)*.
2. Read the CB_REQUESTER_WRITE_OFFSET register (*wr offset*).
3. Read the CB_READ_OFFSET register (*rd offset*).
4. Compute the maximum amount of DWORDs that the UMAP requester can write, referred to as (*max wr size*), using the following formula:

   (*max wr size*) = ((*cb size*) + (*rd offset*) – (*wr offset*) – 1) % (*cb size*)

   Note that the '%' symbol in the above formula represents the modulo operation.
5. The requester can write up to (*max wr size*) DWORDs, over multiple UMAP write packets, into the Circular Buffer before it needs to read CB_READ_OFFSET again. The requester must not write more than CB_MAX_BURST_SIZE DWORDs with each individual UMAP write packet and the end offset of the data written must not extend beyond the (*cb size*). Note that the UCIe management fabric might impose further restrictions on the maximum UMAP packet size (see the available DWORD sizes for the CMPS field in Table 8-11).

   The UMAP requester shall decide on a write size *(wr size)* for each UMAP write packet and the offset of the data written must not extend beyond the (*cb size*). After writing the last valid offset of the buffer, the next write offset *(next wr offset)* shall wrap around 0, computing *(next wr offset)* can be achieved using the following formula:

   (*next wr offset*) = ((*wr offset*) + *(wr size)*) % (*cb size*)

   Note that writing beyond the Circular Buffer size with one single UMAP write packet (where the write offset starts within the Circular Buffer area and the last write offset would be beyond the Circular Buffer size) will be reported as a programming violation error and nothing will be written to the Circular Buffer.
6. The UMAP requester may update the write offset to allocate any entries into the Circular Buffer once their write is acknowledged by the UCIe memory access protocol.

   The UMAP requester can update the write offset after each write is acknowledged, after all writes are acknowledged, or after a subset of all the writes are acknowledged.

The above procedure ensures that the UMAP requester never sets the write offset equal to the management entity read offset since that could cause the Circular Buffer to overflow.

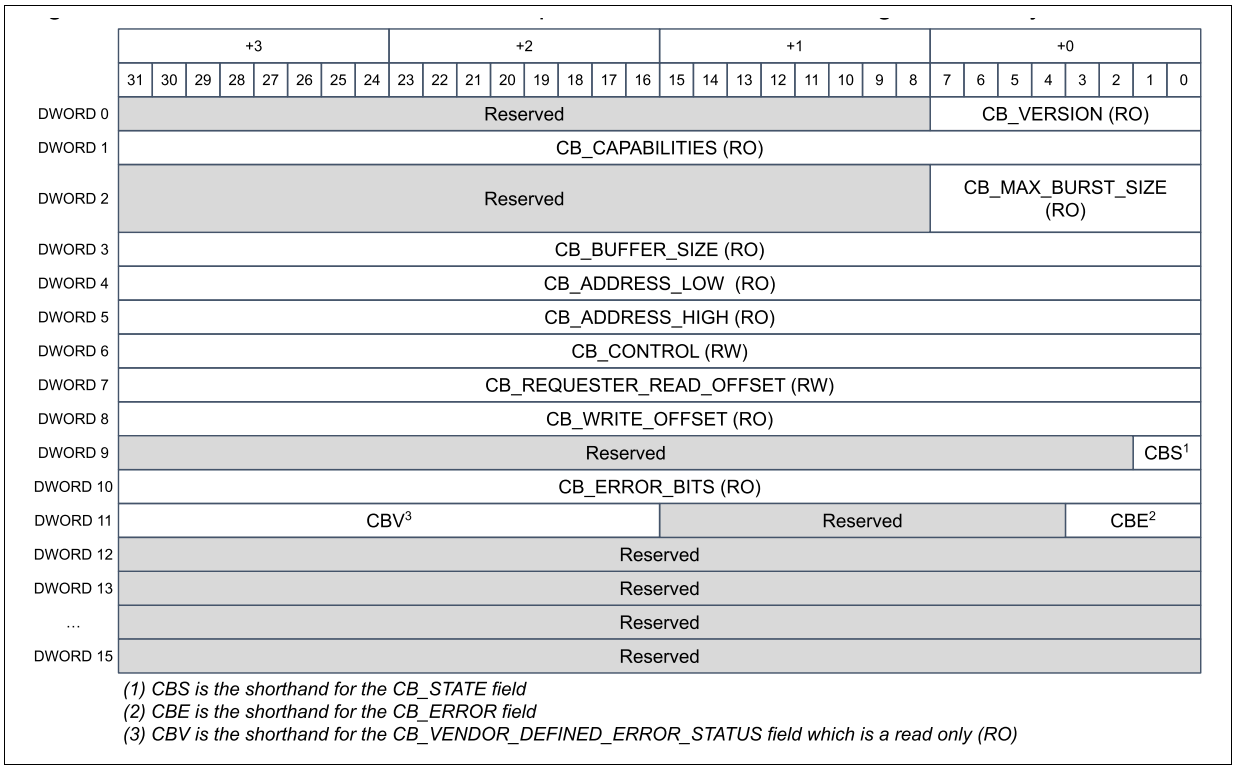
If the UMAP requester follows all the rules defined in this section, the UMAP requester does not need to check the error status between write transactions.

8.1.5.1.9 Source Circular Buffer Structure

Figure 8-30 shows the source Circular Buffer structure. A source Circular Buffer is read from by a UMAP requester using the CB_REQUESTER_READ_OFFSET added to the 64-bit Circular Buffer base address for the start address of a UMAP read.

The management entity that is writing to the Circular Buffer uses CB_WRITE_OFFSET added to the 64-bit Circular Buffer base address for the write's start address (see Figure 8-30 and Table 8-30). The flow to read from a source Circular Buffer is described below.

Figure 8-30. Source Circular Buffer, UMAP Requester Read from and Management Entity Write to



(1) CBS is the shorthand for the CB_STATE field
(2) CBE is the shorthand for the CB_ERROR field
(3) CBV is the shorthand for the CB_VENDOR_DEFINED_ERROR_STATUS field which is a read only (RO)

Table 8-30 defines the fields specific to a source Circular Buffer. See Section 8.1.5.1.7 for the definition of the registers that are common between sink and source Circular Buffers.

Table 8-30. Source Circular Buffer Structure Fields

Register/Field Name	DWORD & Bit Location	Attribute	Description
CB_REQUESTER_READ_OFFSET	7 [31:0]	RW	The offset at which the UMAP requester shall read next from the Circular Buffer.
CB_WRITE_OFFSET	8 [31:0]	RO	The offset at which the management entity writes next to the Circular Buffer.

A Circular Buffer must be initialized and in READY state before a UMAP requester can read from it.

When the source Circular Buffer is ready, the UMAP requester shall perform the following steps to read from the Circular Buffer:

1. Read the CB_BUFFER_SIZE register *(cb size)*.
2. Read the CB_REQUESTER_READ_OFFSET register (*rd offset*).

3. Read the CB_WRITE_OFFSET register (*wr offset*).
4. Compute the maximum amount of DWORDs that the UMAP requester can read (*max rd size*), using the following formula:

   (*max rd size*) = ((*cb size*) + (*rd offset*) – (*wr offset*)) % (*cb size*)

5. The requester can read up to (*max rd size*) DWORDs, over multiple UMAP read packets, from the Circular Buffer before it needs to read again CB_WRITE_OFFSET. The requester must not read more than CB_MAX_BURST_SIZE DWORDs with each individual UMAP read packet and the end offset of the data read must not extend beyond the (*cb size*). Note that the UCIe management fabric might impose further restrictions on the maximum UMAP packet size (see the available DWORD sizes for the CMPS field in Table 8-11).

   The UMAP requester shall decide on a read size *(rd size)* for each UMAP read packet and the end offset of the data read must not extend beyond the (*cb size*). After reading the last valid offset of the buffer, the next read offset *(next rd offset)* shall wrap around 0, computing (next rd offset) can be achieve using the following formula:

   (*next rd offset*) = ((*rd offset*) + *(rd size)*) % (*cb size*)

   Note that reading above the Circular Buffer size with one single UMAP read packet (where the read offset starts within the Circular Buffer area and the last read offset would be beyond the Circular Buffer size) will be reported as a programming violation error.

6. The UMAP requester may update CB_REQUESTER_READ_OFFSET to deallocate entries from the Circular Buffer once the UMAP packet reading the entry has completed.

   The UMAP requester can incrementally update CB_REQUESTER_READ_OFFSET as reads complete or can wait until all reads have completed before updating CB_REQUESTER_READ_OFFSET.

If the requester follows all the rules defined in this section, the UMAP requester does not need to check the error status between read transactions.

8.1.5.1.10 UMAP Requester Coordination

If there are multiple UMAP requesters for the same Circular Buffer, coordination of the requests to the Circular Buffer is the responsibility of the UMAP requesters and beyond the scope of this specification.

8.1.5.1.11 How to Use Circular Buffer

The capability structures which use the Circular Buffer must use the Circular Buffer control structure as shown in Figure 8-29 or Figure 8-30 and with all the fields described in Table 8-28 and either Table 8-29 or Table 8-30 depending if the Circular Buffer is a sink or a source respectively. The Circular Buffer control structure can appear at any DWORD-aligned UMAP address within the capability structure which exposes a Circular Buffer structure.

The capability structure shall define the following mandatory requirement:

- Security asset class for the Circular Buffer control structure.

The capability structure can define the following optional requirement:

- Initialization sequence if initialization is necessary.

8.1.6 Open Drain Configuration Structure

Open Drain signals can be used to notify other chiplets using a low-latency event. The Open Drain is a bidirectional event. The electrical specifications for Open Drain are described in Section 5.14.

Each Open Drain instance will have an associated capability structure. There is no limit on the number of Open Drain signals on a chiplet.

Figure 8-31. Open Drain Detection Capability Structure

	+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
DWORD 0	Rsvd								Management Capability ID								Reserved									Version							
DWORD 1	Sampling Rate Units								Reserved								Sampling Rate Value															
DWORD 2	Vendor-defined																Valid Sample Count															

Table 8-31. Open Drain Detection Capability Structure Fields

Field Name	DWORD & Bit Location	Standard Security Asset Class ID[a]	Attribute	Description
Version	0 [7:0]	17	RO	**Capability Structure Version** This field indicates the version of this capability structure. This field has a value of 00h in this specification.
Management Capability ID	0 [29:16]	17	RO	**Management Capability ID** This field specifies the Capability ID of this Management Capability structure. The Open Drain Configuration structure has a Management Capability ID of 005h.
Sampling Rate Value	1 [15:0]	17	RO	This field reports the chiplet's sampling rate of the input to the Open Drain signal. This value is combined with the Sampling Rate Units to determine the sampling rate. The Sampling Rate Value is equal to this field plus 1 (e.g., a value of 0000h in this field indicates a sample rate value equal to 0001h): Sampling rate (Hz) = 1 / (Sampling Rate Value * Sampling Rate Units) ***Note:*** Maximum Sampling Rate Value is 65,536. The sampling rate tolerance must be ±25% of the reported value.
Sampling Rate Units	1 [31:29]	17	RO	**Sampling Rate Value Units** 000b: ps 001b: ns 010b: us 011b: ms Others: Reserved
Valid Sample Count	2 [15:0]	16	RW	Number of consecutive matching samples of Open Drain input required before detecting the event. The Valid Sample Count is equal to the value of this field plus 1 (e.g., a value of 0000h in this field indicates a Valid Sample Count equal to 0001h). It is expected that this value is programmed once at initialization. This value must be <= *Maximum Sample Count*.

a. See Table 8-7 for a description of Standard Security Asset Class IDs.

8.1.6.1 Example Usage

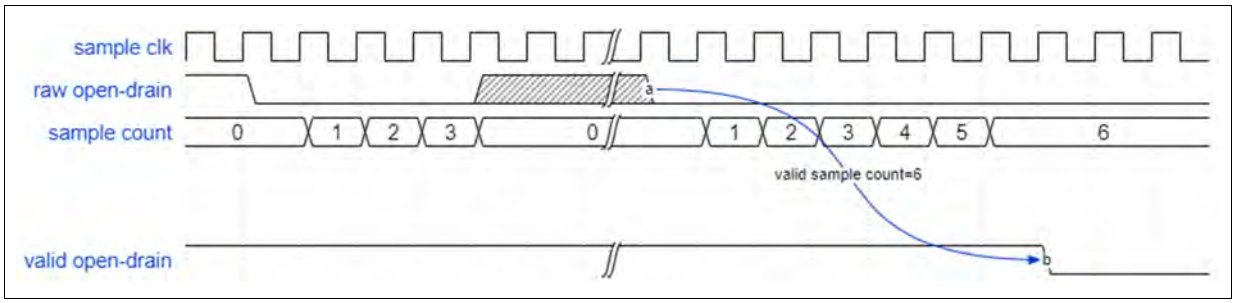
In the example shown in Figure 8-32:

- Raw Open-Drain: Value of shared signal as seen at the input of the chiplets Open Drain ball
- Valid Open-Drain: Value of the Open Drain ball after qualification by Valid Sample Count

An example of the *Valid Sample Count* usage is shown in Figure 8-32. In this example:

1. The *Valid Sample Count* is set to 6 (seven consecutive matching samples required).
2. *Raw Open-Drain* is detected as asserted for four clocks, then detected as de-asserted.
3. The *Valid Sample Count* threshold is not met; the *sampled Open Drain* is not asserted.
4. *Raw Open-Drain* is again detected as asserted for seven consecutive clocks.
5. *Valid Sample Count* threshold is satisfied and the *valid Open Drain* is asserted.

Figure 8-32. Valid Sample Count Example



IMPLEMENTATION NOTE

Re-arming of Open Drain

The assertion of an Open-Drain signal is governed by the event's Trigger Control fields, which specify a minimum assertion duration. When any chiplet de-asserts the signal, it will be perceived as "high" by other chiplets in less than 2 us. It is vendor-defined implementation to prevent false detection during this time.

8.2 Management Transport Packet (MTP) Encapsulation

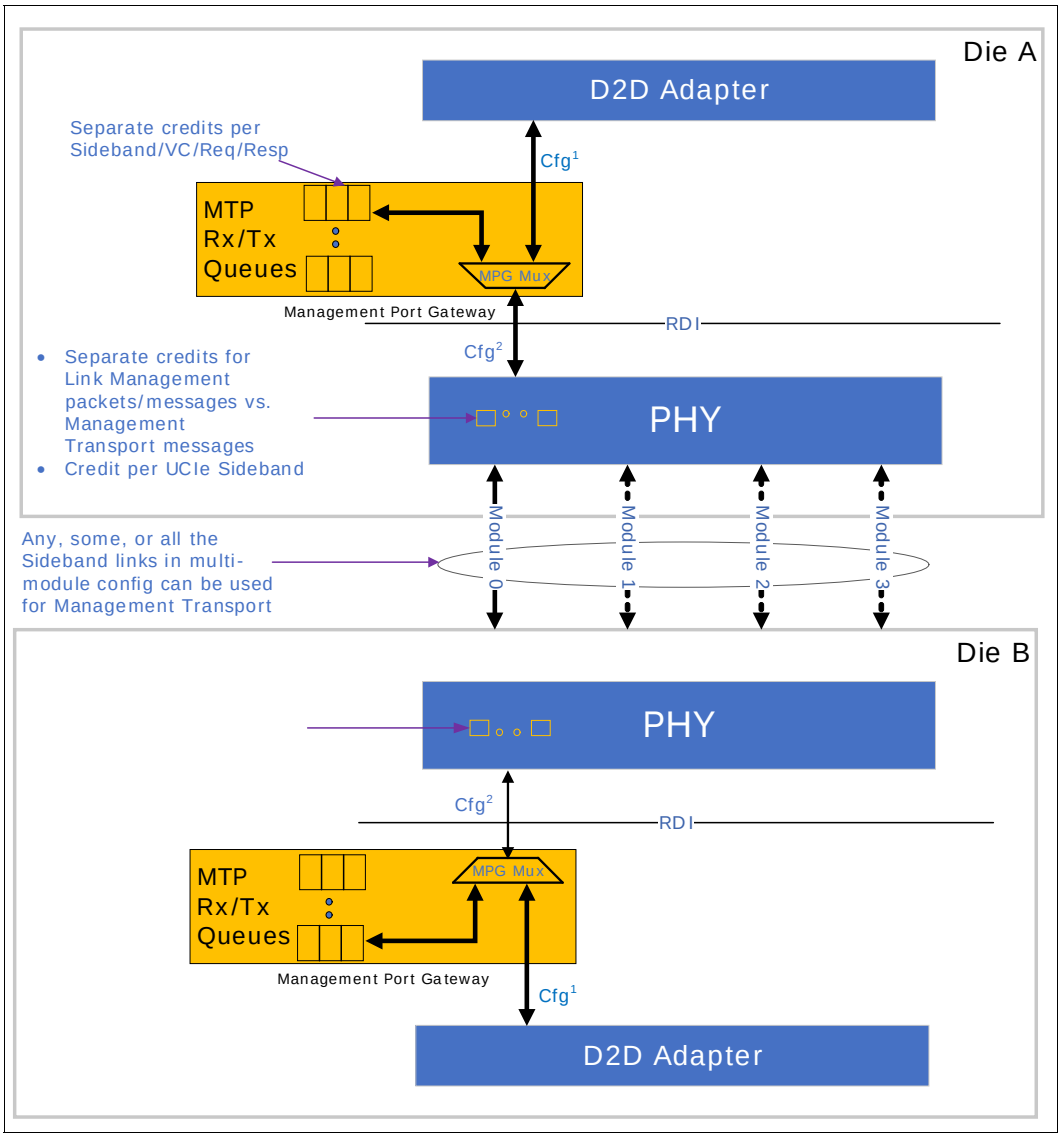
8.2.1 MTP Encapsulation Architecture Overview

Management Transport Packet (MTP) is the message used for management-related functionality on a management network in UCIe-based chiplets (see Section 8.1.1 for details). This section deals with how MTPs are sent/received over UCIe Sideband and Mainband links through the process of Encapsulation. All Management Ports (as defined in Section 8.1.2) in a chiplet must support Encapsulation. Chiplet support for Management Port on a UCIe sideband link or a UCIe mainband link are independently optional, subject to rules stated in Section 8.1.2. When Management Port is supported on a UCIe link (sideband or mainband) the management path on the link is setup as described in Section 8.2.3.1 for sideband and Section 8.2.3.2 for mainband. After the management path is set up on a link, the Encapsulated MTPs can be sent and received. Throughout this document the term Management Port Messages (MPM) is used to refer to all Sideband and Mainband messages that relate to Encapsulation. For the Sideband interface, these messages are defined in Table 7-1. For the mainband, these messages are defined in Table 8-32 and a "Management Flit" is defined (see Table 3-4 and Table 3-5) to carry these messages.

See Figure 8-33 and Figure 8-34 for a high-level view of the MTP transport path over UCIe sideband and mainband respectively. In this architecture, MTPs are transported between Management Port Gateways (MPGs) on each end of the UCIe link using either the sideband or the mainband path. In this context, an MPG is an entity that provides the bridging functionality when transporting an MTP from/to a local SoC management fabric (which is an SoC-specific implementation) to/from a UCIe

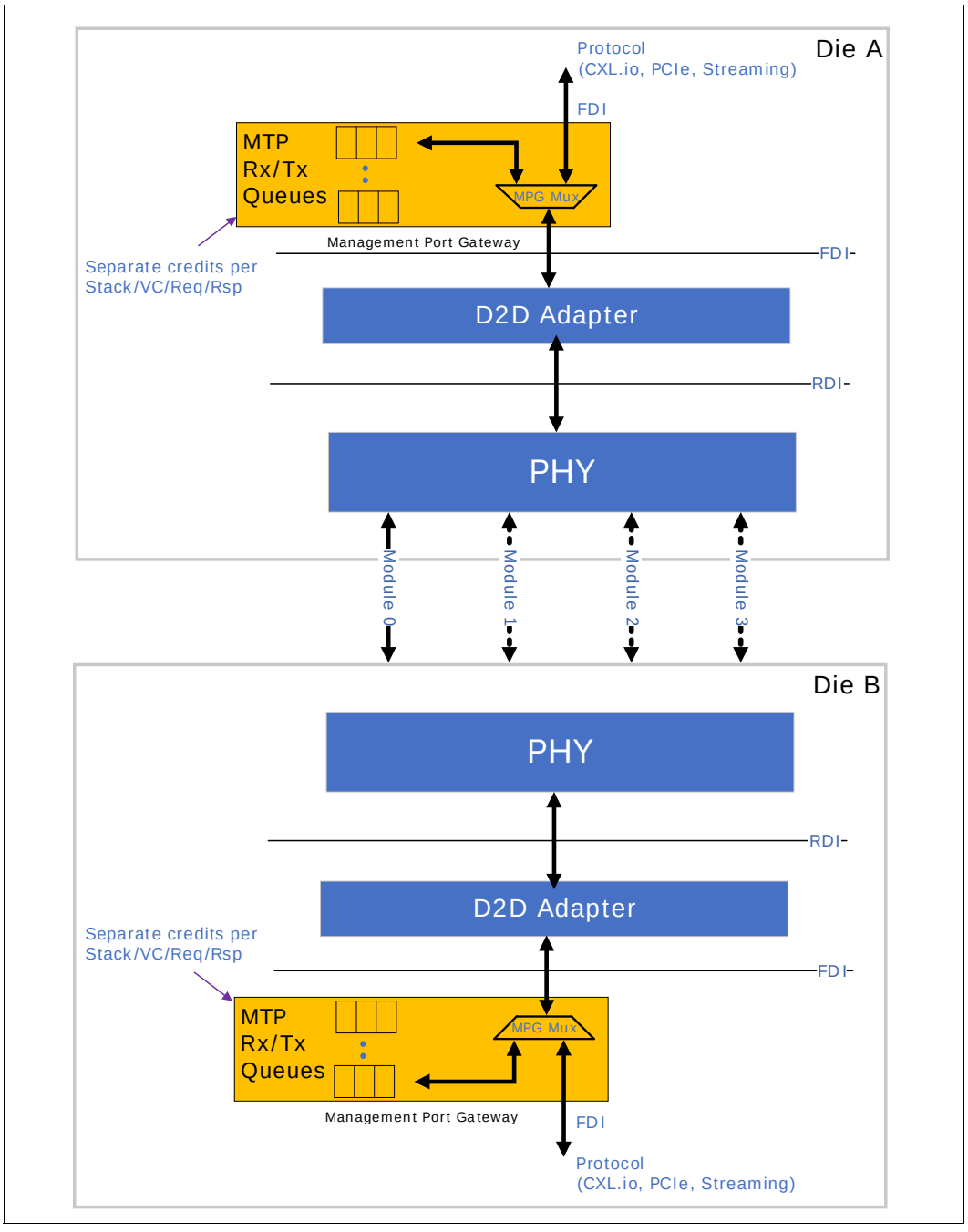
link. The MPG has credited buffers for receiving MTPs (called RxQ) from the link and their sizes are exchanged during initial link training. These credited buffers are separately maintained for Sideband and Mainband paths when management transport is supported on both. Up to eight VCs can be supported on a Management Port. Dedicated buffering is required for each VC negotiated. Support for VC0 is mandatory when management transport is negotiated, and all other VCs are optional.

Figure 8-33. UCIe Sideband Management Path Architecture[a] [b]



a. Configuration interface (i.e., **pl_cfg_*** and **lp_cfg_*** signals) described in Table 10-1.
b. Configuration interface (i.e., **pl_cfg_*** and **lp_cfg_*** signals) described in Table 10-1 plus extensions described in Table 10-3.

Figure 8-34. UCIe Mainband Management Path Architecture



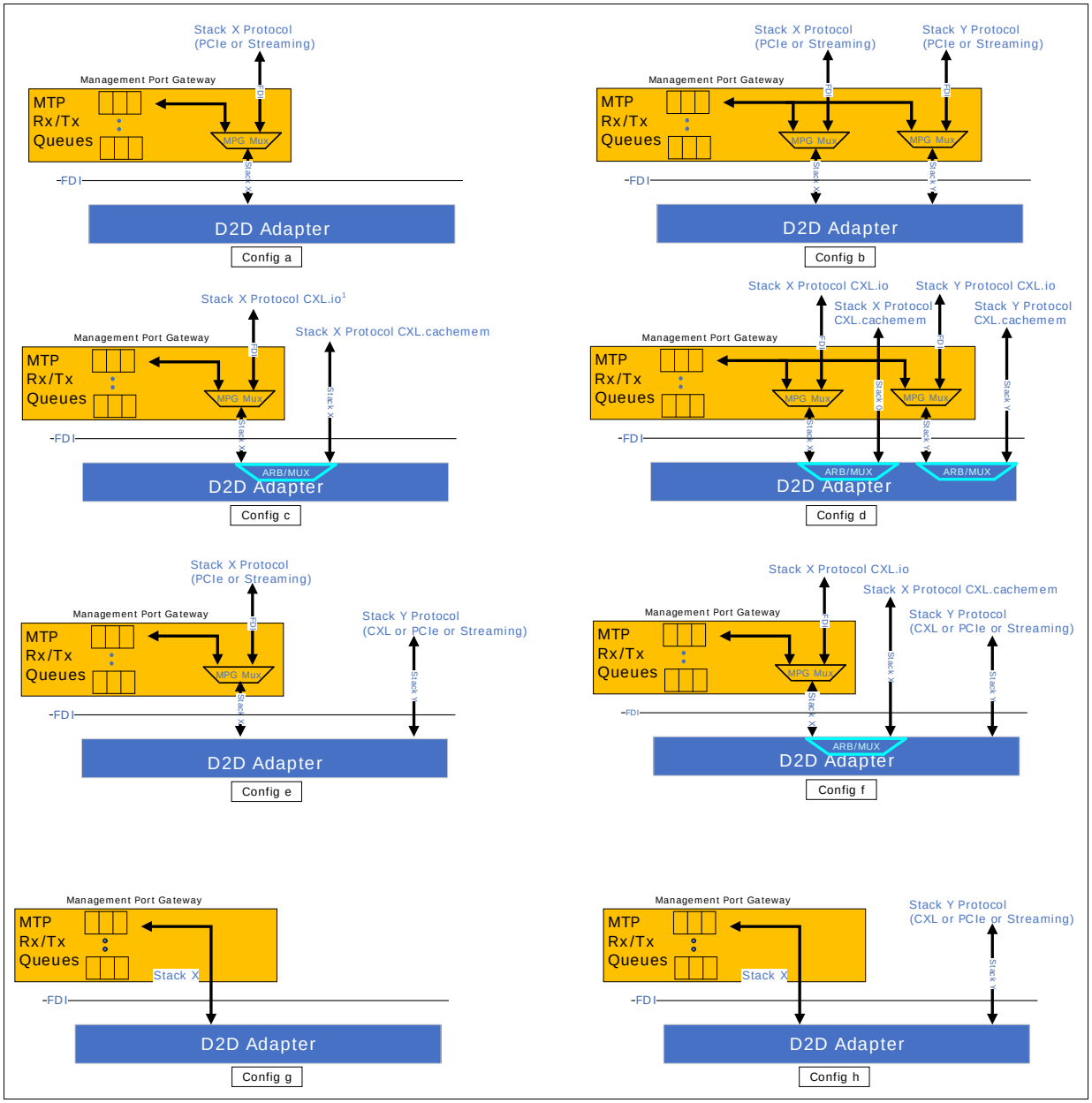
In multi-module or multi-sideband-only link configurations, any, some or all of up to four sideband links can be used for transporting MTPs. Unless stated otherwise, any references to sideband management port behavior/requirements/rules for a multi-module configuration also apply to a multi-sideband-only link configuration. In multi-stack mainband configurations, any or both stacks can be used for transporting MTPs. Ordering is still maintained when transporting packets on multiple sideband links/multiple stacks and this is described in Section 8.2.4.3. Because MTPs can be large (up to 2K payload) and can block the sideband link for regular link management traffic (as described in Table 7-1 except opcodes 10111b and 11000b), there is a mechanism provided to periodically arbitrate the link between link management packets/messages and MPMs over the sideband link. Additionally, to allow management path over sideband (when supported) to operate independent of mainband link status (which is required for certain management use cases such as FW download), UCIe link state machine supports sending/receiving management packets over sideband in all link states including RESET (see Chapter 4.0 for details).

In Figure 8-33 and Figure 8-34, the location of the Management Port Gateway mux is shown for reference purposes only. Implementations can choose to locate the mux elsewhere (e.g., above FDI for sideband path) and details of such implementations are beyond the scope of this specification. Interface definitions for this architecture, seen in Chapter 10.0, and other details discussed around Management Port Gateway integration are with respect to this reference Management Port Gateway mux placement.

The Management Port Gateway interfaces to the D2D Adapter by way of the FDI for mainband transport as shown in Figure 8-34, and FDI is described in Chapter 10.0. The Management Port Gateway can also connect directly to D2D by way of the FDI. Supported configurations of Management Port Gateway connectivity to D2D are shown in Figure 8-35.

The terminology used throughout this chapter will be in reference to the concepts shown in Figure 8-33 and Figure 8-34. In case of CXL protocol, the Management Port Gateway mux is on the CXL.io FDI.

Figure 8-35. Supported Configurations for Management Port Gateway Connectivity to D2D Adapter on Mainband



8.2.2 Management Port Messages

8.2.2.1 Sideband

There are currently two MPM opcodes defined as shown in Table 7-1, “Sideband Packet Opcode Encodings Mapped to Sideband Packet Types”. See Section 7.1.2.4 for more information.

8.2.2.2 Mainband

All MPMs on mainband carry a 2-DWORD header referred to as “MPM Header” (see Figure 8-36 and Figure 8-39). In that Header, bits [4:0] in the first DWORD carry the MPM opcode and are defined in Table 8-32. The remainder of this section discusses the format of these opcode messages. See Section 8.2.5.2.3 for details of how these messages are packed in the Management Flit when transmitting over the mainband.

Table 8-32. MPM Opcodes on Mainband

Opcode	Message
10111b	MPM without Data
11000b	MPM with Data
Others	Reserved

8.2.2.2.1 MPMs with Data

Bits [21:14] in the first DWORD of the MPM header (see Figure 8-36) of an MPM with Data message form an 8b msgcode that denotes a specific MPM with Data message. Supported MPM with data messages on the mainband are shown in Table 8-33.

Table 8-33. Supported MPM with Data Messages on Mainband

msgcode	Message
01h	Encapsulated MTP Message
FFh	Vendor-defined Management Port Gateway Message
Others	Reserved

The term “MPM payload” is used in the remainder of this section to refer to the payload in the MPM with Data messages.

8.2.2.2.1.1 Common Fields in MPM Header of MPM with Data Messages on Mainband

Figure 8-36 shows and Table 8-34 describes the common fields in the MPM header of MPM with data messages on the mainband.

Figure 8-36. Common Fields in MPM Header of all MPM with Data Messages on Mainband

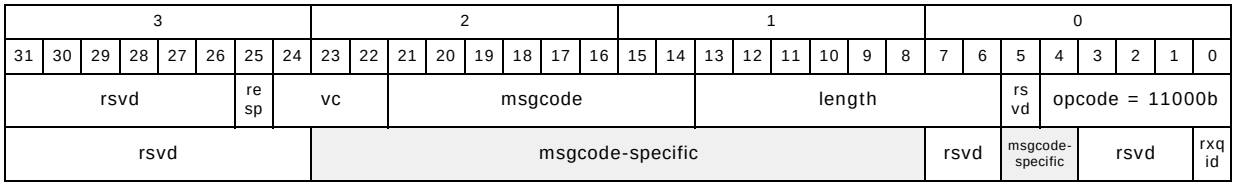


Table 8-34. Common Fields in MPM Header of all MPM with Data Messages on Mainband (Sheet 1 of 2)

Field	Description
opcode	11000b: MPM with Data.
length	MPM Payload length (i.e., 0h for 1 QWORD, 1h for 2 QWORDs, 2h for 3 QWORDs, etc.).
msgcode	Message code as defined in Table 8-33.

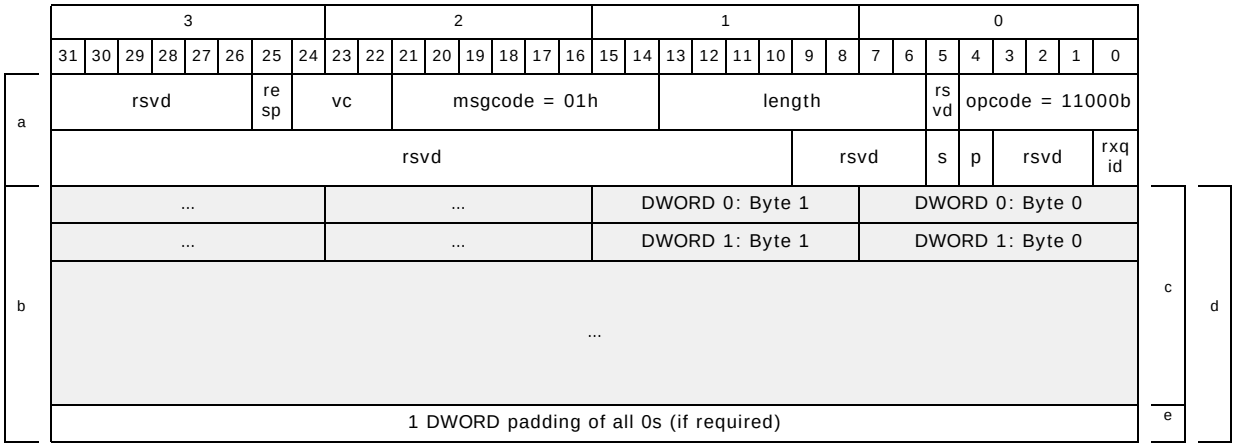
Table 8-34. Common Fields in MPM Header of all MPM with Data Messages on Mainband (Sheet 2 of 2)

Field	Description
vc	Virtual Channel ID.
resp	0: Request MPM. 1: Response MPM. For a Vendor-defined Management Port Gateway Message with Data, this bit is always 0 (see Section 8.2.2.2.1.3).
rxqid	RxQ-ID to which this packet is destined. See Section 8.2.3.2.2 for RxQ details. rxqid=0 corresponds to Stack 0. rxqid=1 corresponds to Stack 1.

8.2.2.2.1.2 Encapsulated MTP Message

Encapsulated MTP on the mainband is an MPM with Data with a msgcode of 01h.

Figure 8-37. Encapsulated MTP on Mainband



a. MPM Header.
b. MPM Payload.
c. Management Transport Packet (MTP).
d. Length in MPM Header.
e. DWORD padding.

Table 8-35. Encapsulated MTP on Mainband Fields

Location	Bit	Description
MPM Header[a]	s	Segmented MTP (see Section 8.2.4.2). The first and middle segments in a segmented MTP have this bit set to 1. The last segment in a segmented MTP will have this bit cleared to 0. An unsegmented MTP also has this bit cleared to 0.
	p	1-DWORD padding of all 0s added at the end of the packet, if required to align to a QWORD boundary.
MPM Payload	—	See Section 8.1.3.1 for details. Note that DWORDx:Bytey in Figure 8-37 refers to the corresponding DWORD, Byte defined in the Management Transport Packet in Figure 8-5.

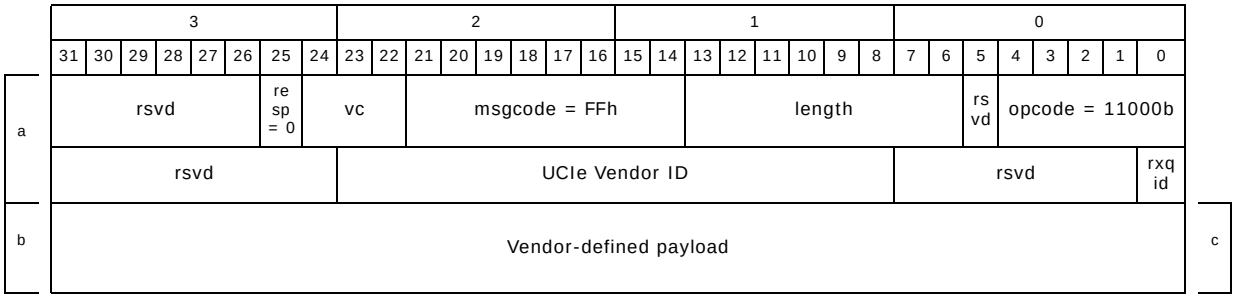
a. See Section 8.2.2.2.1.1 for details of header fields common to all MPMs with data on the mainband.

8.2.2.2.1.3 Vendor-defined Management Port Gateway Message

The Vendor-defined Management Port Gateway message with data is defined for custom communication between MPGs on the two ends of a UCIe mainband link. These messages are not part

of the Management transport protocol, and these messages start at an MPG and terminate at the MPG on the other end of the UCIe mainband link. These messages share the same rxqid buffers as encapsulated MTP messages. If an MPG does not support these messages or does not support vendor-defined messages from a given vendor (identified by the UCIe Vendor ID in the header), the MPG silently drops those messages. Ordering of these messages sent over multiple mainband stacks is subject to the same rules presented in Section 8.2.4.3 for encapsulated MTPs.

Figure 8-38. Vendor-defined Management Port Gateway Message with Data on Mainband



a. MPM Header.
b. MPM Payload.
c. Length in MPM Header.

Table 8-36. Vendor-defined Management Port Gateway Message with Data on Mainband Fields

Location	Field	Description
MPM Header[a]	UCIe Vendor ID	UCIe Consortium-assigned unique ID for each vendor.
MPM Payload	—	Vendor-defined.

a. See Section 8.2.2.2.1.1 for details of header fields common to all MPMs with data on the mainband.

8.2.2.2.2 MPMs without Data

Bits [21:14] in the first DWORD of the MPM header of an MPM without Data message form an 8b msgcode that denotes a specific MPM without Data message. Table 8-37 lists the supported msgcodes.

Table 8-37. Supported MPM without Data Messages on Mainband

msgcode	Message
01h	Management Port Gateway Capabilities Message
03h	Init Done Message
FFh	Vendor-defined Management Port Gateway Message
Others	Reserved

8.2.2.2.2.1 Common Header Fields of MPM without Data Messages on Mainband

Figure 8-39 shows and Table 8-38 describes the common fields in the MPM header of MPM without data messages on the mainband.

Figure 8-39. Common Fields in MPM Header of all MPM without Data Messages on Mainband

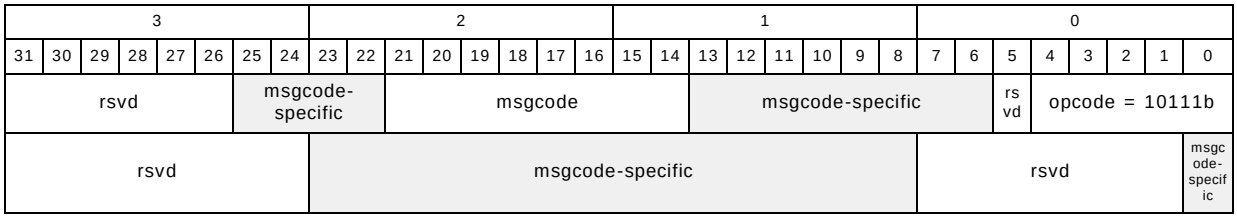


Table 8-38. Common Fields in MPM Header of all MPM without Data Messages on Mainband

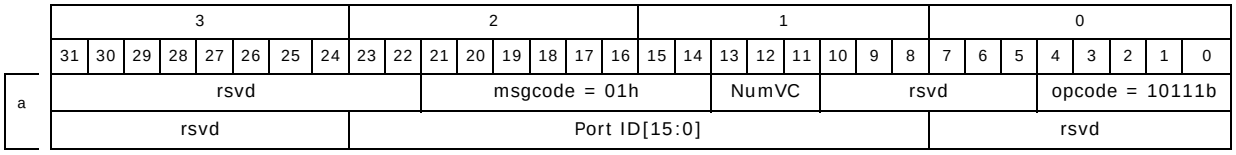
Field	Description
opcode	10111b: MPM without Data.
msgcode	Message code as defined in Table 8-37.

8.2.2.2.2.2 Management Port Gateway Capabilities Message

See Section 8.2.3.2.2 for details of how this message is used during mainband management path initialization.

Figure 8-40 shows and Table 8-39 describes the Management Port Gateway Capabilities message format on the mainband.

Figure 8-40. Management Port Gateway Capabilities MPM on Mainband



a. MPM Header.

Table 8-39. Management Port Gateway Capabilities MPM Header Fields on Mainband[a]

Field	Description
NumVC	Number of VCs supported by the Management Port Gateway that is transmitting the message.
Port ID	Port ID number value of the Management port associated with the Management Port Gateway that is issuing the message (see Section 8.1.3.6.2.1).

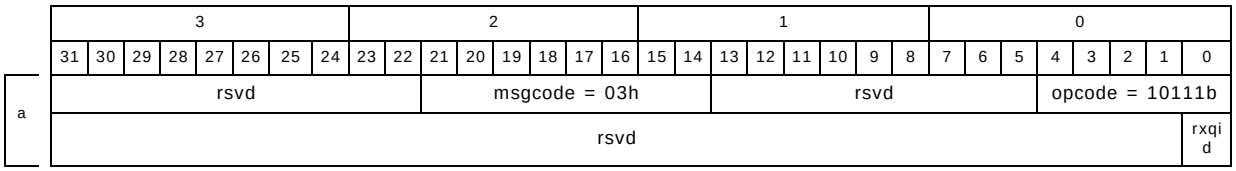
a. See Section 8.2.2.2.2.1 for details of header fields common to all MPMs without data on the mainband.

8.2.2.2.2.3 Init Done Message

See Section 8.2.3.2.2 for details of how this message is used during mainband management path initialization.

Figure 8-41 shows and Table 8-40 describes the Init Done message format on the mainband.

Figure 8-41. Init Done MPM on Mainband



a. MPM Header.

Table 8-40. Init Done MPM Header Fields on Mainband[a]

Field	Description
rxqid	RxQ-ID associated with the message. See Section 8.2.3.2.2 for RxQ details.

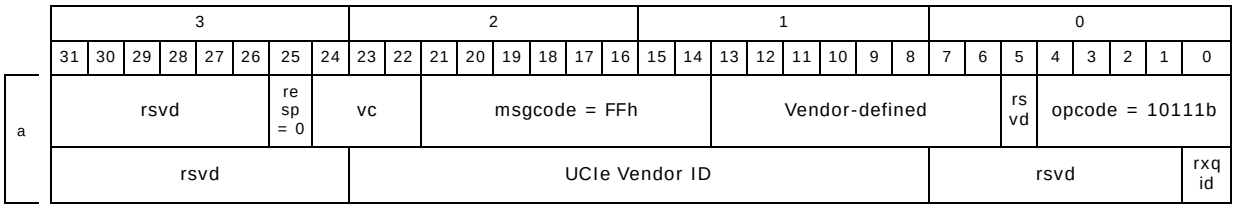
a. See Section 8.2.2.2.2.1 for details of header fields common to all MPMs without data on the mainband.

8.2.2.2.2.4 Vendor-defined Management Port Gateway Message

The Vendor-defined Management Port Gateway message without data is defined for custom communication between the MPGs on both ends of a UCIe mainband link. These messages are not part of the management transport protocol, and these messages start at an MPG and terminate at the MPG on the other end of the UCIe mainband link. These messages share the same rxqid buffers as encapsulated MTP messages. If an MPG does not support these messages or does not support 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 mainband has the format shown in Figure 8-41.

Figure 8-42. Vendor-defined Management Port Gateway Message without Data on Mainband



a. MPM Header.

Table 8-41. MPM Header Vendor-defined Management Port Gateway Message without Data on Mainband[a]

Field	Description
vc	Virtual Channel ID.
resp	Vendor-defined Management Port Gateway message without data always uses the Request channel.
UCIe Vendor ID	UCIe Consortium-assigned unique ID for each vendor.
rxqid	RxQ-ID to which this packet is destined. See Section 8.2.3.2.2 for RxQ details.

a. See Section 8.2.2.2.2.1 for details of header fields common to all MPMs without data on the mainband.

8.2.3 Management Transport Path Setup

Management transport path setup occurs in two distinct phases:

- **Negotiation phase** — In this phase, support for management transport, and when supported, the number of RxQs present in the partner chiplet, are negotiated. This is required for backward compatibility. This negotiation is done separately for sideband and mainband.
- **Initialization phase** — In this phase, the number of VCs supported is negotiated, and the RxQs in the Management Port Gateways on both ends of the link are initialized through credit exchanges for each supported VC. Port IDs are also exchanged.

Section 8.2.3.1 describes the setup process for the sideband. Section 8.2.3.2 describes the setup process for the mainband.

8.2.3.1 Sideband

Sideband Management Transport path setup occurs after a Management Reset or when Software writes 1 to the 'Retrain Link' bit in the Sideband Management Port Structure register (see Section 8.1.3.6.2.1). After setup is complete, management transport path over sideband remains active until the next Management Reset or until a 'Heartbeat timeout' is detected (as described in Section 8.2.5.1.3).

8.2.3.1.1 Negotiation Phase Steps

Negotiation occurs in the MBINIT.PARAM state. See Section 4.5.3.3.1.1 for details.

8.2.3.1.2 Initialization Phase Steps

If the Negotiation phase indicates support for Management transport and the SB_MGMT_UP flag (see Section 4.5) is cleared, Initialization phase steps are performed as indicated in this section.

A few general rules for RxQs that are initialized in this phase:

- Management Port Gateway maintains separate Rx queues for each sideband link over which it can receive MPMs. The Management Port Gateway can limit the number of Rx queues to be the same or smaller than the number of modules in the design. For example, in a design with four modules, a Management Port Gateway can choose to limit Rx queues to three or two or one.
- Each Rx queue in the Management Port Gateway is assigned a separate RxQ-ID and it is relevant for maintaining ordering when interleaving MTPs across multiple sideband links. See Section 8.2.4.3.
- See Section 8.2.4.1 for details of credit buffers that are required in each Rx queue.

- Number of RxQs finalized for transmitting and receiving MPMs is 0 to MIN{RxQ-Local, RxQ-Remote}, where RxQ-Local and RxQ-Remote are defined in Section 4.5.3.3.1.1.
- Transmission of MPMs with a given RxQ-ID is always associated with a specific local module that is design-specific. For example, an MPM with an RxQ-ID of 0 can be sent on any Module's sideband and that choice is design-specific. However, the choice is static and cannot change after the first MPM with that RxQ-ID is sent.
- Credits associated with each RxQ-ID are exchanged with a remote link partner by way of Credit Return messages as discussed below.

Initialization phase steps:

After **pm_param_done** is asserted and there is >0 module count negotiated for management transport for the local and remote sides, the Management Port Gateway begins the initialization process to the remote MPG for each RxQ-ID that the MPG needs to enable.

1. The initialization phase starts (shown in Figure 8-43, Figure 8-44, and Figure 8-45) with each Management Port Gateway sending the Management Port Gateway Capabilities message (see Figure 7-9 for message format).
   - Message can be sent on any RxQ-ID path, but sent only once per initialization phase from a chiplet to the partner chiplet.
   - Port ID value in the transmitted message is the value in Port ID field (see Table 8-12).
   - Port ID value in the received message is recorded in the "Remote Port ID" field (see Table 8-12).
   - NumVC field is the number of VCs supported by the transmitting Management Port Gateway. The number of VCs supported is the value in the NumVC field + 1. For example, if only one VC (VC0) is supported, NumVC is 0h. If two VCs are supported (VC0, VC1), then NumVC is 1h, etc.
   - MIN{Transmitted NumVC, Received NumVC}+1 number of VCs is enabled by each Management Port Gateway in the subsequent steps. The value of the enabled VCs starts from 0 and increments by 1 for each enabled VC up to MIN{Transmitted NumVC, Received NumVC}.
2. Management Port Gateway then sends credit Return messages for each enabled VC for each type (requests and responses), across all enabled RxQ-IDs. The Management Port Gateway is permitted to send this message. Figure 8-43 shows an example flow for the case of only a single RxQ (RxQ-ID=0) and single VC (VC0) negotiated during the negotiation phase. Figure 8-44 shows an example flow for the case of two RxQs (RxQ-ID=0, 1) and single VC (VC0) negotiated during the negotiation phase. Figure 8-45 shows an example flow for the case of only a single RxQ (RxQ-ID=0) and two VCs (VC0, VC1) negotiated during the negotiation phase.
   - Credit Return message (see Figure 7-10) contains an "RxQ-ID" field. The field must be assigned starting from 0 to MIN{RxQ-Local, RxQ-Remote}-1.
   - Infinite credits are permitted to be advertised. This is performed by sending a value of 3FFh in the "Rx Credit return in QWORDs" field for that VC and Type before the "Init Done".
   - There is no ordering requirement for credit return messages across VCs, types, nor RxQ-IDs. Multiple credit return messages are permitted for a given VC, type, and RxQ-ID.

Figure 8-43. Sideband Management Transport Initialization Phase Example with RxQ-ID=0 and One VC (VC0)

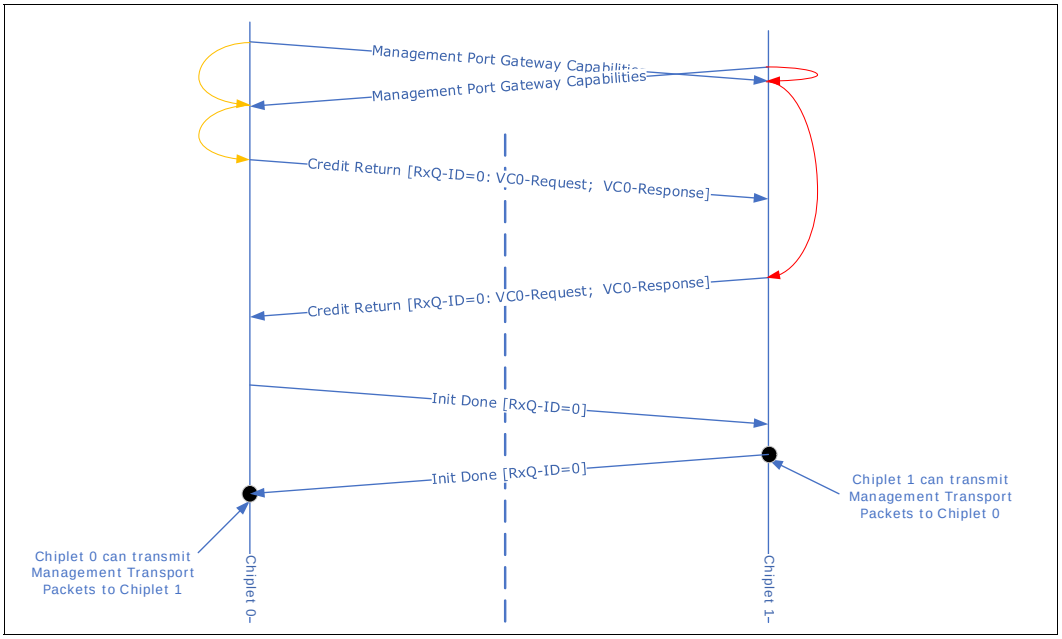


Figure 8-44. Sideband Management Transport Initialization Phase Example with RxQ-ID=0, 1 and One VC (VC0)

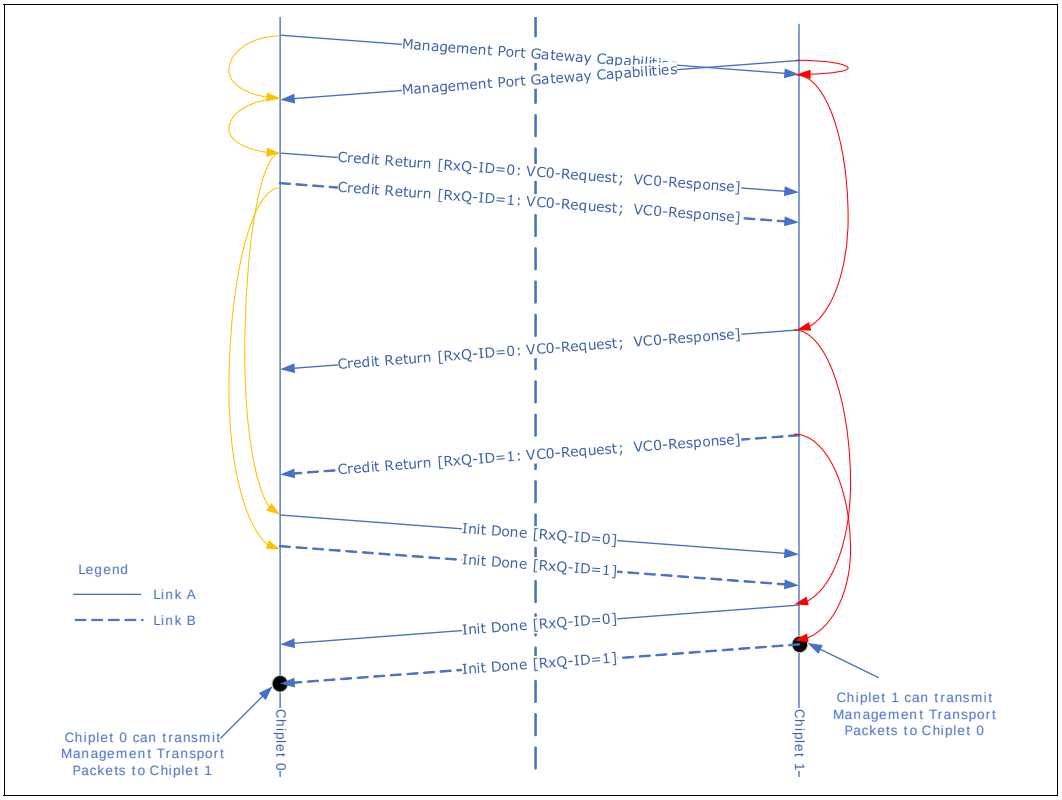
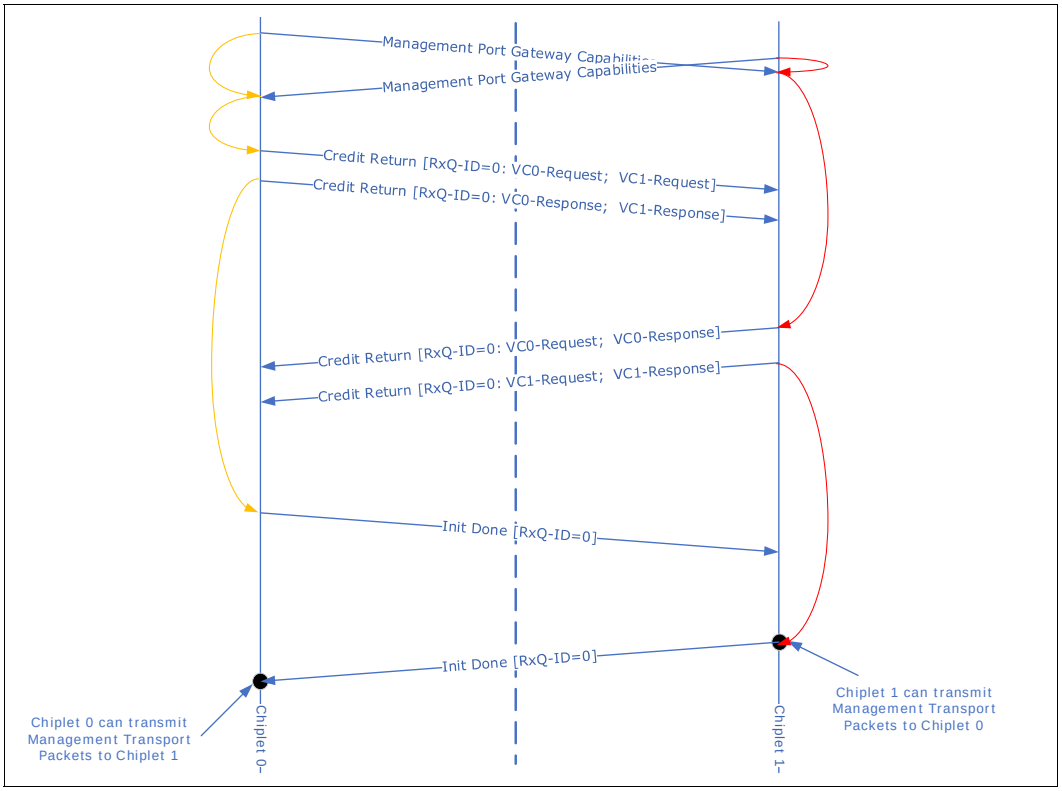


Figure 8-45. Sideband Management Transport Initialization Phase Example with RxQ-ID=0 and Two VCs (VC0, VC1)



3. After the last Credit Return message for a given RxQ-ID, the Management Port Gateway must send an "Init Done" message (see Figure 7-11) for the corresponding RxQ-ID. This informs the remote Link partner that a receiver has finished advertising credits for enabled VCs for the given RxQ-ID.
   - After "Init Done" has been transmitted and received by a Management Port Gateway for all available RxQ-ID paths, the MPG is ready for sending Management Transport packets.
     - Sideband should be able to send/receive management transport packets at this point without any dependency on the mainband link status.
     - Management Port Gateway asserts the **mp_mgmt_up** and **mp_mgmt_init_done** signals to PHY to indicate that the Management Transport Path was successfully initialized. PHY sets the SB_MGMT_UP flag when both **mp_mgmt_up** and **mp_mgmt_init_done** are asserted. The flag remains set until the management path goes down. In case of any fatal error (e.g., credit return messages were received for an RxQ-ID that is not expected, a timeout occurred while waiting for the Init Done message, etc.) during RxQ credit exchange, the **mp_mgmt_up** signal will remain de-asserted with the **mp_mgmt_init_done** signal asserted.
     - Note that the Management Port Gateway is unaware of PHY states and thus, after the **mp_mgmt_up** signal is asserted, the Management Port Gateway assumes that the management path through the sideband is available unless there is a Management Reset or the MPG detects an error through the mechanism described in Section 8.2.5.1.3.

    - After the SB_MGMT_UP flag is set, sideband link is available for sideband packet (MPMs or any other sideband packets) transmission/reception in all state machine states including RESET/SBINIT.
  - After the Management Port Gateway receives the “Init Done” message for a given RxQ-ID, the MPG must be ready to receive MTPs with that RxQ-ID.

The PHY Layer routes a message with a given RxQ-ID (specified by the **mp_rxqid** signal) to a specific module’s sideband link and that association is design-specific. Note that because RxQ-ID association to a module sideband is design-specific, on the same sideband link, messages with different RxQ-IDs in each direction are possible.

8.2.3.1.3 Other Sideband Management Transport Path Rules

- Sideband interfaces successfully initialized for management transport are available for management transport regardless of the associated mainband module’s state.
  - Note that when management transport is NOT supported and Module 0’s mainband is disabled at runtime, the sideband interface on that module is also disabled and D2D messages are routed to the sideband interface of the next-available lowest-numbered module that is enabled. When management transport is supported and enabled on the sideband link, the sideband link remains active for both management and non-management packets even if the corresponding mainband module is disabled.
- If SW writes 1 to the ‘Retrain Link’ bit in the Management Port Structure register associated with a sideband link when the Management Path is already up on that port, the Management Port Gateway must follow the ‘Heartbeat timeout’ flow (see Section 8.2.5.1.3) to bring the management path down before instructing the PHY to restart link negotiation (by the **sb_mgmt_init_start** signal).

8.2.3.2 Mainband

Mainband Management Transport path setup occurs when a link trains up. After the setup is complete, the management transport path remains active until a Domain Reset or until the link or the associated stack(s) goes down.

8.2.3.2.1 Negotiation Phase Steps

Mainband Management Transport path negotiation occurs on every mainband link training, thereby leveraging the existing D2D adapter protocol negotiation messages/flows. Support for Management Transport protocol within a stack is explicitly indicated with a new bit in the negotiation flow (see Table 3-1).

Section 3.1 and Section 3.2 provide Management Transport protocol negotiation details. At the end of protocol negotiation, the D2D adapter indicates the number of D2D stacks that negotiated Management Transport protocol by signals discussed in Table 10-3.

8.2.3.2.2 Initialization Phase Steps

A few general rules for the RxQs that are initialized in this phase:

- Management Port Gateway maintains separate Rx queues for receiving MTPs over each negotiated stack.
- Each Rx queue within the Management Port Gateway is assigned a separate RxQ-ID, which is necessary for maintaining ordering when interleaving packets across multiple stacks. See Section 8.2.4.3.
- See Section 8.2.4.1 for details of credit buffers that are required in each Rx queue.

- RxQ-ID values are either 0 or 1. A value of 0 is used if only one stack is negotiated for management transport (regardless of the stack-id value negotiated) and values of 0 and 1 are used if two stacks are negotiated for management transport. In the latter case, an RxQ-ID value of 0 is used for Stack 0, and an RxQ-ID of 1 is used for Stack 1.

When FDI transitions to active state (**pl_state_sts**=Active) from reset state (**pl_state_sts**=Reset) and Management transport was negotiated, the Management Port Gateway starts the Initialization phase. In a multi-stack implementation where management transport is present on both stacks, the D2D adapter flit negotiation, and protocol negotiation across both stacks must have completed (as indicated by the **dm_param_exchange_done** signal) before the Management Port Gateway starts its initialization sequence.

The initialization flow follows the similar sequence as sideband and some example flows are illustrated below. The credit exchange is not by way of an explicit message as in sideband, but rather by way of a dedicated DWORD, 'CRD', in management flits whose format is shown in Figure 8-53 and further explained in Chapter 3.0. Management Port Gateway Capabilities and Init Done Message formats for the mainband can be seen in Section 8.2.2.2.2. Note that during initialization, the transmitter can return valid credits in the same Management Flit that carries the Init Done message. All protocol layer bytes in the management flit (minus the CRD and Rsvd bytes) carrying the 'Init Done' MPM are driven with NOPs after the 'Init Done' MPM.

In Figure 8-46, Figure 8-47, and Figure 8-48, the labeling

Mgmt_Flit {<MPM>, CRD[<credits returned>]}

refers to a Management Flit that carries the specified MPM along with credit returns for the indicated credit types. For example:

Mgmt_Flit {Init Done, CRD[RxQ-ID=0: VC0-Request, VC0-Response]}

indicates a Management Flit that carries the Init Done message along with credit returns for the VC0 request and response credit types for RxQ-ID=0.

Figure 8-46. Mainband Management Transport Initialization Phase Example with RxQ-ID=0 and One VC (VC0)

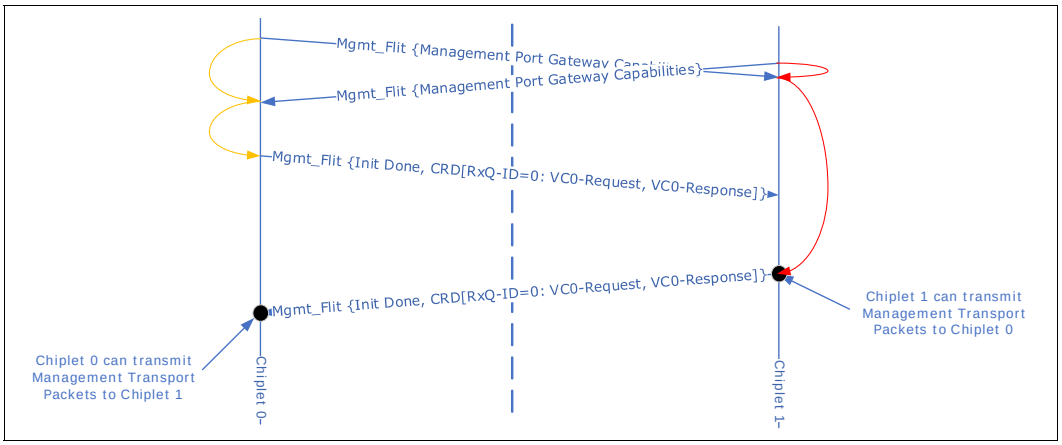


Figure 8-47. Mainband Management Transport Initialization Phase Example with RxQ-ID=0, 1 and One VC (VC0)

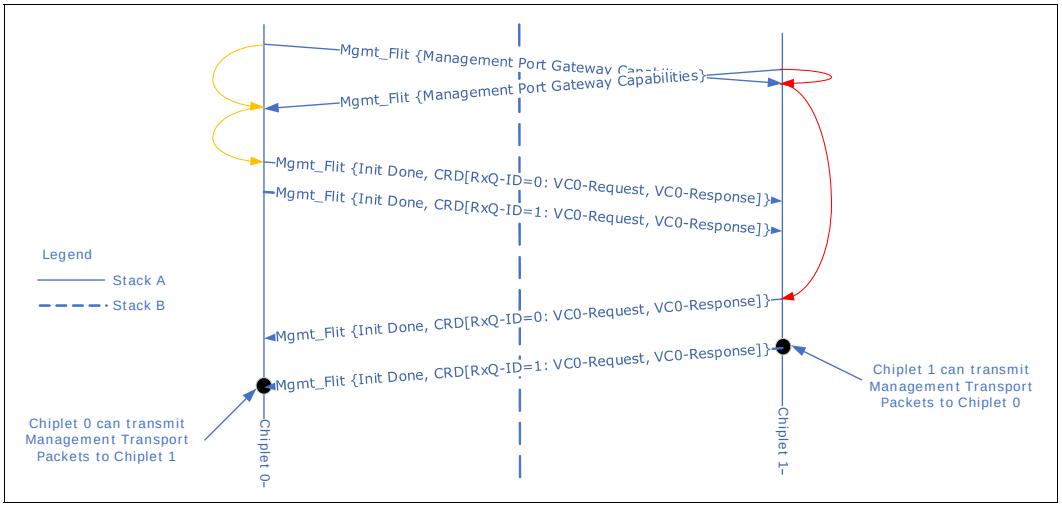
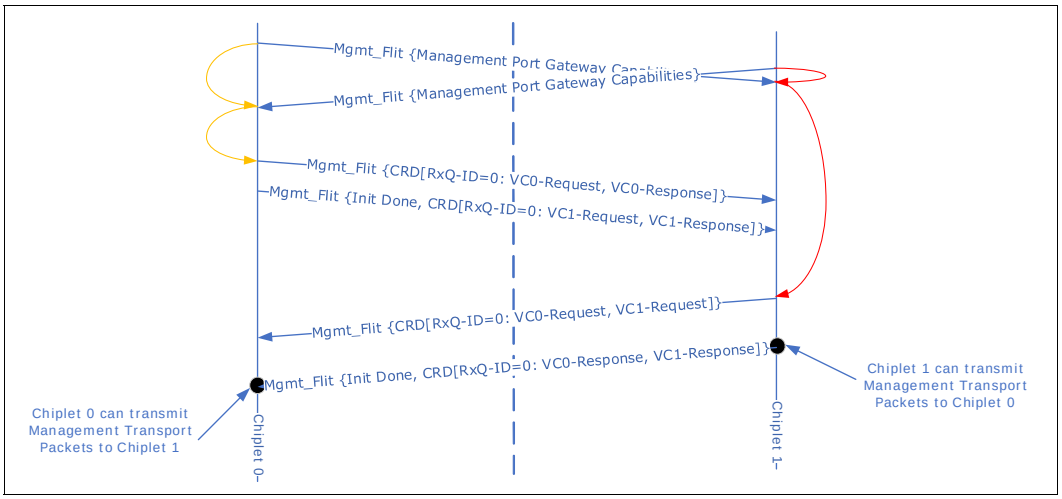


Figure 8-48. Mainband Management Transport Initialization Phase Example with RxQ-ID=0 and Two VCs (VC0, VC1)



8.2.3.2.3 Other Mainband Management Transport Path Rules

The following rules relate to Management Port Gateways and mainband Management Transport:

- During runtime, if the FDI status on any stack that has management traffic negotiated moves to a Link Status=down state, the Management Port Gateway behaves the same as in the ‘Init Done’ timeout scenario (see Section 8.2.4.4) across both stacks, if more than one stack had management transport negotiated.
- Arbitration between Management Flits and regular Protocol Layer Flits is implementation-specific.
- When management Software writes 1 to the ‘Retrain link’ bit in the Management Port Structure register that corresponds to the mainband link, the mainband is retrained, similar to when SW writes 1 to the ‘Start UCIe Link Training’ bit in the UCIe Link Control register in the UCIe link DVSEC. Note that this retraining of the mainband does not affect the management path on the sideband (if that path had been negotiated), if the path was already set up and active.

8.2.4 Common Rules for Management Transport over Sideband and Mainband

8.2.4.1 Management Packet Flow Control

The rules for management transport flow control are as follows:

- Forward progress and Flow Control are managed by the Management Port Gateways.
- Flow control credits are independent for sideband and mainband paths, if both are implemented in a given UCIe port.
- Encapsulated MTPs are credited, and the credits cover both the header and payload portions of the encapsulated MTP.
- Management Port Gateway Capability message, Credit return messages, Init Done messages, and PM-related messages must sink unconditionally at the destination.
- Although the number of VCs supported in both directions is the same, TC-to-VC mapping can be different in each direction. See the Route Entry description in Section 8.1.3.6.2.2 for how SW controls mapping of TC to VC.
- For each RxQ-ID in the Management Port Gateway:
  - Independent credit management is required for each resp type (Requests vs. Responses), and each supported VC.
  - Credits are in QWORD (64-bit) granularity (i.e., one credit corresponds to one QWORD of storage space at the receiver buffer).
  - Minimum three credits are required for each credit type when nonzero credits are advertised.
  - Header and Data portions of an Encapsulated Management Transport packet and Vendor-defined Management Port Gateway Messages use the same type of credit.
  - Receiver implements separate buffers for Requests and Responses per supported VC and advertises the corresponding credits to the remote Management Port Gateway during initialization. Credits are returned when space is freed up in the receiver buffers.
  - Up to eight VCs are permitted — different VC counts are permitted on sideband vs. mainband.
    - Support for VC0 is mandatory for all implementations.
    - For every VC supported, it is mandatory to initialize credits for Request types and Response types.
    - Credits advertised for a VC:Resp credit type during the initialization phase can be either 0 across all RxQ-IDs or nonzero across all RxQ-IDs.
    - If a VC is initialized, credits for that VC must be advertised on all enabled RxQ-IDs and Resp types. For example, it is NOT permitted to have a configuration where VC1 is supported on RxQ-ID 0 but not on RxQ-ID 1. However, it is not required to advertise the same number of credits on all enabled Paths. This rule is important to simplify Transmitter/Receiver implementations at Management Port Gateways for interleaving MTPs across multiple Links while maintaining ordering across them (see 1.4.3 for concept of interleaving).
  - During management transport initialization and before the Init Done message is received, if multiple credit returns are received for the same VC:Resp credit type, the value from the latest credit return overwrites the previous value.
- Number of RxQs (in the partner chiplet's Management Port Gateway) to which a Management Port Gateway can transmit management messages is always same as the number of RxQs to which the MPG can receive these messages (from the partner chiplet's Management Port Gateway). For

example, if two RxQs were negotiated, both send and receive of management traffic must be on two RxQs.

- If the initial credit advertised was infinite for a credit type, there cannot be any credits returned for that type at run time (i.e., after the Init Done message has been sent), with one exception for the “VC0 request infinite credit” scenario for which a runtime credit return of 0 is permitted.
- Credits advertised during the initialization phase are the maximum number of credits that can be outstanding at the transmitter at any point during runtime.
- See Section 8.2.4.3 for the rules for maintaining ordering when interleaving MTPs/MTP Segments across different RxQ-IDs.
- Chiplets can optionally check for the following error conditions during management path initialization flow, and abort the flow when these conditions are detected:
  - Receiving credit returns for more RxQ-IDs than what was negotiated in the Negotiation Phase.
  - Receiving credit returns for more VCs than what was implicitly negotiated by the Management Port Gateway Capabilities message.
  - Not receiving credit returns or receiving incomplete credit returns for any of the negotiated RxQ-IDs prior to receiving the ‘Init Done’ message for the RxQ-ID.

8.2.4.2 Segmentation

The Management Port Gateway is permitted to break up (i.e., segment) one large MTP and send the individual segments across multiple RxQ-IDs (i.e., interleave; see Figure 8-49 for an example). This is useful for cases in which the MTP message sizes are asymmetric. When segmenting:

- Management Port Gateway sets the s bit in the Encapsulated MTP message header within each individual segment except the last segment that completes the MTP transfer. If an MTP is not segmented, the s bit is 0. Segments with the s bit set to 1 must not also have the p bit set to 1.
- Transmitter must ensure that no other Encapsulated MTP OR no other credited MPM packet (e.g., Vendor-defined Management Port Gateway messages), from the same VC:Resp credit type is interleaved until the segmented management packet completes.

Note that segmentation is visible only from Management Port Gateway-to-Management Port Gateway and is not end-to-end on the UCIe Management Fabric.

See Section 8.2.4.3 for the rules for reassembling the segments and maintaining ordering when interleaving Segments across different RxQ-IDs.

Figure 8-49. Example Illustration of a Large MTP Transmitted over Multiple RxQ-IDs on Sideband with Segmentation[a]

Management Transport Packet (MTP)	
QWORD 0	MTP Header
QWORD 1	MTP Data 0
QWORD 2	MTP Data 1
QWORD 3	MTP Data 2
QWORD 4	MTP Data 3
QWORD 5	MTP Data 4
QWORD 6	MTP Data 5
QWORD 7	MTP Data 6
QWORD 8	MTP Data 7
QWORD 9	MTP Data 8
QWORD 10	MTP Data 9
QWORD 11	MTP Data 10
QWORD 12	MTP Data 11
QWORD 13	MTP Data 12
QWORD 14	MTP Data 13
QWORD 15	MTP Data 14

1st Segment[b] — This goes on RxQ-ID=x	
QWORD 0	MPM Header (s = 1, length = 6h)
QWORD 1	MTP Header
QWORD 2	MTP Data 0
QWORD 3	MTP Data 1
QWORD 4	MTP Data 2
QWORD 5	MTP Data 3
QWORD 6	MTP Data 4
QWORD 7	MTP Data 5
2nd Segment[b] — This goes on RxQ-ID=MOD((x+1)/N)	
QWORD 8	MPM Header (s = 1, length = 6h)
QWORD 9	MTP Data 6
QWORD 10	MTP Data 7
QWORD 11	MTP Data 8
QWORD 12	MTP Data 9
QWORD 13	MTP Data 10
QWORD 14	MTP Data 11
QWORD 15	MTP Data 12
3rd Segment[b] — This goes on RxQ-ID=MOD((x+2)/N)	
QWORD 0	MPM Header (s = 0, length = 1h)
QWORD 1	MTP Data 13
QWORD 2	MTP Data 14

a. N = Number of RxQ-IDs negotiated.
x = Start value of RxQ-ID for an MTP.
b. A segment is an Encapsulated MTP with its s bit set to 1.

8.2.4.3 Interleaving and Multi-module Sideband and Multi-stack Mainband Ordering

When multiple RxQ-IDs are negotiated, the Management Port Gateway must interleave different MTPs of a given VC:Resp credit type across the different RxQ-IDs. For example, when the transmitter does not support Segmentation (see Section 8.2.4.2), if there are two MTPs, Pkt 1 and Pkt 2, both on VC0 and of Resp=0 type and two RxQ-IDs were negotiated, these must be sent on two different RxQ-IDs. This is called interleaving. When the transmitter supports Segmentation, individual Segments are also interleaved. This section discusses transmitter and receiver rules when interleaving so that the original management packet ordering (see Section 8.1.3.1.1) is maintained when the MTPs eventually make it to the management network on the receiving partner chiplet. For the purposes of discussing these rules in this section, the nomenclature of RxQ-IDx:VCy:Respz is used to refer to the credit buffer of RxQ-ID=x (x=0 to 3), VCy (y=0-7) and Respz (z=0 for Request and 1 for Response) type.

8.2.4.3.1 Transmitter Rules

- First Encapsulated MTP after Management path setup, of a given VCy:Respz credit type is transmitted to the RxQ-ID0:VCy:Respz credit buffers of the partner chiplet.

  - When the MTP is not segmented, the MTP is fully transmitted to the associated credit buffers and this could take multiple Encapsulated MTPs. In that scenario, each Encapsulated MTP carries the same MPM header but with the length field adjusted for the data length in that message. cr_ret_* fields are also refreshed in every Encapsulated MTP (on the sideband) and indicate 0 if there is no new credit to return. On the mainband path, credits can be refreshed every management flit.
  - RxQ-ID is incremented by 1 for transmitting the next MTP of the same VCy:Respz credit type (i.e., the next MTP of VCy:Respz credit type is sent to RxQ-ID1:VCy:Respz credit buffers).
  - When the MTP is segmented, a single Encapsulated MTP belonging to the MTP is transmitted to the associated buffers with the “s” bit set to 1. RxQ-ID is incremented by 1 (with wraparound as indicated later in this section) for transmitting each subsequent segment of the same MTP until the MTP is fully sent. After the MTP is fully sent, the RxQ-ID is incremented by 1 again (with wraparound as indicated later in this section) for transmitting the next MTP of the same VCy:Respz credit type.
- The above scheme is repeated independently for traffic within each VCy:Respz credit type. Transmission of packets on different VCy:Respz queues have no dependencies between them.
- RxQ-ID value wraps around after the maximum-negotiated RxQ-ID.
- Transmission to multiple RxQ-ID buffers can occur in parallel on sideband links or mainband stacks.

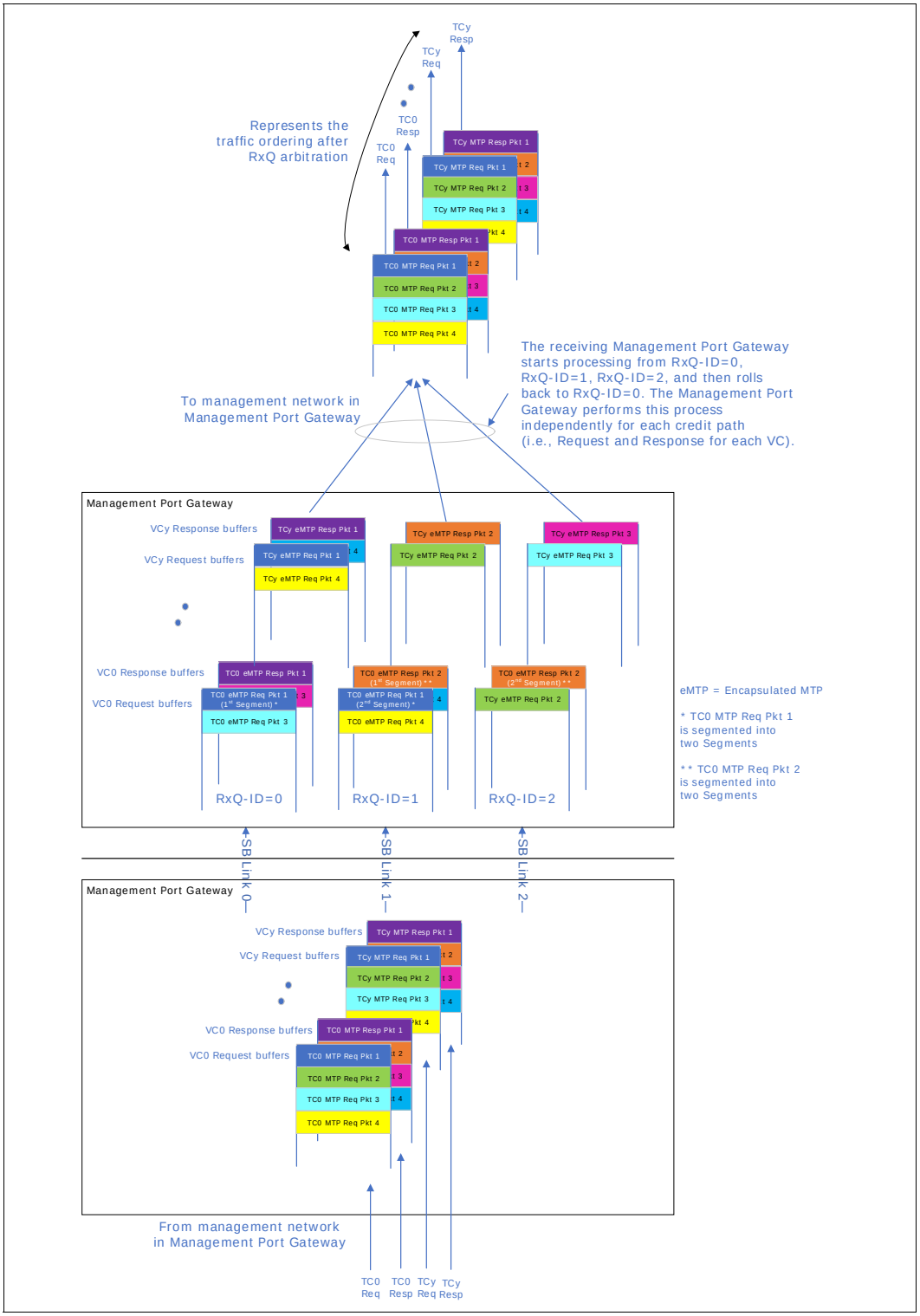
8.2.4.3.2 Receiver Rules

- On the Rx side, after a Management path setup, the Management Port Gateway services a full MTP (or in the case of Segmentation, one Encapsulated MTP of a MTP) on RxQ-ID0:VCy:Respz queue for a given VCy:Respz credit type. Note that receiving a full MTP could take multiple Encapsulated MTPs.
  - Gateway then services the next MTP (or in case on Segmentation, the next Encapsulated MTP of the management packet) on the RxQ-ID1:VCy:Respz queue, and then on the RxQ-ID2:VCy:Respz queue (if supported), etc.
  - In case of segmentation, the receiver can look at the “s” bit being cleared to 0 (from being set to 1 in prior segments) to know the last segment of an MTP.
  - RxQ-ID value wraps around after the maximum-negotiated RxQ-ID.
- The above receiver scheme applies independently for each VCy:Respz credit type and there are no dependencies between them.
- Messages that do not consume credits must not be allocated into the credited Rx queues (credit returns, PM wake/ack/sleep messages) — and are unconditionally consumed by the Receiver.

Figure 8-50 illustrates the ordering mechanism for an example scenario with three RxQ-IDs, and y VCs (where y=0-7) on the sideband. For the purposes of this illustration — TC0 management port traffic is mapped to VC0 on the sideband management path. TCy management port traffic is mapped to VCy on the sideband management path. Note that in the figure, TC0 Req Pkt 1 and TC0 Resp Pkt 2 are segmented to two segments, to show the impact of segmentation on interleaving and ordering. Other MTPs are not segmented. Similar ordering applies for packets that are interleaved over multiple stacks on the mainband.

Vendor-defined Management Port Gateway messages also use the same credited buffers as MTPs. Transmitter and receiver interleaving rules for these messages are the same as discussed earlier for Encapsulated MTPs.

Figure 8-50. Conceptual Illustration of Sideband Multi-module Ordering with Three RxQs



8.2.4.4 'Init Done' Timeout Flow

During the Management Transport Initialization Phase, a 16-ms timeout (also referred to as 'Init Done' timeout) is applied for receiving an "Init Done" MPM from the start of initialization, across all available RxQ-IDs. If an 'Init Done' timeout occurs:

- Management Port Gateway cannot schedule any new MPMs across any RxQ-ID and the MPG silently discards any MPMs received, and resets all the RxQ-ID credit counters and pointers.
- Management Port Gateway indicates this status to management FW by way of the management port capability structure (see Section 8.1.3.6.2.1) and waits for SW to retrigger management path retraining.

8.2.5 Other Management Transport Details

8.2.5.1 Sideband

8.2.5.1.1 Management Port Gateway Flow Control over RDI

See Section 7.1.3.1 for details.

8.2.5.1.2 MPMs with Data Length Rules

When supporting MPMs with Data (see Section 7.1.2.4) over the sideband, to prevent these messages from occupying the sideband interface for extended periods of time (and thus blocking its usage for mainband link management packets), the following rules must be observed:

- An MPM with Data (e.g., Encapsulated MTP) can have a maximum length field value of seven QWORDs
- Receivers must not check for violation of this transmit rule.
- If the original MTP was larger than seven QWORDs, multiple Encapsulated MTPs are sent until the full MTP is transmitted. It is also permitted to send Encapsulated MTPs smaller than seven QWORDs even when the original MTP is larger than seven QWORDs. This can occur because of credit availability for transmitting the Encapsulated MTP.
- The above rules allow for the link to be arbitrated for any pending Link management packet OR any pending higher priority MEM packet of a different VC:Resp credit type (waiting behind an MPM with Data that is in transmission) with an upper bound on the delay to transmit them. An example of a higher-priority MPM packet that needs to be serviced in a time-bound fashion is a TC1 MTP (see Section 8.1.3.1.1).
- Segmentation, when performed, must follow the rules described above for each individual Segment of the MTP. See Section 8.2.4.2 for description of segmentation.

Figure 8-51 provides a pictorial representation of splitting a large MTP into multiple smaller Encapsulated MTPs (based on the length rules stated above) and how the Encapsulated MTPs are sent on the sideband link. If the MTP is also segmented, then each Encapsulated MTP is sent on a different RxQ-ID. See Section 8.2.4.2 and Section 8.2.4.3.

See Section 4.8 for how the PHY arbitrates between MPMs and Link Management packets.

Figure 8-51. Example Illustration of a Large MTP Split into Multiple Smaller Encapsulated-MTPs for Transport over Sideband, without Segmentation[a]

Management Transport Packet (MTP)	
QWORD 0	MTP Header
QWORD 1	MTP Data 0
QWORD 2	MTP Data 1
QWORD 3	MTP Data 2
QWORD 4	MTP Data 3
QWORD 5	MTP Data 4
QWORD 6	MTP Data 5
QWORD 7	MTP Data 6
QWORD 8	MTP Data 7
QWORD 9	MTP Data 8
QWORD 10	MTP Data 9
QWORD 11	MTP Data 10
QWORD 12	MTP Data 11
QWORD 13	MTP Data 12
QWORD 14	MTP Data 13
QWORD 15	MTP Data 14

SB Encapsulated MTP 0 — This goes on RxQ-ID=x	
QWORD 0	MPM Header (s = 1, length = 6h)
QWORD 1	MTP Header
QWORD 2	MTP Data 0
QWORD 3	MTP Data 1
QWORD 4	MTP Data 2
QWORD 5	MTP Data 3
QWORD 6	MTP Data 4
QWORD 7	MTP Data 5
SB Encapsulated MTP 1 — This goes on RxQ-ID=x	
QWORD 8	MPM Header (s = 1, length = 6h)
QWORD 9	MTP Data 6
QWORD 10	MTP Data 7
QWORD 11	MTP Data 8
QWORD 12	MTP Data 9
QWORD 13	MTP Data 10
QWORD 14	MTP Data 11
QWORD 15	MTP Data 12
SB Encapsulated MTP 2 — This goes on RxQ-ID=x	
QWORD 0	MPM Header (s = 0, length = 1h)
QWORD 1	MTP Data 13
QWORD 2	MTP Data 14

a. N = Number of RxQ-IDs negotiated.
x = Start value of RxQ-ID for an MTP.

8.2.5.1.3 Sideband Runtime Management Transport Path Monitoring — Heartbeat Mechanism

After the management transport path Initialization Phase completes, receiver starts an 8-ms 'Heartbeat' timer that restarts whenever an MPM (i.e., opcode 10111b or 11000b) is received. Implementations are permitted to implement this timer as a global timer across all RxQ-IDs or as a timer per RxQ-ID. If the timer times out, the Management Port Gateway de-asserts the **mp_mgmt_up** signal after 16 ms which in turn clears the SB_MGMT_UP flag in the PHY and de-asserts the **mp_mgmt_port_gateway_ready** signal. After a Heartbeat timeout, Management Port Gateway functions similar to what occurs during a 'Init Done timeout' (see Section 8.2.4.4 for details). The Heartbeat timer stops after L1/L2 entry negotiation on the sideband path successfully completes, and restarts when L1/L2 exit negotiation starts. See Section 8.2.5.1.4 for details of Management path PM entry/exit flows.

After the 'Init Done' message is been transmitted on an RxQ-ID path during the initialization Phase, the Management Port Gateway (MPG) must guarantee an MPM transmission of no more than 4 ms apart on the RxQ-ID path. If there are no scheduled messages to send on an RxQ-ID path, the MPG must send a credit return message (unless there was a Heartbeat timeout on the receiver side as stated in the previous paragraph) with VC value set to VC0, Resp value set to 0 and cr_ret value set

to 0h. Note that the latter applies even if the MPG takes longer than 8 ms to exit L1/L2 before the MPG sends the associated PM exit message.

If a control parity error is detected on any received MPM, the Management Port Gateway invokes the 'Heartbeat timeout' flow.

8.2.5.1.4 Sideband Management Path Power Management Rules

On the sideband interface, it is expected that there is higher-level firmware/software managing the deeper power states of Management Port Gateways on both sides. The sleep and wake req/ack/nak messages (see Figure 7-12) are provided to negotiate shutdown/wake of the management transport path for deep power states in which the Management Port Gateway logic can be clock gated or powered down (as coordinated by the higher-level firmware). It is especially useful for a low-power chiplet and/or SiP states flows to take advantage of these handshakes and coordinate entry and wake up of the Management Transport Path. These messages and negotiation must occur independently for each RxQ-ID path, and each direction. While not in a PM state, the Management Port Gateway must keep the **mp_wake_req** signal asserted and this informs the Physical Layer adapter to keep the logic up and running.

8.2.5.1.4.1 Sideband PM Entry Rules

- Management Port Gateway Transmitter that initiates the PM entry ensures that no other packets will be transmitted (other than credit returns and PM messages) on any of the enabled RxQ-ID paths.
- Following the above, the Transmitter sends a "Sleep req" message on each of the RxQ-ID paths. After a "Sleep req" message is sent on an RxQ-ID path, only credit return messages can be transmitted on the path until a "Sleep ack" message is received on the path or a Sleep ack timeout occurs (see last bullet in this section below). If the former scenario, additional message transmissions are not permitted until the subsequent PM exit. In the latter scenario, message transmission can resume soon after the timeout.
- After receiving a "Sleep req" message, the receiving Management Port Gateway must ensure that the corresponding Rx buffer is empty, and that all pending credit returns have been sent to the remote Link partner. After these conditions are met, a "Sleep ack" message is scheduled.
- If a chiplet responded with a "Sleep ack" message, the chiplet must send a "Sleep req" message (if not already sent) within 16 ms of sending the "Sleep ack" and receive a response to complete the flow; otherwise, sleep entry is aborted.
- After a Management Port Gateway has sent and received a "Sleep Ack" message on all paths, the MPG is permitted to clock gate or power down, etc. The Management Port Gateway must de-assert the **mp_wake_req** signal before entering the clock gated or power down state.
- If Sleep Nak was sent or received, the sleep entry is aborted.
- Transmitter of a "Sleep req" message can wait for an implementation-dependent timeout to receive a "Sleep ack" before aborting the flow. In multi-module implementations, the "Sleep ack" message must be received across all negotiated RxQ-ID paths before the timeout.

8.2.5.1.4.2 Sideband PM Exit Rules

- Management Port Gateway Transmitter that initiates the exit performs the **mp_wake_req/ack** handshake with its Physical Layer and schedules the "Wake req" message on each RxQ-ID path.
- Partner chiplet's Physical Layer that receives the "Wake req" message over the sideband wakes up its Management Port Gateway by performing the **pm_clk_req/ack** handshake before transmitting the "Wake req" message in response.

- After the partner chiplet's Management Port Gateway receives the "Wake Req" message, that Management Port Gateway must respond with a "Wake ack" message when the MPG is ready to receive credited packets into its Rx buffer. Moreover, the Management Port Gateway must initiate its own "Wake req" message to the remote Link partner if the MPG has not already done so.
- After a "Wake ack" message is sent and received on all negotiated RxQ-IDs, the PM exit flow is complete and regular packet transfer can begin as soon as the last "Wake ack" message is transmitted.

8.2.5.1.5 Management Port Gateway Mux Arbitration

There is no prescribed arbitration mechanism for the Management Port Gateway mux on the Sideband. Additionally, the size of Management Port Gateway Flow control buffers over RDI (see Section 8.2.5.1.1) is not specified for Management Port Gateway-initiated traffic. Implementations should take care to ensure that the PHY arbitration rules specified in Section 4.8 are not violated.

8.2.5.2 Mainband

8.2.5.2.1 NOP Message

The Management Port Gateway inserts NOP messages whose format is shown in Figure 8-52, in all QWORD locations in a Management flit in which there is no MPM to send. NOP messages can start only at MPM boundaries within a flit.

Figure 8-52. Management Flit NOP Message on Mainband

3								2								1								0							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0000_0000h																															
0000_0000h																															

8.2.5.2.2 Credit Return DWORD Format

Figure 8-53. Management Transport Credit Return DWORD (CRD) Format on Mainband

3								2								1								0							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cr_ret_resp_a	cr_ret_ vc_a			cr_ret_a										rsvd		cr_ret_resp_b	cr_ret_ vc_b			cr_ret_b										rsvd	rxqid

See Section 3.3.3 and Section 3.3.4 for details on where this DWORD is sent in a Management Flit for various Flit formats.

The following rules apply:

- rxqid field in the DWORD applies to both credit returns a and b shown in Figure 8-53
- During VC initialization, on the Management Flit that carries the Management Port Gateway Capability message, all credit return fields must be set to 0
- If there is no credit to return in credit return slots a or b (as shown in Figure 8-53), a value of 0 is used for all associated credit return fields
- If credit returns a and b carry the same vc:resp fields, then the total credit returned for that rxqid:vc:resp credit type is the sum of cr_ret_a and cr_ret_b

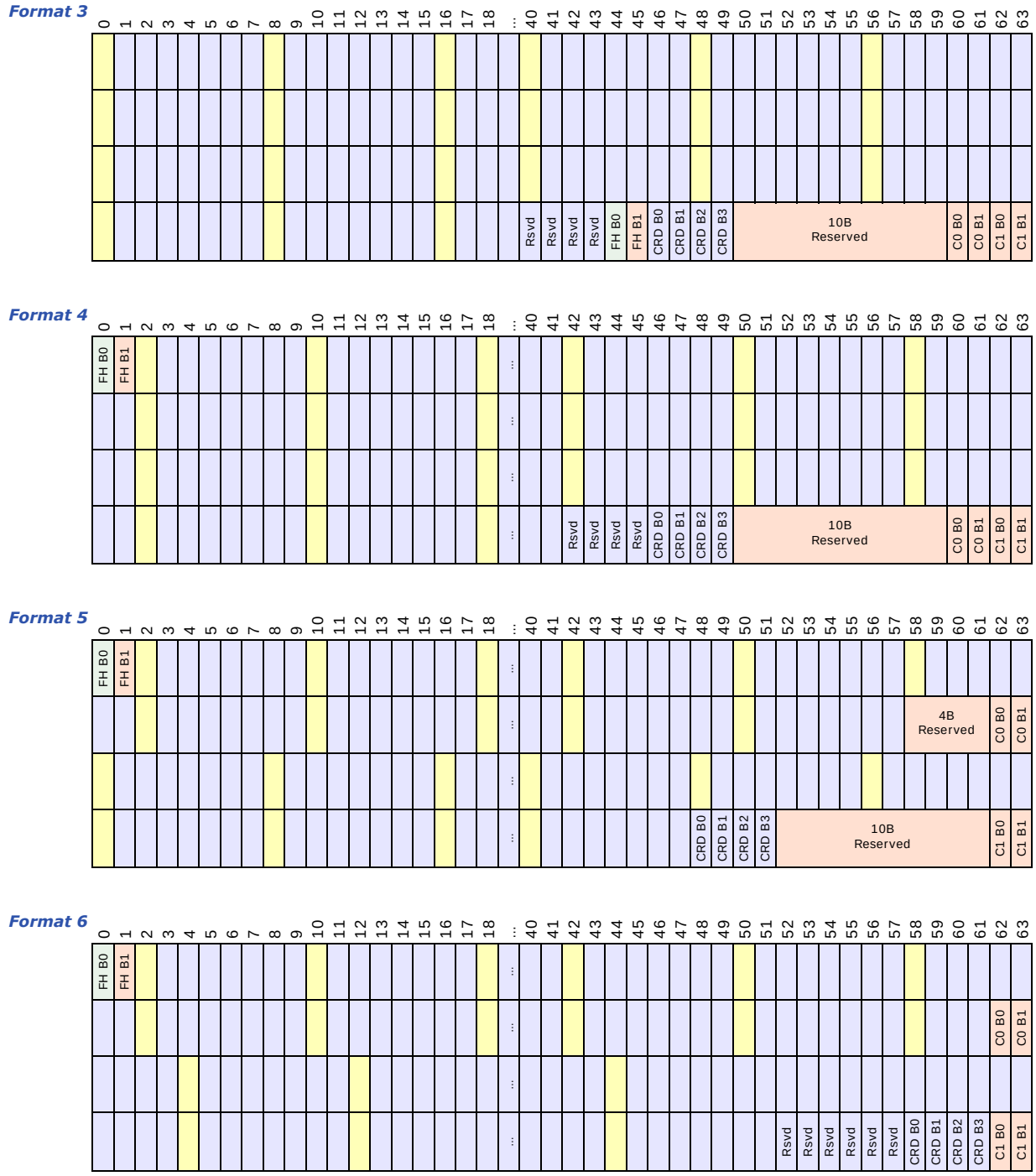
8.2.5.2.3 Management Flit Formats

On the mainband, MPMs are supported only over Flit *Format 3* through *Format 6*.

See Section 3.3.3 and Section 3.3.4 for a D2D view of the Management Protocol mapping over Flit *Format 3* through *Format 6*. If Flit *Format 1* and *Format 2* are negotiated, the Management Protocol on that stack is disabled (if supported). Management flits have bits [7:6] of Byte 1 set to 10b. See Section 8.2.2.2 for packet format of MPMs over the mainband. Mapping of these MPMs over Flit *Format 3* through *Format 6* is as follows:

- MPM header and each QWORD of MPM payload (when applicable) can be placed only at specified byte locations in the Management flit, and can start at the 1st byte in the Management flit in which “all bits are populated by protocol layer” (see Figure 2-1 for reference), and at subsequent 8B increments within the flit. While incrementing, only bytes in which “all bits are populated by the Protocol Layer” are considered, excluding CRD byte locations and bytes marked as rsvd for Protocol Layer (e.g., Flit *Format 3*, Bytes 40 through 43). This is pictorially shown in Figure 8-54.

Figure 8-54. Valid MPM Header Start Locations for Various Flit Formats[a] [b]

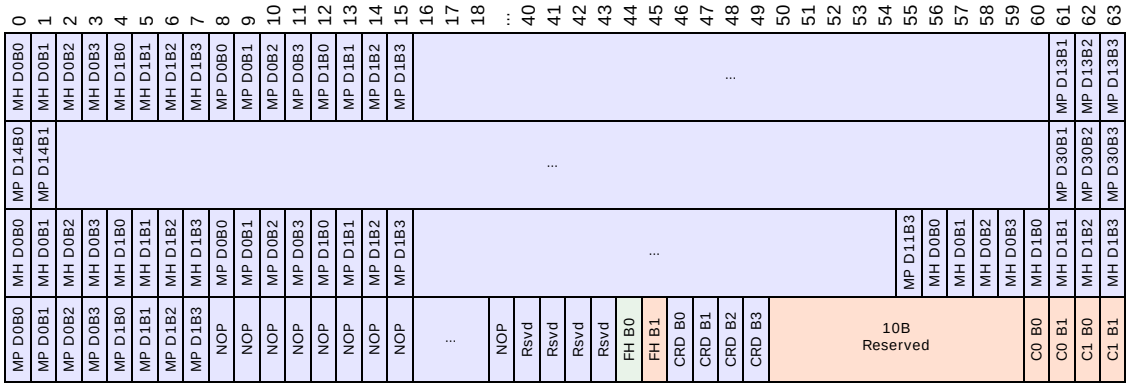


a. Yellow cells indicate a Valid Management Port Message (MPM) header or Payload QWORD start location. For the other colors, see Figure 2-1 for color mapping.
b. B = Byte, C = CRC, CRD = Credit Return DWORD, FH = Flit Header, Rsvd = Reserved. All are numbered, as appropriate, except for Rsvd.

Starting at a valid MPM header byte location (as discussed above), Byte 0 of the first DWORD of the MPM header is sent at that byte, followed by Byte 1 of the first DWORD of the header at starting byte location+1 until Byte 3 of the 2nd DWORD of the header. This is followed by Byte 0 of the 1st DWORD of the MPM payload (if one exists), followed by Byte 1, Byte 2, Byte 3, etc., placed at incrementing byte locations. Non-CRD bytes, bytes that are not marked as reserved and those that are driven by the protocol layer are contiguously packed with MPM bytes after an MPM transmission starts and until the transmission ends. If an MPM cannot be fully transmitted within a Management Flit, the MPM continues in the subsequent Management Flit of the same stack. NOP message(s) (see Section 8.2.5.2.1) can be inserted between MPMs within a Management Flit. It is also valid to send a Management Flit with all NOP messages in the protocol layer-driven non-CRD bytes and non-reserved byte locations. CRD bytes in a Management Flit always carry the credit return information per the rules stated in Section 8.2.5.2.2.

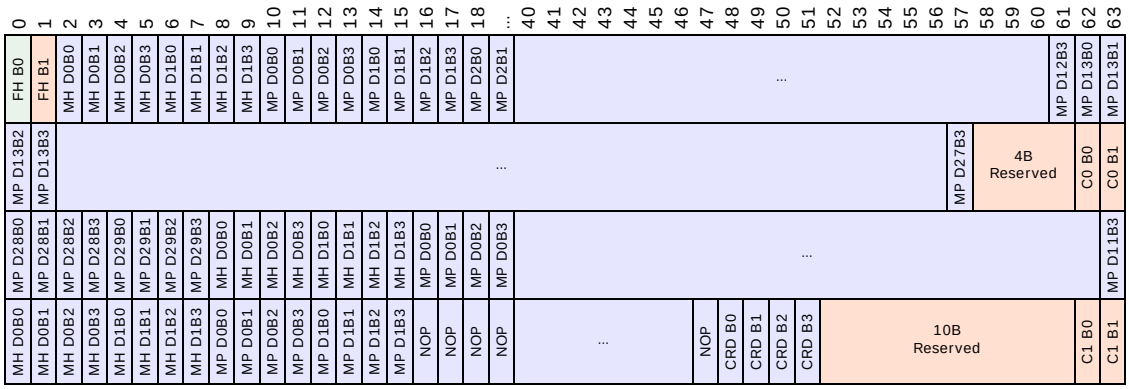
Figure 8-55 and Figure 8-56 show example mappings of three MPMs inside Flits of *Format 3* and *Format 5*, respectively. The 1st MPM is of an MPM with Data type with a payload size of 15 QWORDs. The 2nd MPM is also of an MPM with Data type with a payload size of 6 QWORDs. The 3rd MPM is an MPM with a payload of size of 1 QWORD. NOPs are inserted after the end of the 3rd MPM until the end of the flit.

Figure 8-55. Example Mapping of MPMs and NOPs in Flit of Format 3[a] [b]



a. See Figure 2-1 for color mapping.
b. B = Byte, C = CRC, CRD = Credit Return DWORD, D = DWORD, FH = Flit Header, MH = MPM Header, MP = MPM Payload, NOP = No Operation, Rsvd = Reserved.

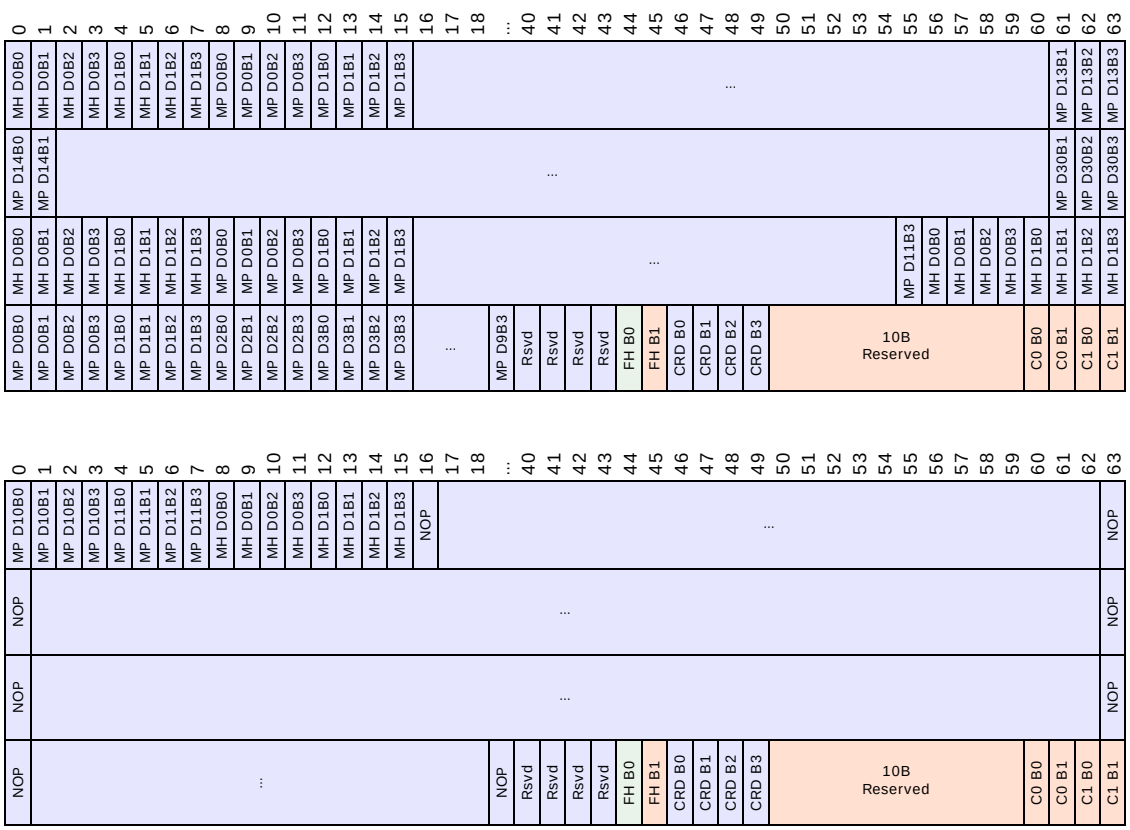
Figure 8-56. Example Mapping of MPMs and NOPs in Flit of Format 5 [a] [b]



a. See Figure 2-1 for color mapping.
b. B = Byte, C = CRC, CRD = Credit Return DWORD, D = DWORD, FH = Flit Header, MH = MPM Header, MP = MPM Payload, NOP = No Operation.

Figure 8-57 shows an example mapping of four MPMs inside a Format 3 flit. The 3rd MPM rolls over into the 2nd flit. The 1st MPM in this example is of an MPM with Data type with a payload size of 15 QWORDs. The 2nd MPM is also of an MPM with Data type with a payload size of 6 QWORDs. The 3rd MPM is an MPM with payload size of 6 QWORDs, where the 6th QWORD is sent in the 2nd Flit. The 4th MPM in this example is a 1-QWORD Vendor-defined Management Port Gateway message without data. The remainder of the 2nd flit is all NOPs.

Figure 8-57. Example MPM Mapping to Management Flit for Format 3 with MPM Rollover to Next Flit[a] [b]



a. See Figure 2-1 for color mapping.
b. B = Byte, C = CRC, CRD = Credit Return DWORD, D = DWORD, FH = Flit Header, MH = MPM Header, MP = MPM Payload, NOP = No Operation, Rsvd = Reserved.

8.2.5.2.4 L1/L2 Link States and Management Transport

See Section 3.6 for details.

8.2.5.2.5 Link Reset/Link Disable and Management Transport

For Management Transport on the mainband that has a Management Port Gateway mux, it should be noted that if the associated protocol stack resets or disables the link, the Management Transport path is also reset/disabled. If this is not desired, it is recommended that protocol stacks in such configurations have a way to disable sending link reset and link disable requests on the FDI so that the Management Transport path is not affected.

8.2.6 Retimers and Management Transport

On the sideband, retimers can support management transport if the retimers need to be part of the UCIe management network. This support is optional. If supporting management transport, the retimers must abide by all the requirements stated earlier in this chapter for a generic chiplet implementation. When retimers are part of the management network, the retimers can be fully managed and be able to forward management packets between their UCIe interface and the retimed interface.

On the mainband, retimers can optionally support management transport.

8.3 UCIe Debug and Test Architecture (UDA)

8.3.1 Overview

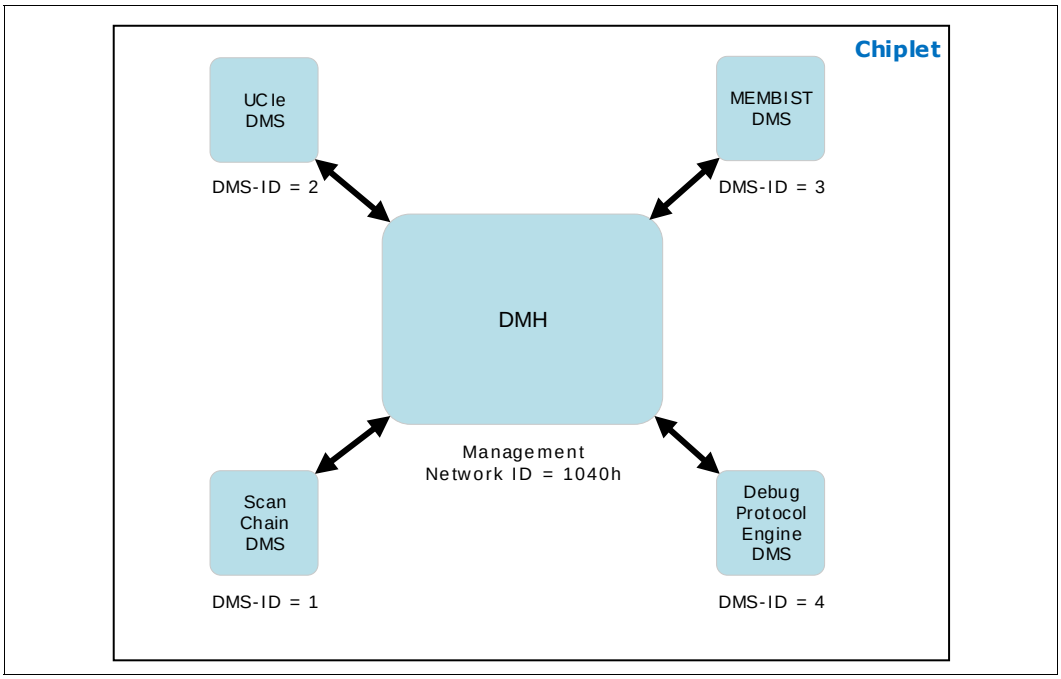
UCIe Debug and Test (DFx) Architecture (UDA) provides a standardized test and debug infrastructure for UCIe-based chiplets and SiPs to enable standard test and debug methods in a UCIe-based open chiplet ecosystem.

UDA is architected on top of UCIe Manageability Infrastructure and uses the architectural elements of that infrastructure for Chiplet-level and SiP-level testing and debug (see Section 8.1 for details of UCIe Manageability Architecture). UDA requires functional UCIe links (Sideband and/or Mainband) and a functional management network for test and debug purposes. Debug and bring-up of UCIe links and elements that comprise the UDA (see Section 8.3.1.1 through Section 8.3.1.4) can be performed by any sideband interface of choice (e.g., JTAG, GPIO, Sideband-only UCIe, etc.) of the chiplet vendor, and are beyond the scope of this specification.

Within each chiplet, UDA is architected in a Hub-Spoke model. In this model, DFx Management Hub (DMH) is the Management Element that implements the Debug and Test Protocol(s). UDA allows for SW/FW to discover debug capabilities present in a chiplet, and provides for global security control/ status for test/debug functionality present in the chiplet. Chiplet test/debug functionality is implemented in DFx Management Spoke (DMS). Some examples of test/debug functionality are Scan controller, Memory BIST, SoC fabric debug, Core debug, trace protocol engine, etc.

In Figure 8-58, there is one DMH with a Management Network ID of 1040h and 4 DMSs connected to it with DMS-IDs (also referred to as Spoke-IDs) from 1 to 4. Management Network ID is used to route DFx and other relevant manageability packets to DMH in the manageability fabric. See Section 8.1.3.2 for how to interpret this ID. DMS-ID is used to route ID-routed Test and Debug packets to the correct Spoke within DMH.

Figure 8-58. UDA Overview in Each Chiplet – Illustration



8.3.1.1 DFx Management Hub (DMH)

Key points about DMH:

- DMH implements a set of registers that allow Management Firmware to:
  - Enumerate test/debug capabilities within the chiplet
  - Globally access control to/from debug functionality of DMS
  - Reliably report status of test/debug functionality usage within the chiplet
- DMH provides appropriate routing of Manageability packets that target various Spokes under it
- There can be multiple DMHs within a chiplet
- DMH can coexist with other protocol entities under the same Management Network ID
- DMH registers are accessed by the UMAP protocol (see Section 8.1.4 for details)

8.3.1.2 DFx Management Spoke (DMS)

Key points about DMS:

- Spokes provide the required test/debug functionality in a chiplet.
  - Some examples of test/debug functionality that can be implemented in a Spoke are MEMBIST, Scan controller, Core debug, SoC internal debug/test, PCIe link debug, UCIe link debug, Trace protocol, etc.
  - Spoke definition is left to the chiplet vendor to decide based on the chiplet's test/debug requirements.
- Spokes support the UMAP protocol and are discovered by SW as discussed in Section 8.3.5.

- Spokes can optionally support Vendor-defined Test and Debug messages, and these messages are routed to the destination Spoke within a DMH using a DMS-ID.
- Valid DMS-IDs for Spokes are from 1 to 254. A value of 0h is assigned for DMH. A value of FFh is reserved.
- DMS-ID is unique within a given DMH.
- DMH provides a pointer to the first DMS in a linked list of DMSs present within the DMH.
- Each Spoke identifies itself as one of these types (see Section 8.3.5.3.2.8 for more details):
  - UCIe.Physical_Layer
  - UCIe.Adapter
  - UCIe.Adapter_Physical_Layer
  - Vendor-defined
- Each Spoke implements a simple standard register set that helps to uniquely enumerate each Spoke and to allow custom SW to be loaded to interact with the Spoke.
  - All Spokes minimally support DWORD Register Rd/Wr accesses. Support for sizes beyond that are optional.
- Vendor-defined sections of the register space can be used for a vendor to implement any Spoke functionality such as triggering BIST, reading internal debug registers, array dump, etc.

8.3.1.3 Supported Protocols

8.3.1.3.1 UCIe Memory Access Protocol (UMAP)

Used to access registers in DMH/DMS. See Section 8.1.4 for details of this protocol.

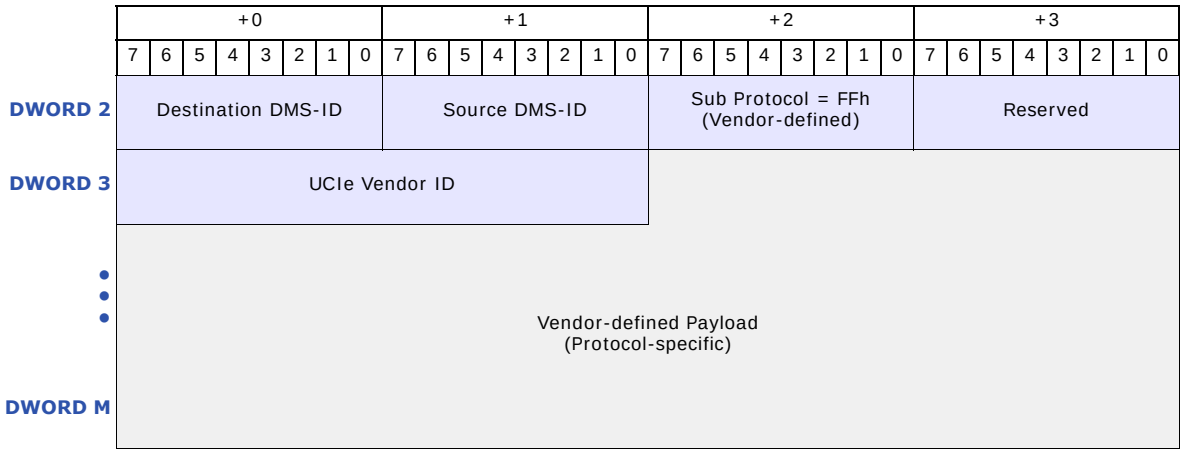
8.3.1.3.2 Vendor-defined Test and Debug Protocol

Used for test as discussed in Section 8.3.2 and Section 8.3.3 and for any other vendor-defined functionality. Format of DWORDS 2 to M of Vendor-defined Test and Debug UCIe DFx Messages (or Vendor-defined UDM, for short) is shown below. DWORDs 0 to 1 of these messages follow the standard format of Management Transport packet described in Section 8.1.3, with the Management Protocol field set to ‘Test and Debug Protocol’. Packet Integrity Protection DWORDs (that appear after DWORD M) are as defined in Section 8.1.

- These messages are routed to the correct Spoke within a DMH, using the Destination DMS-ID field in Byte 8 of the message.
- UCIe Vendor-ID field is the UCIe Consortium-assigned Vendor ID for the Spoke’s IP Vendor

In Figure 8-59, UCIe Vendor ID[15:8] is sent on Byte 0[7:0] of DWORD 3, and UCIe Vendor ID[7:0] is sent on Byte 1[7:0] of DWORD 3.

Figure 8-59. Vendor-defined Test and Debug UDM



> **IMPLEMENTATION NOTE**
>
> A Spoke's support for Vendor-defined UDM is negotiated/discovered using vendor-defined mechanisms. The Spoke Vendor ID and Spoke Device ID can be used to determine a specific Spoke implementation from a specific Vendor. Vendor-defined registers in the Spoke can be used to negotiate/discover the Vendor-defined Payload format of Vendor-defined UDM.

8.3.1.4 UDM and UCIe Memory Access Protocol Message Encapsulation over UCIe

See Section 8.2 for details of how manageability messages (of which UCIe Memory Access Protocol and UDM are two subtypes) are negotiated, encapsulated, and transported over the UCIe sideband and Main band.

8.3.1.5 UCIe Test Port Options and Other Considerations

See Chapter 5.0 for different port options for testing.

8.3.1.5.1 Determinism Considerations

Testing with low-cost ATE typically requires cycle-accurate determinism. When using UCIe as a test port, how the determinism is achieved end-to-end is implementation-specific and beyond the scope of this specification.

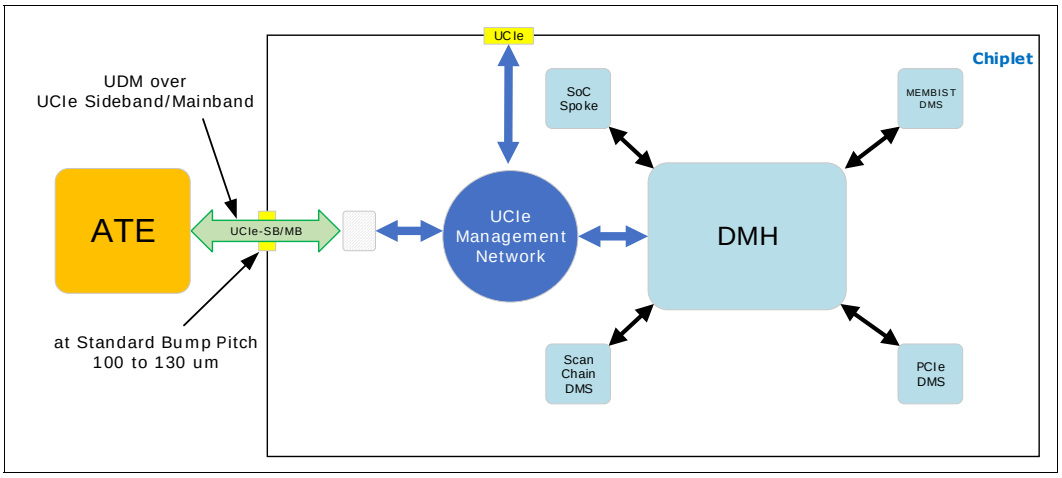
8.3.1.6 DFx Security

See Section 8.1.3.5 for details.

8.3.2 Sort/Pre-bond Chiplet Testing with UDA

This section covers an overview of chiplet-level testing at Sort/Pre-bond using UCIe as the test port. Support for this testing scheme is optional. Figure 8-60 captures this scenario.

Figure 8-60. UCIe-based Chiplet Testing/Debugging at Sort



- UCIe sideband and/or mainband can be used for this testing if they have a bump pitch of 100 um to 130 um.
- For sending/receiving scan test patterns, Vendor-defined UDMs (see Figure 8-59) are used over UCIe Sideband or mainband. These messages can target the appropriate Spoke (using the DMS-ID field) in the design that implements the scan functionality.
- For general-purpose testing/debugging using register reads and writes, UCIe UMAP messages can be used (e.g., for triggering in-build self-test mechanisms in a chiplet, a UCIe register read/write mechanism can be used to trigger a test and then read the test results).

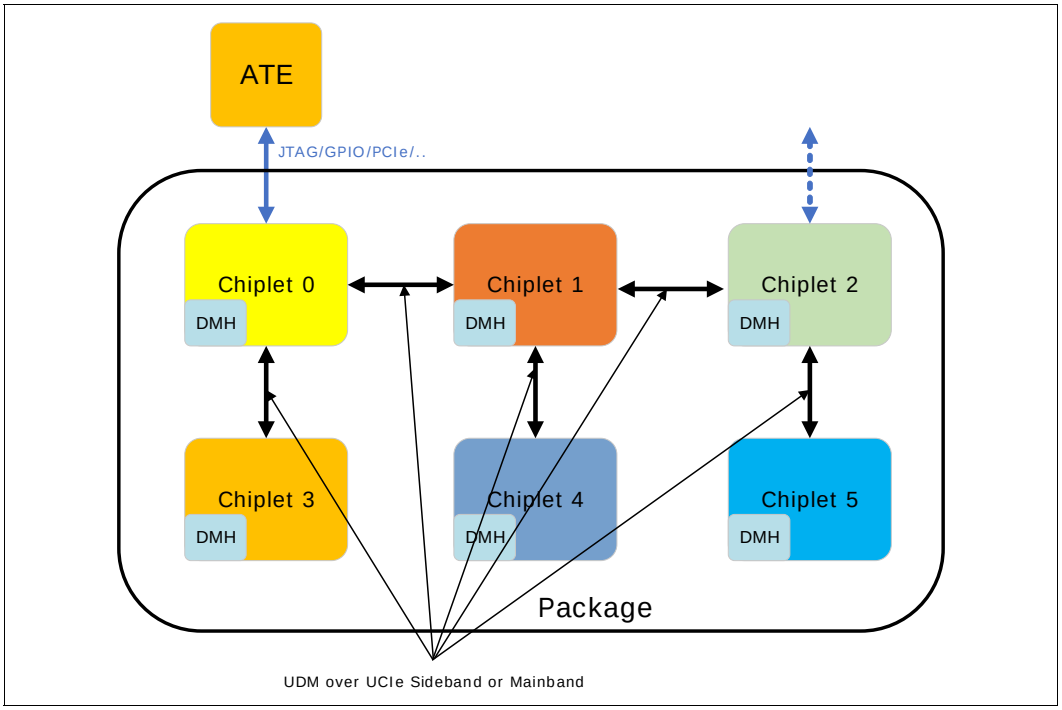
In Figure 8-60, a UCIe Management port embedded in the UCIe controller provides access from the tester to the chiplet's manageability/test/debug fabric. The access control mechanism for ATE to acquire access to the UCIe Management network is implementation-specific.

While the ATE interfaces covered above are UCIe sideband and mainband, other interfaces such as JTAG, GPIO, and PCIe are also possible. Vendors can implement a bridge from these interfaces, with appropriate security control, to the UCIe Management network.

8.3.3 SiP-level Chiplet Testing with UDA

This section covers chiplet-level testing in a package that uses UCIe. Figure 8-61 provides an overview of this. Support for this testing scheme is optional.

Figure 8-61. UCIe-based Testing of Chiplets in an SiP

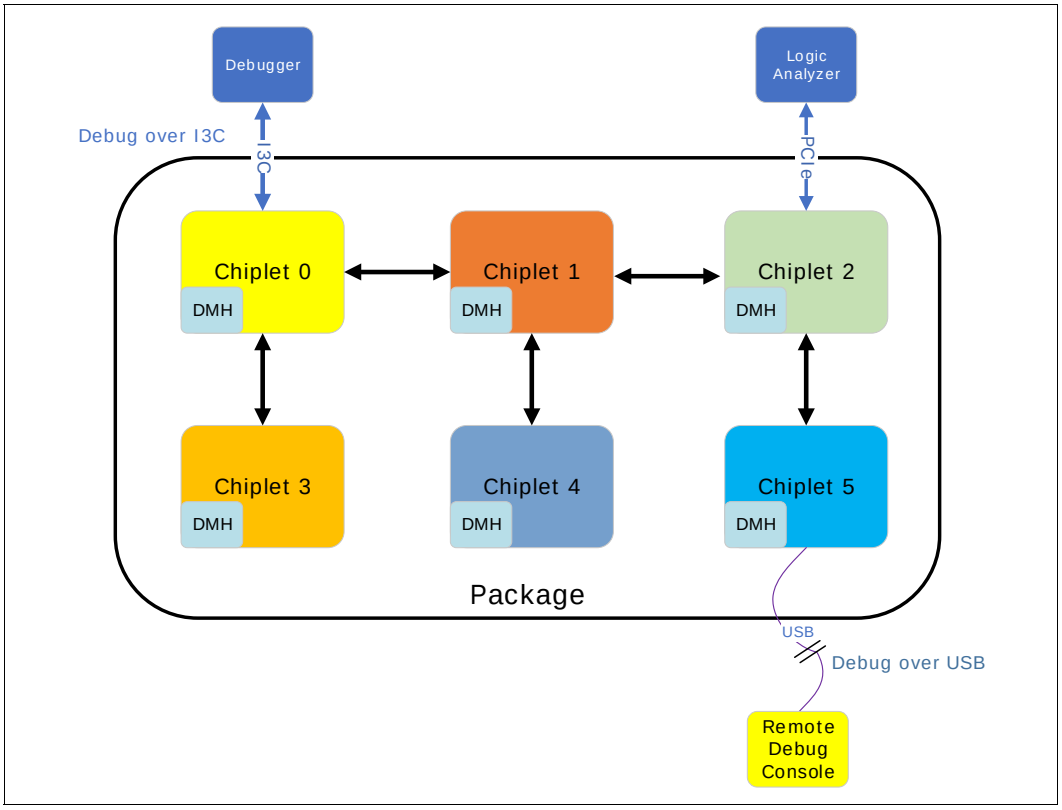


- There is at least one test/debug port pinned out in the package for SiP-level testing/debugging.
  - The port could be any of JTAG, GPIO, PCIe, USB, SMBus, and/or I2(3)C.
- More than one package port can be used for speeding up package-level test/debug.
- Vendors can implement bridges, with appropriate security control, from these interfaces to the UCIe Management network.
  - On the UCIe Management network, bridged packets follow the UCIe Management Transport Packet format.
- Accesses from package ports are forwarded over UCIe sideband or mainband if they target other chiplets. See Section 8.1.3.2 for details of how the target chiplet of a Manageability packet is determined.
- See Section 8.2.1 for details of how UDMs are encapsulated on the UCIe sideband and mainband.
- Similar to sort testing,
  - For sending/receiving scan test patterns, Vendor-defined UCIe DFx Messages (UDM) are used over UCIe. These messages can target the appropriate Spoke in the design that implements the scan control functionality.
  - For general-purpose testing/debugging using register reads and writes, UMAP messages can be used, as defined in Section 8.1.4.

8.3.4 System Debug with UDA

For system-level debug, various interfaces can be used for Controllability/Observability. In Figure 8-62, an I3C interface running a debug protocol is the connection between the debugger and the DFx hooks on each chiplet. A x16 PCIe interface is used to send debug data to a logic analyzer. PCIe debug VDM packets can be used to carry debug information on this interface. Similarly, a remote debugger could access debug data over USB using an appropriate protocol. Note that per the UCIe Management Network Architecture requirements, a management bridge is required at the USB interface in Chiplet 5, or I3C interface in Chiplet 0, or x16 PCIe interface in Chiplet 2.

Figure 8-62. UCIe-based System Testing/Debug



8.3.5 DMH/DMS Registers

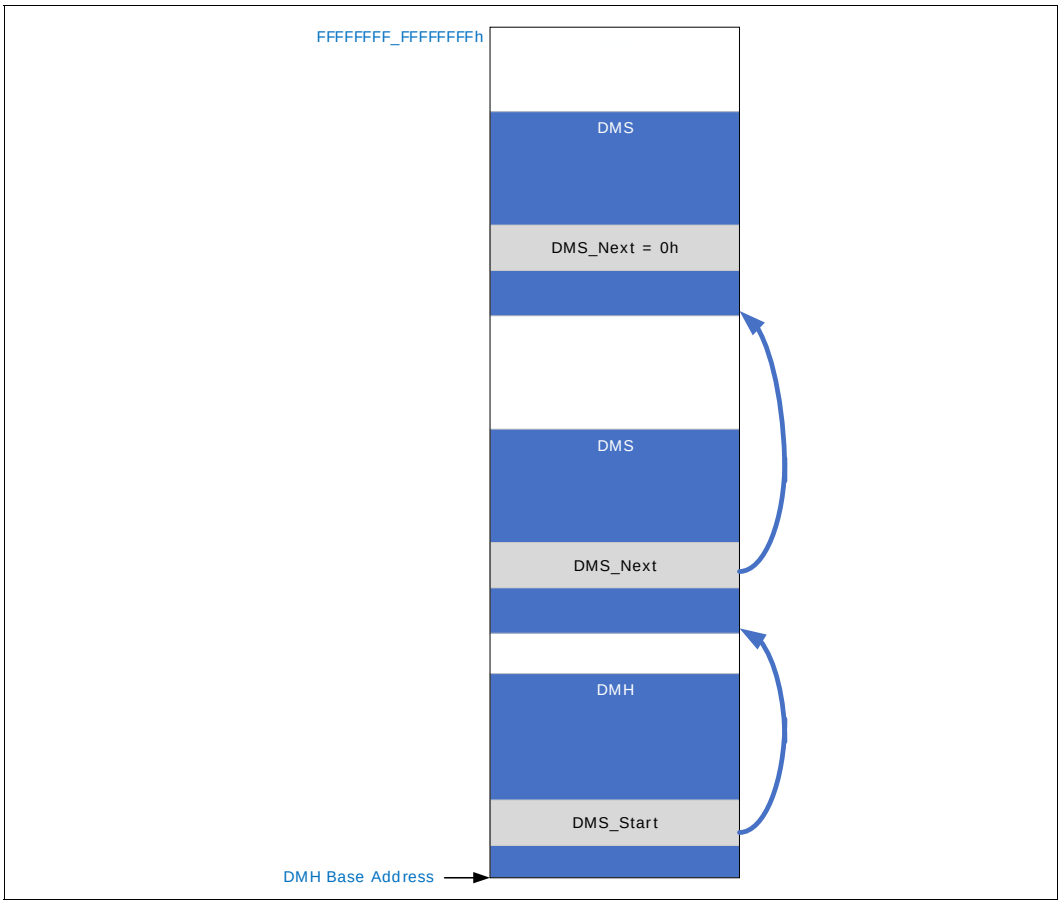
8.3.5.1 DMH/DMS Register Address Space and Access Mechanism

DMH and DMS registers are located within the memory space of the Management Element in which they reside. UMAP is used to access these registers. DMH and DMSs all share the same memory address space of the associated management element, and they each occupy specific address ranges within the address space. DMH and DMSs are discovered using a linked list with pointers in DMH and DMS register space, respectively. In Figure 8-63, DMH has two Spokes connected to it. The pointer in DMH points to the first DMS which then points to the next DMS. The pointer in the second DMS indicates the end of Spokes in the DMH with a value of all 0s in its Next pointer.

All spec-defined registers in DMH and DMS are accessed in DWORD size only.

The DMH base address in Figure 8-63 is from the Capability Directory of Management Element that hosts a DMH (see Section 8.1.3.6.1 for details).

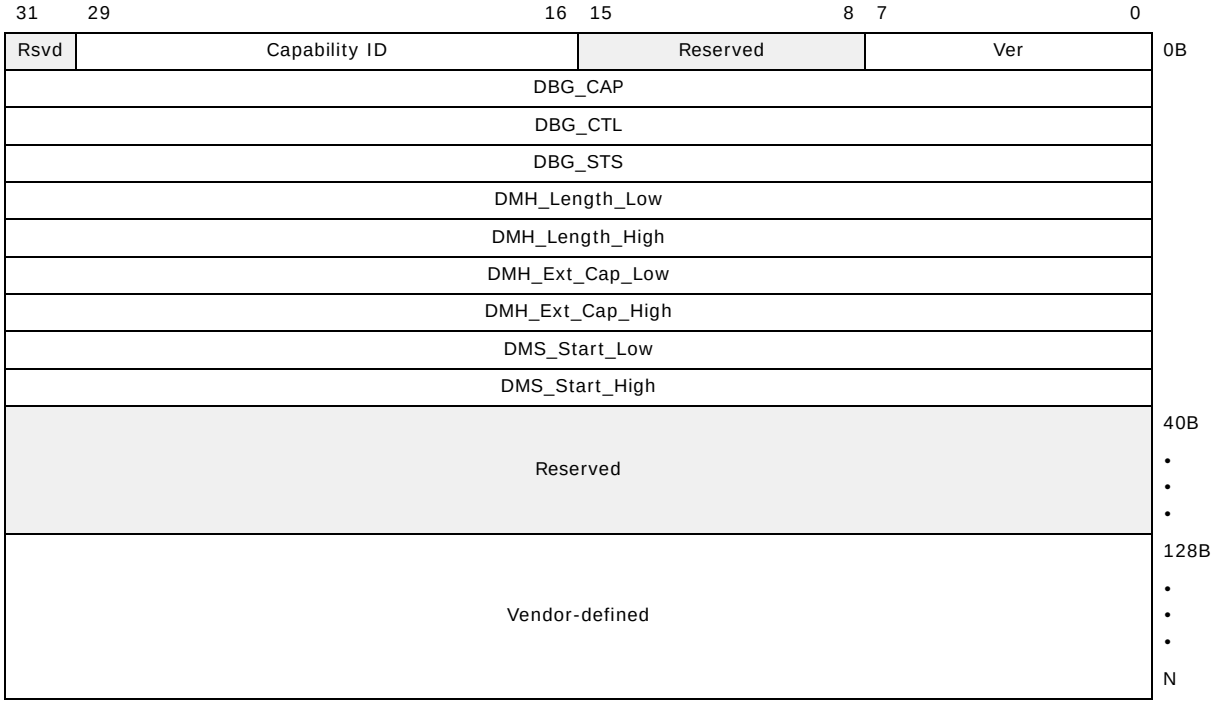
Figure 8-63. DMH/DMS Address Mapping



8.3.5.2 DMH Registers

DMH registers are defined in the DMH Capability shown in Figure 8-64. DBG_STS falls within the “Chiplet Status” Asset Class. All other spec-defined registers fall within the “Chiplet Configuration” asset class.

Figure 8-64. DMH Capability Register Map



8.3.5.2.1 Ver (Offset 00h)

Table 8-42. Version

Bit	Attribute	Description
7:0	RO	**Version** Set to 00h.

8.3.5.2.2 Capability ID (Offset 02h)

Table 8-43. Capability ID, Ver

Bit	Attribute	Description
13:0	RO	**Capability ID** Set to 2h to indicate DMH.

8.3.5.2.3 DBG_CAP — Debug Capabilities (Offset 04h)

Table 8-44. Debug Capability

Bit	Attribute	Description
3:0	RO	**Version** Set to 0h.
31:4	RsvdP	**Reserved**

8.3.5.2.4 DBG_CTL — Debug Control (Offset 08h)

Table 8-45. Debug Control

Bit	Attribute	Description
0	RWL	**Disable Accesses to/from DMS** 1: Disables UMAP and Test/Debug Vendor-defined UDM accesses to/from DMSs connected to the DMH from/to the UCIe Management network. 0: Enables accesses to DMSs connected to the DMH. Default is 1b. This bit is locked for writes if bit 1 in this register is set to 1.
1	RO	**Lock 'Disable Accesses to DMS'** Default value is 0. After SW writes 1 to this bit, this bit cannot be modified further until the next Management Reset.
31:2	RsvdP	**Reserved**

8.3.5.2.5 DBG_STS — Debug Status (Offset Ah)

Table 8-46. Debug Status

Bit	Attribute	Description
0	RO	**DMS Accessed** At least one DMS was accessed from the management network since the last Management Reset. This bit is cleared on each Management Reset.
31:1	RsvdP	**Reserved**

8.3.5.2.6 DMH_Length_Low — DMH Register Space Length Low (Offset 10h)

Table 8-47. DMH_Length_Low

Bit	Attribute	Description
31:12	RO	Lower 20 bits of the length of the DMH register space in multiples of 4K. Bits [11:0] in this register are reserved to ensure 4k multiples of length. A value of 1000h for {DMH_Length_High :: DMH_Length_Low} indicates a length of 4K. A value 2000h indicates a length of 8K, etc.
11:0	RsvdP	**Reserved**

8.3.5.2.7 DMH_Length_High — DMH Register Space Length High (Offset 14h)

Table 8-48. DMH_Length_High

Bit	Attribute	Description
31:0	RO	Upper 32 bits of the length of the DMH register space in multiples of 4K. A value of 1000h for {DMH_Length_High :: DMH_Length_Low} indicates a length of 4K. A value 2000h indicates a length of 8K, etc.