# 2.2.10 TLP前缀规则

以下规则适用于任何包含TLP前缀的TLP:

•对于任何TLP, TLP第0字节Fmt[2:0]字段值为100b表示存在TLP Prefix, Type[4]位表示TLP Prefix的类型。

◦Type[4]位的值为0b，表示存在Local TLP Prefix;

◦Type[4]位的值为1b，表示存在End-End TLP Prefix

•TLP前缀的字节1到字节3的格式由其TLP前缀类型定义。

•包含TLP前缀的TLP必须有一个底层的TLP报头。收到的违反此规则的TLP将作为畸形TLP处理。这是一个与接收端口相关的报告错误(参见6.2节)。

•允许一个TLP包含多个任何类型的TLP前缀

◦当TLP中存在Local和End-End TLP前缀的组合时，需要所有Local TLP前缀位于任何End-End TLP前缀之前。收到的违反此规则的TLP将作为畸形TLP处理。这是一个与接收端口相关的报告错误(参见6.2节)。

•每个TLP前缀的大小为1dw。可以重复使用TLP前缀，以便为附加数据提供空间。

•如果Fmt和Type字段的值表示存在Local TLP Prefix，请按照2.2.10.1章节的Local TLP Prefix处理方法进行处理。

•如果Fmt和Type字段的值表示存在End-End TLP前缀，请按照2.2.10.2节的End-End TLP前缀处理方法进行处理。

## 2.2.10.1 Local TLP Prefix Processing

Local TLP前缀的规则如下:

•使用Type字段的L[3:0]子字段确定本地TLP前缀类型

◦Type[4]必须为0b

◦Local TLP Prefix L[3:0]的定义如表2-36所示

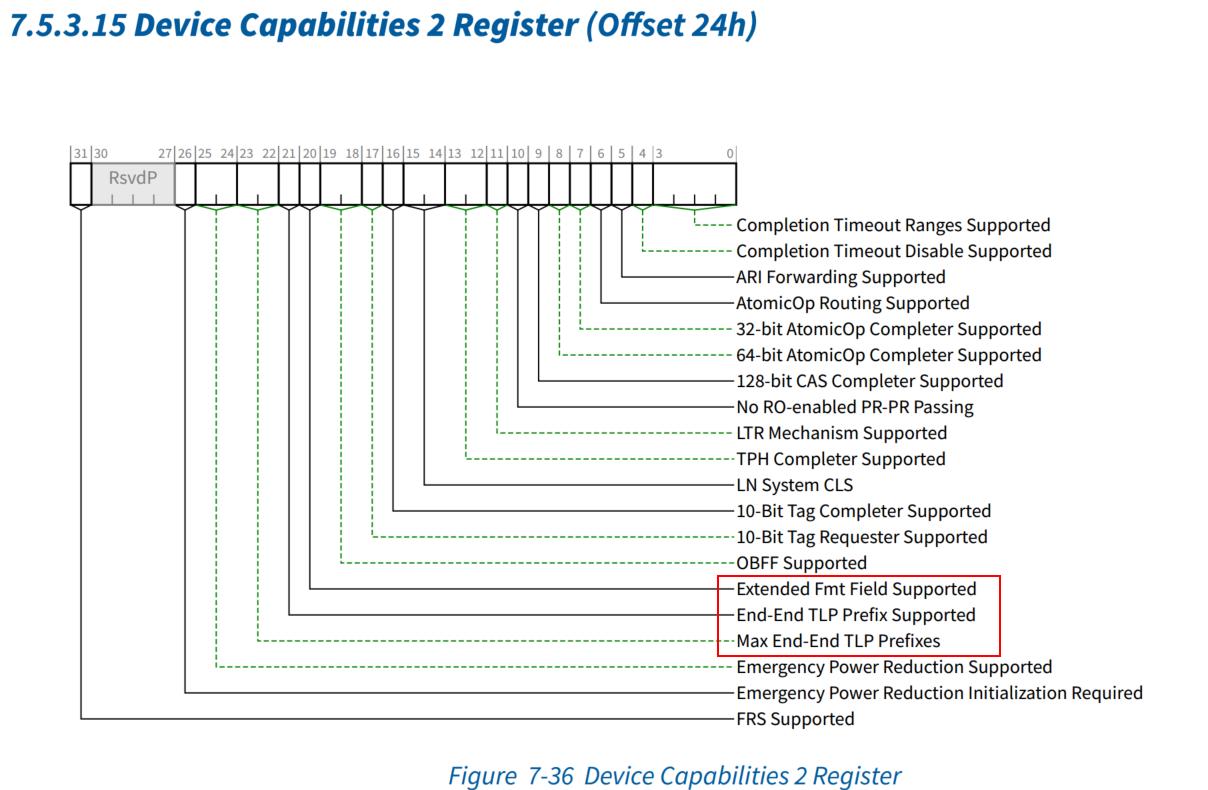
表2-36本地TLP前缀类型

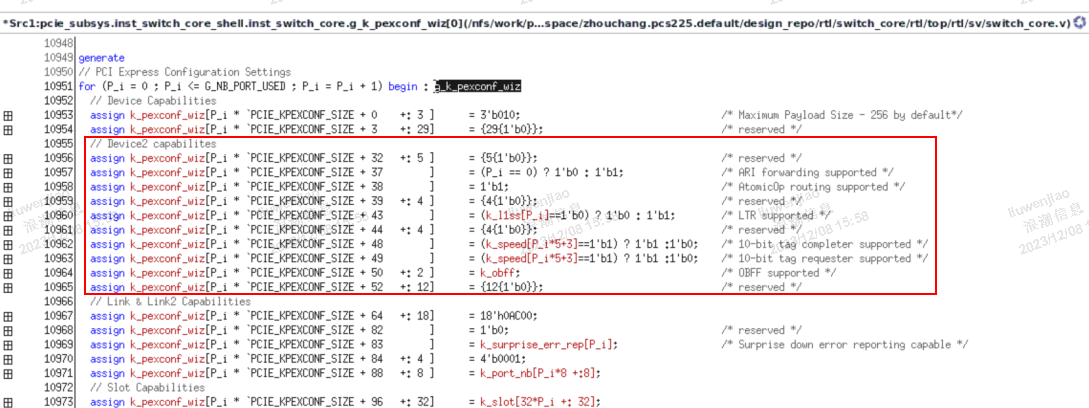
|  |  |  |
| --- | --- | --- |
| 本地TLP前缀类型 | L[3:0](b) | Description |
| MR-IOV | 0000 | MR-IOV TLP前缀-详细信息请参见[MR-IOV]规范。 |
| VendPrefixL0 | 1110 | 厂商定义的本地TLP前缀—详细信息参见2.2.10.1.1节。 |
| VendPrefixL1 | 1111 | 厂商定义的本地TLP前缀—详细信息参见2.2.10.1.1节。 |
|  |  | 所有其他编码是Reserved的。 |

•大小、路由和流量控制规则根据不同的Local TLP Prefix类型而不同。

•接收到接收方不支持的本地TLP前缀类型的TLP错误。如果设置了扩展Fmt字段支持位，则违反此规则的TLP将作为畸形TLP处理，除非在另一个规范中有明确的不同说明。这是一个与接收端口相关的报告错误(参见6.2节)。如果扩展Fmt字段支持位为Clear，则行为与设备有关。

•即使底层TLP受ECRC保护，本地TLP前缀也不受ECRC保护。





### 2.2.10.1.1厂商定义的本地TLP前缀

如表2-36所示，“VendPrefixL0”和“VendPrefixL1”类型保留作为厂商自定义的本地TLP前缀。为了最大限度地提高互操作性和灵活性，以下规则适用于这些前缀:

•组件不能发送包含供应商定义的本地TLP前缀的TLP，除非已显式启用(使用供应商特定的机制)。

•支持任何供应商定义的本地TLP前缀的组件必须支持Fmt字段的3位定义，并具有扩展Fmt字段支持位集(参见章节7.5.3.15)。

•建议组件是可配置的(使用供应商特定的机制)，以便所有供应商定义的前缀都可以使用两种供应商定义的本地TLP前缀编码中的任何一种发送。这样的配置不必是对称的(例如，链路的每一端可以使用不同的编码传输相同的前缀)。

## 2.2.10.2 End-End TLP前缀处理

End-End TLP前缀的规则如下

•End-End TLP前缀类型由Type字段的子字段E[3:0]确定

◦Type[4]必须为1b

◦End-End TLP前缀E[3:0]的值定义如表2-37所示

表2-37端到端TLP前缀类型

|  |  |  |
| --- | --- | --- |
| END-ENDTLP前缀类型 | L[3:0](b) | Description |
| TPH | 0000 | TPH -请参阅第2.2.7.1节和第6.17节了解更多细节 |
| PASID | 0001 | PASID -详细信息请参见第6.20节。 |
| VendPrefixE0 | 1110 | 供应商定义的End-End TLP前缀-详细信息请参见2.2.10.2.1节。 |
| VendPrefixE1 | 1111 | 供应商定义的End-End TLP前缀-详细信息请参见2.2.10.2.1节。 |
|  |  | 所有其他编码是保留的。 |

•在一个TLP中允许的最大End-End TLP前缀数为4:

◦支持TLP前缀的Receiver必须检查此规则。如果接收方认定该TLP违反了此规则，则该TLP为畸形TLP。这是一个与接收端口相关的报告错误(参见6.2节)。

•End-End TLP前缀的存在不会改变该TLP的路由。tlp基于2.2.4节介绍的路由规则进行路由。

•函数通过Device Capabilities 2寄存器中的Max End-End TLP Prefixes字段表示它们支持多少个End-End TLP前缀(参见章节7.5.3.15)。

◦对于根端口，允许Max End-End TLP前缀字段返回一个值，表示支持比根端口硬件实际实现的更少的End-End TLP前缀;但是，错误处理语义仍然必须基于字段中包含的值。收到的TLP包含的End-End TLP前缀数超过根端口支持的数目时，处理方法如下。

建议将请求作为不受支持的请求处理，否则它们必须作为畸形的tlp处理。建议将完成作为意外完成处理，否则必须作为畸形tlp处理。对于入端口收到的TLPs，这是一个与入端口相关的报告错误。对于内部接收到的要从出口端口发送出去的tlp，这是一个与出口端口相关的报告错误。参见6.2节。

◦对于所有其他功能类型，收到的TLP包含更多的End-End TLP前缀，而不是一个功能所支持的，必须作为畸形TLP处理。这是一个与接收端口相关的报告错误(参见6.2节)。

高级错误报告(AER)日志记录(如果支持)按照章节6.2.4.4的规定发生。

•如果设置了End-End TLP前缀支持位，交换机必须支持转发最多4个End-End TLP前缀的TLP。

•不同根端口的End-End TLP前缀支持位集允许报告不同的最大End-End TLP前缀值。

• 如果底层TLP受ECRC保护，则所有End-End TLP前缀都受ECRC保护。

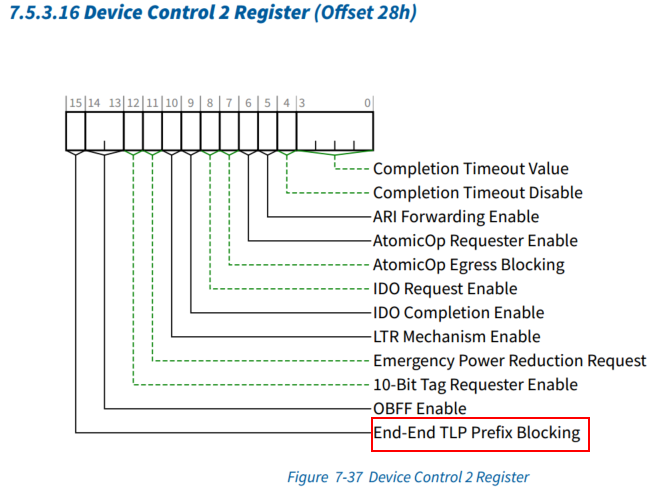
•接收不支持End-End TLP前缀的接收器收到带有End-End TLP前缀的TLP会出错。违反此规则的TLP将作为畸形TLP处理。这是一个与接收端口相关的报告错误(参见6.2节)。

•软件应确保包含End-End TLP前缀的TLP不会发送到不支持它们的组件。扩展Fmt字段支持位为Clear的组件可能会误解包含TLP前缀的TLP

•如果上游端口的某个功能设置了End-End TLP前缀支持位，则该上游端口的所有功能都必须将收到的包含不支持的End-End TLP前缀类型的请求作为不支持的请求处理。这是一个与接收端口相关的报告错误(参见6.2节)。

•如果上游端口的某个函数设置了End-End TLP前缀支持位，则该上游端口的所有函数都必须将收到的包含不支持的End-End TLP前缀类型的Completion处理为Unexpected Completion。这是一个与接收端口相关的报告错误(参见6.2节)。

•对于路由元素，每个出口端口的End-End TLP前缀阻塞位决定包含End-End TLP前缀的TLP是否可以通过该出口端口传输(参见7.5.3.16节)。如果转发被阻断，则整个TLP被丢弃，并报告TLP Prefix blocked Error。如果被阻塞的TLP是一个非发布请求，则出口端口返回一个带有Unsupported Request Completion Status的Completion。TLP前缀阻塞错误是一个与出口端口相关的报告错误(参见6.2节)。



•对于启用多播的路由元素(参见6.14节)。End-End TLP前缀在一个TLP的所有组播副本中被复制。TLP前缀Egress组播报文的阻塞在每个Egress端口独立进行

### 2.2.10.2.1厂商定义的End-End TLP前缀

如表2-37所示，“VendPrefixE0”和“VendPrefixE1”类型保留作为厂商自定义的端到端TLP前缀。为了最大限度地提高互操作性和灵活性，以下规则适用于这些前缀:

•组件不能发送包含供应商定义的End-End TLP前缀的TLP，除非已明确启用(使用供应商特定的机制)。

•建议组件可配置(使用供应商特定的机制)，以使用两种供应商定义的End-End TLP前缀编码中的任何一种。这样做允许在单个PCI Express拓扑中同时使用两个不同的供应商定义的端到端TLP前缀，而不要求每个源都了解其发送的每个TLP的最终目的地。

### 2.2.10.2.2支持带有End-End TLP前缀的根端口

支持根端口之间包含End-End TLP前缀的TLP对等路由是可选的，依赖于具体实现。如果RC支持两个或多个根端口之间的End-End TLP前缀路由能力，则必须通过Device Capabilities 2寄存器中的End-End TLP前缀支持位在每个关联的根端口中指出该能力。

对于所有设置了“End-End TLP前缀支持位”的根端口对，RC不需要支持End-End TLP前缀路由。带有End-End TLP前缀的请求需要在不支持的根端口对之间进行路由，必须作为UR处理。带有End-End TLP前缀的补全(Completion)需要在不支持的根端口对之间进行路由，必须作为意外补全(Unexpected Completion, UC)处理。在这两种情况下，该错误都由“发送”端口报告。

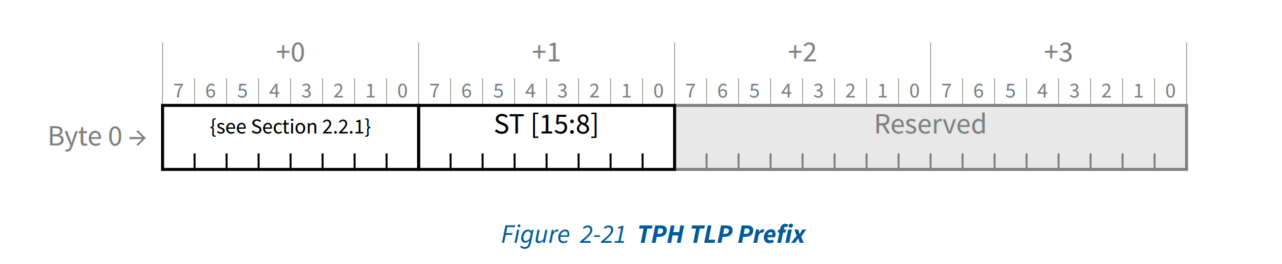
对于支持转发由主机软件或rciep (Root Complex Integrated Endpoints)发起的带有End-End TLP前缀的TLP的根端口，必须设置“End-End TLP前缀支持位”。对于支持将入端口接收到的带有End-End TLP前缀的TLP转发到rciep的根端口，必须设置“End-End TLP前缀支持位”。

设置了End-End TLP前缀支持位集的不同根端口允许报告不同的最大End-End TLP前缀值。

当RC在根端口之间进行点对点路由时，如果将一个TLP拆分为更小的TLP，则必须在每个更小的TLP中复制原TLP的End-End TLP前缀(见1.3.1节)。

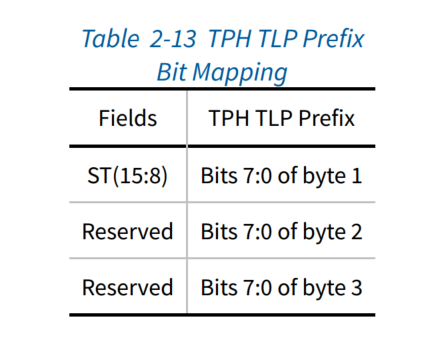
# 2.2.7.1 TPH规则

•TPH指定了两种格式。基线TPH格式(如图2-22和图2-23所示)必须用于所有提供T PH的请求。带有可选TPH TLP前缀的格式扩展了TPH字段(见图2-21)，为转向标签(ST)字段提供了额外的位。



•可选的TPH TLP Prefix用于扩展TPH字段。

◦TPH TLP前缀的存在是通过解码字节0确定的。



•对于以内存空间为目标的请求，TH位的值为1b表示TLP报头中存在TPH和可选的TPH TLP前缀(如果存在)。

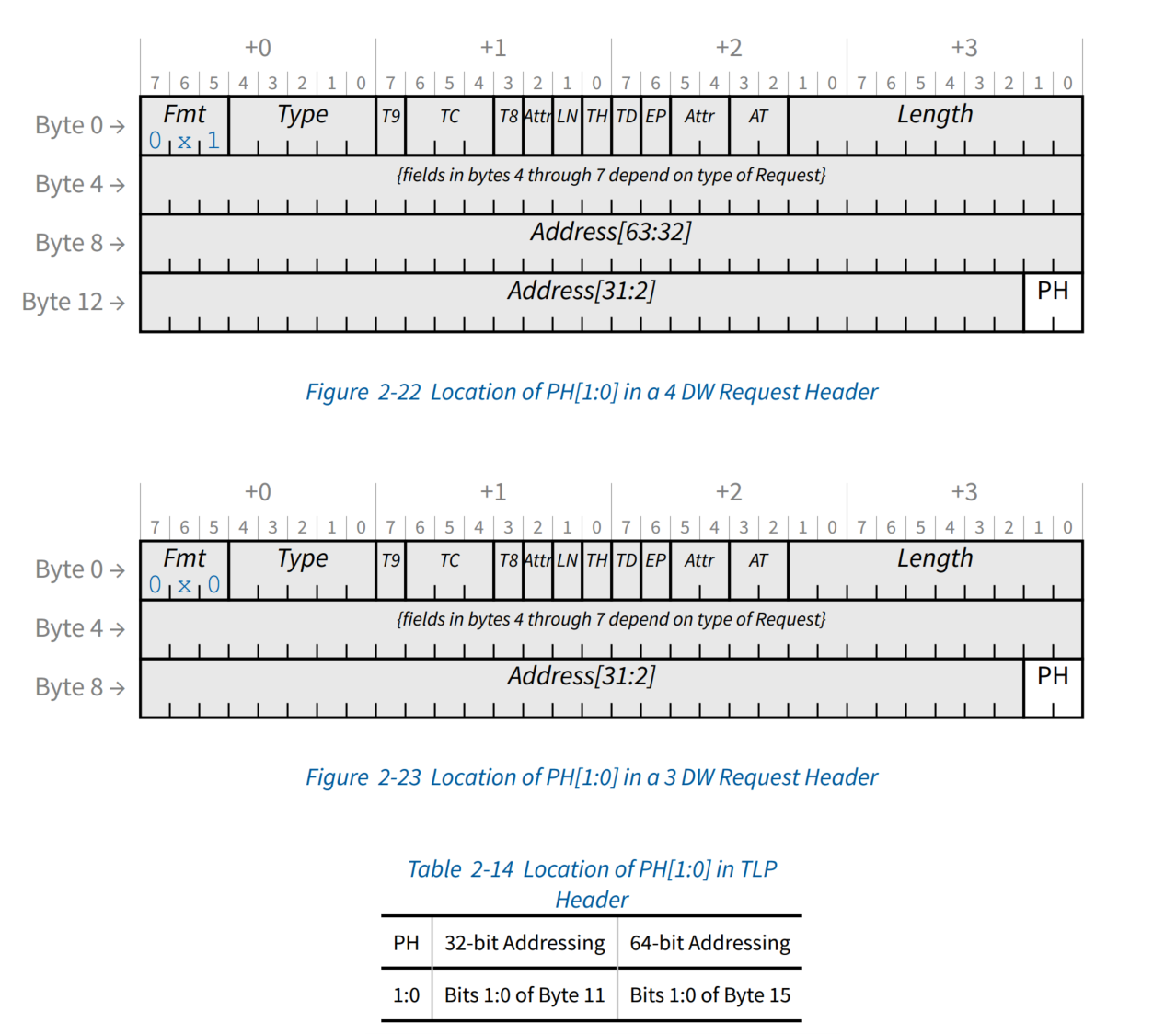
◦对于提供TPH的请求，TH位必须设置。

◦带TPH TLP前缀的请求必须设置TH位。

◦TH位为Clear时，PH字段为Reserved。

◦TH位和PH字段不适用，保留给所有其他请求。

•PH值字段的对应关系如图2-22、图2-23和表2-14所示。

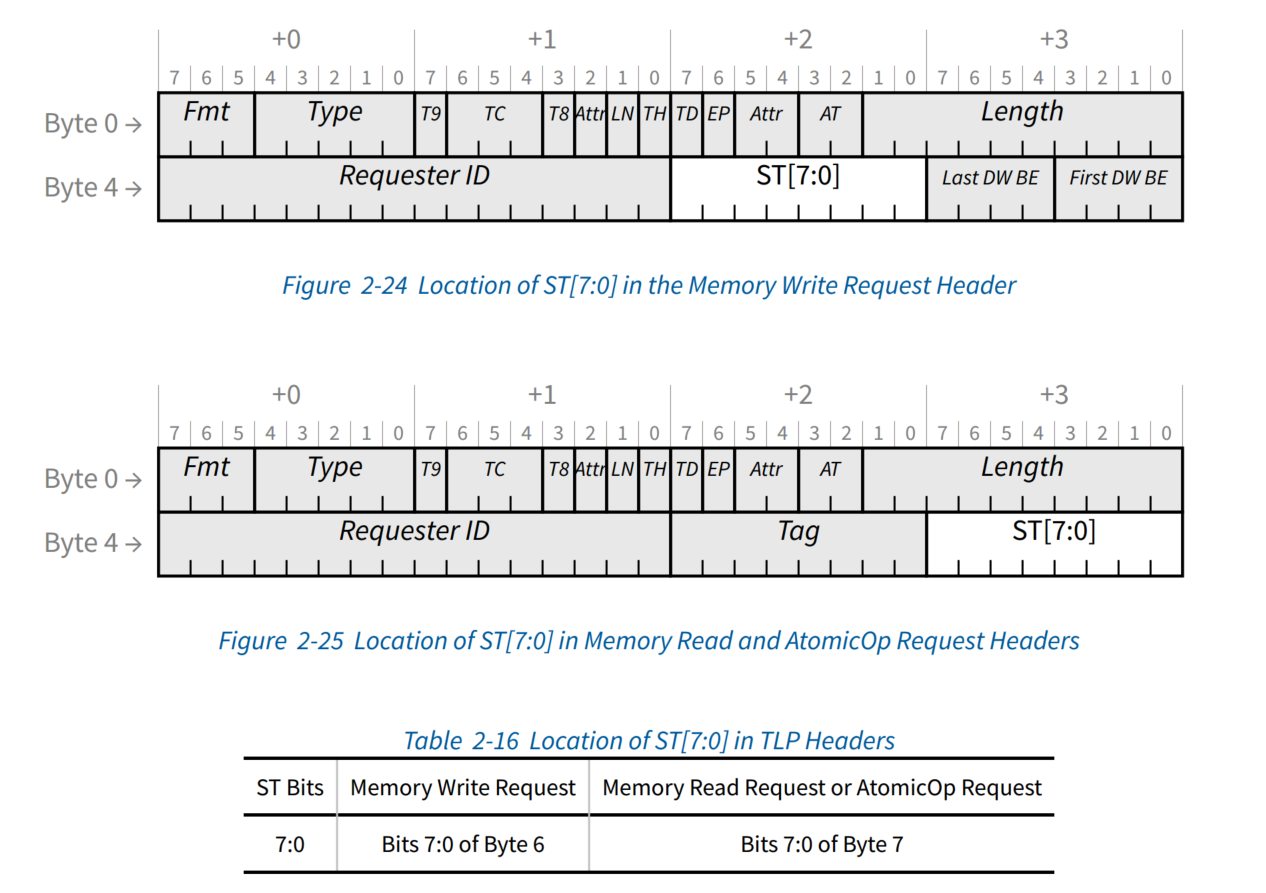


PH[1:0]字段提供有关数据访问模式的信息，其定义如表2-15所示

表2-15处理提示编码

|  |  |  |
| --- | --- | --- |
| PH[1:0](b) | Processing Hint（处理提示） | 描述 |
| 00 | Bi-directional data structure  （双向数据结构） | 表示主机和设备对数据的频繁读写访问 |
| 01 | Requester | 表示设备对数据进行频繁的读/写访问 |
| 10 | Target | 表示主机对数据进行频繁的读/写访问 |
| 11 | Target with Priority | 表示主机频繁的读和/或写访问，并且访问的数据具有较高的时间局部性 |

ST (Steering Tag)字段映射到TLP报头，如图2-24、图2-25和表2-16所示



•ST[7:0]字段携带Steering Tag值

◦所有0的值表示没有转向标签偏好

◦总共提供255个唯一的转向标签值

•不支持TPH补全或路由功能的函数，如果接收到TH位设置的事务，则需要忽略TH位，并以与没有TH位设置的相同事务类型的请求相同的方式处理请求。

# 6.17 TLP处理提示(TPH)

TLP处理提示是一个可选特性，它在请求TLP报头中提供提示，以促进针对内存空间的请求的优化处理。这些处理提示使系统硬件(例如，根复体和/或端点)能够优化平台资源，例如以每个TLP为基础的系统和内存互连。TPH机制定义了提供端点和根复合体之间通信模型信息的处理提示。转向标记是特定于系统的值，用于标识请求程序显式针对的处理资源。系统软件发现并识别TPH功能，以确定支持TPH的每个功能的转向标签分配。

## 6.17.1 Processing Hints

请求者向根复合体或其他目标提供关于主机和/或设备对数据和数据结构的预期使用的提示。提示是由请求者提供的，它知道即将到来的请求模式，而完成者无法自主地(以良好的准确性)推断出这些提示。需要用这些暗示来区分的案例包括:

DWHR:设备写入，然后主机很快读取

HWDR:主机写入数据，设备很快读这段取据

D\*D\*:设备写/读，然后设备很快会再次读/写

Includes DWDW, DWDR, DRDW, DRDR

双向:主机和设备共享且读写权限相等的数据结构。

使用模型对应的Processing Hint编码如表6-12所示。

表6-12处理提示映射

|  |  |  |
| --- | --- | --- |
| PH[1:0](b) | Processing Hint（处理提示） | 使用模型 |
| 00 | Bi-directional data structure  （双向数据结构） | D\*D\* |
| 01 | Requester | DWHR、HWDR |
| 10 | Target | 表示主机对数据进行频繁的读/写访问 |
| 11 | Target with Priority | 与目标相同，但具有暂时重用优先级 |

## 6.17.2转向标签（Steering Tags）

打算将TLP定位到特定处理资源(如主机处理器或系统缓存层次结构)的函数需要目标缓存的拓扑信息(例如，哪个主机缓存)。转向标记是特定于系统的值，它提供有关系统缓存层次结构中的主机或缓存结构的信息。这些值用于将平台内的处理元素与请求的处理相关联。

要使用的软件可编程转向标签值存储在ST表中，该表可以位于TPH请求者扩展能力结构中(见第7.9.13节)，也可以与MSI-X表结合在一起(见第7.7节)，但对于给定的功能，不能同时位于两个位置。当ST表与MSI-X表结合时，每个MSI-X表项的矢量控制寄存器的2个最重要的字节用于包含转向标签值

ST表位置的选择是具体实现的，并且可以由软件发现。实现MSI-X的函数允许在任意位置定位ST表(参见章节7.9.13.2)。一个同时实现MSI和MSI- x的函数被允许将ST表和MSI- x表结合使用，即使MSI- x被禁用(即当MSI被启用时)。每个ST表项是2字节。ST表的大小在TPH请求者扩展能力结构中表示。

对于某些使用模型，不需要或不提供转向标记，在这种情况下，允许函数在ST字段中使用全零的值来表示没有ST首选项。每个请求与ST表项的关联是特定于设备的，超出了本规范的范围。

## 6.17.3 ST操作方式

TPH请求者扩展能力结构中的ST表位置字段指出了函数在哪里(如果有的话)实现了ST表。如果实现了ST表，则软件可以用系统特定的转向标记值对其进行编程

表6-13 ST的操作方式

|  |  |  |
| --- | --- | --- |
| ST Mode Select  [2:0](b) | ST Mode Name | Description |
| 000 | 无ST模式(No ST Mode) | 该函数必须对所有转向标记使用全0的值 |
| 001 | 中断矢量模式(Interrupt Vector Mode) | 每个转向标签由一个MSI/MSI- x中断向量号选择。该函数需要使用ST表项中的转向标签值，该表项可以由有效的MSI/MSI- x中断向量号索引 |
| 010 | 设备特定模式(Device Specific Mode) | 建议函数使用ST表项中的转向标记值，但不强制 |
| 所有其他编码 | Reserved | 保留以备将来使用。 |

在No ST操作模式下，函数必须为每个转向标记使用全零的值，从而允许在没有软件提供的转向标记的情况下使用处理提示。

在中断矢量操作模式中，使用MSI/MSI- x中断矢量号从ST表中选择转向标签。对于启用了MSI的函数，该函数需要在MSI Capability结构中的Multiple Message Enable字段指定的范围内选择标签。对于启用了MSI-X的函数，该函数需要在MSI-X表大小的范围内选择标签。如果ST表大小小于中断向量数的启用范围，则允许该函数对某些事务不使用TPH，将TPH与转向标记一起使用0，或者将TPH与用于从ST表中选择转向标记值的实现定义机制一起使用。如果ST表大小大于中断向量号的使能范围，则函数忽略超出范围的中断向量号对应的ST表项。

在特定于设备的操作模式中，将转向标记分配给请求是特定于设备的。函数使用的转向标签的数量被允许不同于为函数分配的中断向量的数量，而不考虑ST表的位置，并且请求中使用的转向标签值不需要来自架构的ST表

需要一个能够生成TPH请求的函数来支持No ST操作模式。支持其他ST操作模式是可选的。通过编程ST模式选择，一次只能选择一种ST操作模式

**实现注意**

**ST表编程**

**为了确保请求中使用确定性的转向标记值，建议软件在执行ST表更新的过程中暂停函数或禁用TPH请求程序功能**。

**如果不这样做，可能会导致在ST表更新期间使用不确定的ST值。**

## 6.17.4 TPH Capability

TPH功能是可选的规范。每个能够使用TPH生成请求tlp的函数都需要实现TPH请求者扩展能力结构。支持将TPH作为补全器处理tlp的函数需要通过Device Capabilities 2寄存器指示TPH补全器功能。TPH的架构适用于以内存空间为目标的事务，适用于设备到主机、设备到设备和主机到设备之间的事务流。在每种情况下，要支持TPH, Requester、Completer和所有中间路由元素都必须支持相关的TPH功能。

软件通过TPH请求者扩展能力结构发现请求者能力，通过设备能力2寄存器发现完成者能力(见第7.5.3.15节)。软件必须在TPH请求者扩展能力结构中对TPH请求者启用字段进行编程，以使该函数能够发起带有TPH的请求。

软件通过TPH请求者扩展能力结构发现请求者能力，通过设备能力2寄存器发现完成者能力(见第7.5.3.15节)。软件必须在TPH请求者扩展能力结构中对TPH请求者启用字段进行编程，以使该函数能够发起带有TPH的请求。

# 6.20 PASID TLP前缀

PASID TLP前缀是在2.2.1节中定义的End-End TLP前缀。PASID TLP前缀布局如图6-20和表6-14所示。

当存在PASID TLP前缀时，前缀中的PASID值与请求者ID一起标识与请求相关联的进程地址空间ID。每个函数都有一组不同的PASID值。一个函数使用的PASID值与其他函数使用的PASID值无关。

在以下情况下允许使用PASID TLP前缀:

•未转换地址的内存请求(包括AtomicOp请求) (见2.2.4.1节)。

•地址转换请求，ATS无效消息，页面请求消息和PRG响应消息(见章节10.1.3)。

其他TLP不允许使用PASID TLP前缀。

## 6.20.1管理PASID TLP前缀使用情况

必须特别使能PASID TLP前缀的使用。除非使能，否则组件不允许发送PASID TLP前缀。

对于端点函数(包括根复合集成设备)，遵循以下规则:

•一个函数不允许发送和接收带有PASID TLP前缀的TLP，除非设置了PASID Enable(参见章节7.8.8.3):

•一个函数必须有一个机制来动态地将一个PASID的使用与一个特定的函数上下文关联起来。这种机制是特定于设备的。

•一个函数必须有一个机制来请求它优雅地停止使用一个特定的PASID。此机制是特定于设备的，但必须满足以下规则:

◦一个函数可能同时支持有限数量的PASID停止请求。软件应该延迟发出新的停止请求，直到旧的停止请求完成。

◦一个函数中的停止请求不能影响任何其他函数的运行。

◦停止请求不得影响函数内任何其他PASID的操作。

◦停止请求不能影响与PASID无关的事务的操作

◦当停止请求机制指示完成时，Function具有:

▪停止为这个PASID排队新的请求。

▪完成所有与此PASID相关的未发布请求。

▪在所有被PASID使用的tc中，向主机内存发送所有已发送的请求。用于此的机制是特定于设备的(例如:非放松的post - Write到主机内存或处理器读取函数可以刷新TC0;读到主机内存的零长度可以刷新非零tc)。

▪可选地将所有点对点发布请求刷新到它们的目的地。为此使用的机制是特定于设备的。

▪遵守地址翻译服务(第10章)中描述的附加规则，如果地址翻译或页面请求是代表该PASID发出的。

对于根复合体，以下规则适用:

•根复体必须有一个设备特定的机制来指示对PASID TLP前缀的支持。

•支持PASID TLP前缀的根综合体必须具有特定于设备的使能机制。缺省情况下，不允许使用PASID TLP前缀。

•支持PASID TLP前缀的根复合体可以选择具有特定于设备的机制，以比整个根复合体更细的粒度启用它们(例如，distinct启用特定的根端口，请求者ID，总线号，请求者ID或请求者ID/PASID组合)。

## 6.20.2 PASID TLP布局

一个TLP最多只能包含一个PASID TLP前缀。

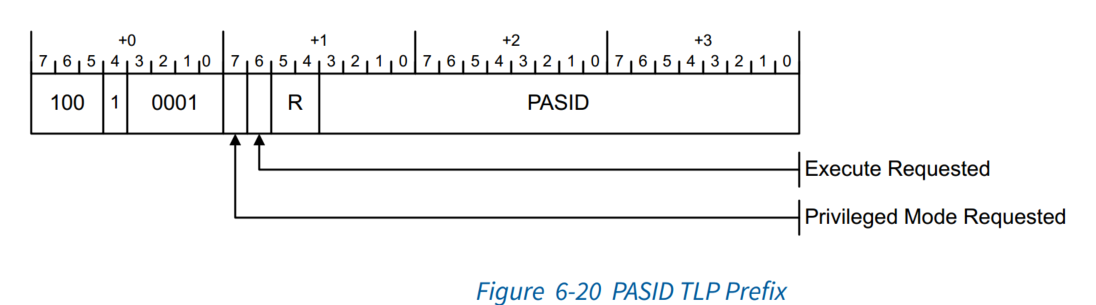


表6-14 PASID TLP前缀

|  |  |
| --- | --- |
| Bits | Description |
| Byte 0 bits 7:5 | 100b – 指示TLP前缀 |
| Byte 0 bit 4 | 1b – 指示End-End TLP Prefix |
| Byte 0 bits 3:0 | 0001b -表示PASID TLP前缀 |
| Byte 1 bit 7 | **特权模式请求**-如果Set表示端点中的特权模式实体正在发出请求，如果Clear表示端点中的非特权模式实体正在发出请求。该位的使用在章节6.20.2.3中有详细说明。 |
| Byte 1 bit 6 | **执行请求**-如果设置表示端点正在请求执行权限。如果清除，表示端点没有请求执行权限。  该位的使用在章节6.20.2.2中有详细说明。 |
| Byte 1 bit 3: Byte3 bit 0 | 进程地址空间ID (PASID) -该字段包含与TLP相关联的PASID值。该字段的用法在章节6.20.2.1中定义。 |

### 6.20.2.1 PASID field

PASID字段标识与请求相关联的用户进程。该字段存在于所有的PASID TLP前缀中。

PASID字段的宽度为20位。端点和根复合体不需要支持整个域范围。对于端点，最大PASID宽度字段表示支持的PASID值范围(章节7.28.2)。对于根复合体，使用特定于实现的机制来提供此信息。

终端不允许发送带有PASID TLP前缀的TLP，除非设置了PASID使能位(章节7.28.3)。

支持PASID TLP前缀的端点在接收到带有PASID TLP前缀且PASID Enable位为Clear的TLP时，必须发出Unsupported Request (UR)信号。

根复体可以选择支持带有PASID TLP前缀的TLP。检测根复合体是否支持PASID TLP前缀的机制取决于具体实现。

对于端点，适用以下规则:

•终端不允许发送PASID值大于或等于2^Max PASID宽度的TLPs。

•当接收到一个PASID值大于或等于2^Max PASID宽度的请求时，端点可选择允许发出错误信号。这是一个与接收端口相关的不支持请求错误(参见6.2节)。

对于根复合体，以下规则适用:

•根综合体不允许发送超过其支持的PASID值的TLP。

当Root Complex接收到一个PASID值大于它所支持的值的请求时，它被允许发出错误信号。这是一个与接收端口相关的不支持请求错误(参见6.2节)。

对于完成者，以下规则适用:

•对于未翻译的内存请求，PASID值和未翻译地址都用于确定用于满足请求的翻译地址。

对于地址转换相关的tlp，该字段的使用在地址转换服务(第10章)中定义。

**实现说明:PASID宽度同质性**

**每个函数的PASID值是唯一的，因此最初的意图是该函数支持的PASID值的宽度可以基于该函数的需要。然而，当前的系统软件通常不遵循该模型，而是在访问特定地址空间的所有函数中使用相同的PASID值。为了实现这一点，系统软件通常会为根复合体和持久翻译代理确保一个通用的系统PASID宽度。此类系统软件通常会在任何热插拔的端点功能或翻译代理上禁用ATS，报告PASID宽度支持小于普通系统的PASID宽度。**

**根复合体、端点和转换代理通常独立于系统软件实现，因此强烈建议硬件实现20位的最大宽度，以确保与系统软件的互操作性。**

**端点可以以特定于实现的方式，将20位系统PASID映射到携带较小宽度的内部表示。如果这样做，端点在不影响系统软件的情况下这样做是至关重要的，因为系统软件没有机制来区分这种实现与那些本地实现完整20位宽度的实现。**

### 6.20.2.2 Execute Requested（请求执行）

如果设置了执行请求位，则端点正在请求端点在与此请求关联的内存范围内执行指令的权限。执行权限的含义超出了本规范的范围。

除非设置了执行权限支持位(章节7.8.8.2)和执行权限使能位(章节7.8.8.3)，否则终端不允许发送带有执行请求位设置的tlp。

对于根复合体，以下规则适用:

•支持由根复合体请求的执行是可选的。用于确定根复合体是否支持执行请求的机制是特定于实现的。

•支持执行请求位的根复合体应该有一个特定的实现机制来允许它使用该位。

•支持执行请求位的根复合体可能具有特定的实现机制，以便在更细的粒度上使用该位(例如，用于特定的根端口，特定的总线号，特定的请求者ID或特定的请求者ID/PASID组合)，其默认值是特定于实现的。

对于完成者，以下规则适用:

•完成者会识别有效位。对于给定的请求，如果支持执行请求位，并且该请求启用了它的使用，则该位的有效值是请求中的值;否则该位的有效值为0b

•对于未转换的memory读请求，完成者使用该位的有效值作为保护检查的一部分。如果此保护检查失败，则完成程序将请求视为未映射内存。

•对于未转换的memory请求，除了未转换的memory读请求外，该位为预留位。

对于地址转换相关的tlp，这个位的使用在地址转换服务(第10章)中定义。

### 6.20.2.3请求特权模式

如果设置了特权模式请求，则端点将发出一个针对与特权模式关联的内存的请求。如果特权模式请求为Clear，则端点正在发出一个针对与非特权模式关联的内存的请求

特权模式和非特权模式的含义以及端点在特权模式或非特权模式下操作的含义取决于系统的保护模型，不在本规范的范围之内。

终端不允许发送设置了特权模式请求位的TLP，除非设置了特权模式支持位(章节7.8.8.2)和特权模式启用位(章节7.8.8.3)

对于根复合体，以下规则适用:

•Root Complex是否支持特权模式请求位是可选的。用于确定根复合体是否支持特权模式请求位的机制是特定于实现的。

•支持特权模式请求位的根复合体应该有一个特定的实现机制来允许它使用该位。

•支持特权模式请求位的根复合体可能具有特定的实现机制，以便在更细的粒度上使用该位(例如，用于特定的根端口，用于特定的总线号，用于特定的请求者ID，或用于特定的请求者ID/PASID组合)

对于完成者，以下规则适用:

•完成者会识别有效位。对于一个给定的请求，如果特权模式请求位被支持并且它的使用为请求启用，则该位的有效值是请求中的值;否则该位的有效值为0b。

•对于未转换的memory请求，补全器使用位的有效值作为其保护检查的一部分。如果此保护检查失败，则完成程序将请求视为未映射内存

•对于地址转换相关的tlp，此位的使用在地址转换服务(第10章)中定义。