**文件控制记录**

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 版本 | 编制人/修订人 | 修订说明 |
| 2024.1.31 | V0.3 | 宁国勋 | 新建 |
| 2024.7.17 | V0.5 | 牛少平 | 1. 对照PCIE5.0\_1.0协议，翻译、修订了全篇内容； 2. 增加了对PCIe Phantom Function等的个人理解内容。 |
|  |  |  |  |
|  |  |  |  |

### 2.2.6 Transaction Descriptor（事务描述符）

#### 2.2.6.1 概述

事务描述符是请求方和完成方之间传递事务信息的一种机制。事务描述符由三部分组成：

* + Transaction ID（事务ID）-用于标识未完成的事务；
  + Attributes（属性）-表明事务的特性；
  + Traffic Class传输等级-表明事务所需的服务类型。

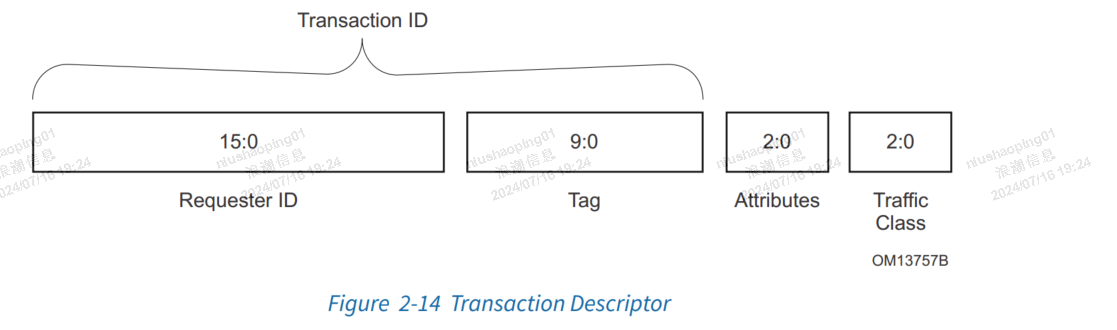
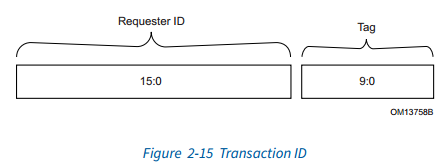


图2-14中展示了事务描述符的各个域段。需要注意的是，图中为了容易理解将它们放在了一起，实际上，在事务包头中它们并不全是连续的。

#### 2.2.6.2 事务描述符——Transaction ID

Transaction ID 包括两部分：Requester ID和Tag，如图2-15所示。



在PCIe-4.0协议中引入的10bit Tag功能将总Tag字段大小从8位增加到10位。两个额外的Tag位，Tag[8] (T8)和Tag[9] (T9)，与TLP包头中的其他Tag[7:0]位并不相邻。这两个新增加的位在本协议的先前版本中是保留位。

* Tag[9:0]是由每个请求者生成的一个10位字段，对于需要Completion的请求者，其所有未完成的请求的Tag[9:0]都必须是唯一的。不支持10bit Tag Requester Capability的请求者必须将Tag[9:8]设置为00b。
  + - 支持16.0 GT/s或更高数据速率的Function(包括Switch中的Function)必须支持10bit Tag Completer capability。如果一个Function支持10bit Tag Completer capability，它可以选择支持10bit Tag Requester capability。参见7.5.3.15节和本节后面的“实现10bit Tag能力的注意事项”的实现说明。
    - 对于含有声明支持10bit Tag Completer capability的elements的RCs，其内部所有作为PCIe 请求者目标的寄存器和存储器，RCs都必须能正确处理10bit Tag的请求。例如，DAM请求或RCiEP中的MMIO区域所对应的主机内存。
* 每个声明支持的RP必须处理由其入口端口接收的此类请求。
* 每个声明支持的RCiEP都必须处理来自支持的内部路径的此类请求，包括通过RP的请求。
* 如果RC包含支持10bit Tag Requester capability的RCiEP，那么RC必须正确处理来自这些RCiEP的、所有寄存器和存储器区域作为这些RCiEP的目标所支持的10bit Tag Requests；例如，DMA请求或RCiEP中的MMIO区域所对应的主机内存。
* Receivers/Completers必须正确处理8bit的Tag值，而不管它们的Extended Tag Field Enable位的设置(参见章节7.5.3.4)。有关bridge处理Extended Tag的详细信息，请参阅PCI Express到PCI/PCI- X桥规范。
* 支持10bit Tag Completer capability的Receivers/Completers必须正确处理10bit Tag值，无论其10bit Tag Requester Enable位如何设置。参见7.5.3.16节。
* 10bit Tag capability不是为PCI Express到PCI/PCI- X桥而设计的，它们（桥）不能声明10bit Tag Requester capability或10bit Tag Completer capability。
* 如果10bit Tag Requester Enable位清0，Extended Tag Field Enable位清0，则每个功能的最大未处理请求数应限制为32，并且仅使用Tag字段的低5位，其余的高5位要求为0 0000b。
* 如果10bit Tag Requester Enable位清0， Extended Tag Field Enable位置1，则最大值增加到256，并且只使用Tag字段的低8位，其余的高2位需要保持为00b。
* 如果10bit Tag Requester Enable位置1，则对一个Completer的最大未完成请求数量增加到768。Requester被允许在向它认为合适的Completer发送10bit Tag Requests时使用Tag字段的所有10位，尽管Requester仍然被允许向其他Completer发送较小的Tag请求。以下内容适用于设置了“10bit Tag Requester Enable”位的10bit Tag capable Requesters。
* 如果一个Endpoint支持向其他Endpoint(而不是host memory)发送请求，除非有特定实现的机制可以确定 目标Endpoint 支持 10-Bit Tag Completer capability，该Endpoint 不得向另一个给定 Endpoint 发送 10-Bit Tag Requests。对于某些实现，根本不向其他端点发送10bit Tag Requests可能是可以接受的。更复杂的机制超出了本协议的范围。
* 如果一个PIO请求者具有10bit Tag Requester capability,，那么请求者如何确定何时使用10位Tag及更小的Tag不在本规范的范围之内。
* 对于10位Tag，有效的Tag[9:8]值为01b、10b或11b。Tag[9:8]等于00b的10位Tag值是无效的，请求者不得生成。这使请求者能够确定其接收到的应具有10位Tag的completion是否包含无效标记，这通常是由于完成者不支持10bit Tag Completer capability造成的。
* 如果请求者向不支持10bit Tag Completer capability的完成者发送10bit Tag Request，则返回Completion的Tag中Tag[9:8]等于00b。由于请求者被禁止为10bit Tag生成这些Tag值（指Tag[9:8]等于00b），因此此类Completion将被处理为意外完成包，默认情况下将报告非致命错误（Non-Fatal Errors）。请求者必须遵循标准PCI Express错误处理要求。
* 当请求者将带有无效10bit Tag的Completion处理为意外完成包（Unexpected Completion）时，原始请求可能会引发完成超时（Completion timeout）。如果请求者为了避免数据损坏而以某种特定于设备的方式处理完成超时条件，则允许请求者根据需要，通过标准PCI Express错误处理机制抑制（suppress）完成超时的处理。
* 如果请求者支持向某些完成者发送10bit Tag Requests，同时支持向其他完成者发送较小的Tag Requests，则请求者必须遵守较小Tag Requests的Extended Tag Field Enable位设置（个人理解，这个设置指completer中的寄存器）。也就是说，如果该位清零，则只有Tag字段的低5比特可以是非零的；如果该位置1，则只有Tag字段的较低8位可以是非零的。
* 如果请求者支持向某些完成者发送10bit Tag Requests，并同时支持向其他完成者发送较小的Tag Requests，则请求者必须确保，如果任何10bit Tag Requests由不支持10bit Tag Completer capability的完成者完成，则没有未完成的10bit Tag会与未完成的较小Tag混淆。参见本节后面的“同时使用10bit Tag和较小的Tag”实现说明。（个人理解，本节中较小Tag Request是特指小于8位的Tag）
* Extended Tag Field Enable位的默认值与具体实现有关。10bit Tag Requester Enable位的默认值为0b。
* 如果发送的多个未完成的请求具有非唯一的Tag值，那么Receiver/Completer的行为在规范中是未定义的。
* 若使用幻象功能号（Phantom Function Numbers）来扩展最大请求数，请求方必须保证在其生成的所有未完成且需要返回完成包的请求中，每一个请求包的幻象功能号和Tag域的组合具有唯一性。

————————————————

个人理解（引用）：

2022.04.29 - PCIe Phantom Function 是什么？

Phantom Function，“幻影” Function，看似存在实则不存在的 Function，用于增加单 Function 设备暂存的 NP 事务数量，从而增大 Outstanding 能力。

一般情况下，PCIe 设备中每个 Function 都有一块空间暂存该 Function 发出去的 NP 请求，共 8 块，每块暂存的 NP 请求数量上限取决于 Tag 字段宽度（8-bit TAG -> 256）。若设备只有一个 Function，那多多少少有点浪费。Phantom Function 正是看中了这一点，当设备只有一个 Function 且支持 Phantom Function 时，通过使能设备控制寄存器中 Phantom Function Enable 位，将 BDF 中的 Function Number 字段当 Tag 用，这样每一个 Function 的 NP 暂存空间都能被利用到，从而将设备的 Outstanding 能力变为之前的 8 倍。

注意：Phantom Function 不是 Virtual Function，两者不是一个概念。

————————————————

* 对于Posted Requests，Tag[9:8]是保留的。
* 对于TH位置1的Posted Requests，Tag[7:0]字段被重新用于ST[7:0]字段（详细信息请参考章节2.2.7.1）。对于TH位清零的post Requests, Tag[7:0]字段是未定义的，可以是任何值。(对于某些Vendor\_Defined Messages，此规则的例外情况请参见表F-1。)
* 对于TH位清零的Posted请求，Tag[7:0]字段中的值不能影响接收方对请求的处理。
* 对于TH位置1的Posted请求，ST[7:0]字段中的值可能会影响请求的完成处理(详见2.2.7.1)。
* Requester ID和Tag为系统中的每个事务组成了一个全局性的标识符，即事务ID。
* 请求包和完成包中都包含事务ID。
* Requester ID是一个16位的值，并且系统中的每个PCI Express功能都有一个唯一的请求ID。
* 功能必须捕获由功能完成的所有类型0配置写入请求提供的总线号和设备号，并在设备/功能发起的所有请求的Requester ID的总线号和设备号字段中提供这些编号。建议仅对成功完成的请求捕获相关数字。

例外：Root Complex内设备的总线号和设备号的设置，以及Switch下游端口的设备号设置，可以由具体实现确定。

注意，设备的总线号和设备号可能在工作时发生变化，所以在收到每个配置写请求时，都有必要重新获取总线号和设备号。

建议对未实现功能的配置写请求不会影响到捕获总线号和设备号。

* 当Switch代表自己生成请求（例如，用于错误报告）时，Switch必须使用与桥的主端相关联的请求者ID，该主端在逻辑上与产生请求的端口相关联。（参见第7.1节）
* 在对一个功能进行初始化配置写之前，该功能不允许发送Non-Posted请求（需要有效的请求者ID才能正确路由生成的完成）。
* 例外：Root Complex中的功能可以在软件初始化配置之前，发起访问到系统引导设备（system boot devices）。

请注意，此规则和例外，与现有的用于系统初始化和配置的PCI模型一致。

* 与一个设备相关联的每个功能，都必须设计为这样的，即只响应针对该设备的配置请求的唯一功能号。注：每个非ARI设备最多可包含八个功能，每个ARI设备最多可包含256个功能。
* Switch在传递请求时不允许修改事务ID。
* 在某些情况下，PCI Express-PCI桥需要为来自PCI/PCI-X总线的请求生成事务ID。
* **实现时注意——使用Phantom Functions增加未完成请求数**

为了增加需要Completion的未完成请求的最大可能数量，而不仅仅是使用标签位，如果幻象功能号使能位（Phantom Function Number Enable bit，参见7.5.3.4节）被置位，一个设备可以使用未被分配给功能的功能号来逻辑扩展Tag标识符。对于单功能设备，这种方法可以增加到原来8倍的最大请求数。

未声明的功能号称为幻象功能号（Phantom Function Numbers，PFN）。

幻象功能有很多架构上的限制，包括如下特征都不支持：ARI设备、虚拟功能（VFs）、在启用VFs时的物理功能（PFs）等。另外，地址翻译服务（ATS）和基于ID的排序（IDO）也不兼容幻象功能。因此，对于许多实现来说，使用10bit Tag是增加未完成的Non-Posted Requests数量的最好方法。

* **实现时注意——实现10位Tag功能的注意事项**

10位Tag的使用使请求者能够将其未完成的Non-Posted Requests（NPR）的数量从256增加到768个，对于非常高速率的NPR，这可以避免标签可用性成为瓶颈。以下公式给出了有效载荷带宽、未完成的NPR数量和其他因素之间的基本关系：

BW=S\*N/RTT，

其中：BW = 有效载荷带宽；S = 事务有效负载大小；N = 未完成的NPR数量；RTT = 事务往返时间。

通常，只有使用相对较小事务的高速链路上的高速请求方，能从将未完成NPR数量增加到256个以上中受益，尽管这也有助于在事务往返时间较长的配置中保持或提升性能。

在具有10-Bit Tag Requester capability的请求者需要面对多个完成者的配置中，需要确保请求者只向具有10-Bit Tag Completer capability的完成者发送10bit Tag请求。如果所有完成者都具有此功能，则可以大大简化此操作。

为了在整个行业启用10-bit Tag，强烈建议所有功能支持10bit Tag Completer Capability。通过这些新的实现，不需要同时操作更多数量的NPR的完成者本身通常可以在内部跟踪10位标签，并以适度的增量投资在完成时返回它们。（个人理解，就是在不需要同时操作更多数量的NPR的完成者中，可以根据自身需要处理的NPR数量来简化或减少并发处理Tag的机制。）

实际同时处理更高数量NPR的完成方可能需要大量额外的硬件资源，但除非完成者实际中能同时处理更多数量的NPR，否则通常无法实现10Bit Tag的全部性能优势。

对于RC支持10bit Tag Completer Capability的平台，强烈建议配置PCIe系统的平台固件或操作软件在具有10bit Tag Requester Capability的EP中自动设置10-bit Tag Requester Enable bit。这使能了一类重要的10Bit Tag的适配：仅向主机内存发送内存读取请求。

对于RCiEP以外的EP，可以通过检查其相关RP中10-Bit Tag Completer Supported 位来确定RC是否支持每个EP的10-Bit Tag Completer capability。RCiEP没有相关RP，因此，除非RC支持其10位标签完成器能力，否则不允许设置其10位标记请求器支持的位。因此，软件不需要对RCiEP执行单独的检查。

缺乏10位标签完成器能力的Switch仍然能够正确地转发携带10位标签的NPR和完成包，因为两个新的标签位在以前保留的TLP包头位中，并且交换机需要在不修改的情况下转发保留的TLP-包头位。然而，如果这样的Switch检测到带有10比特标签的NPR的错误，并且该交换机通过充当NPR的完成器来处理该错误，则其产生的完成包将具有无效的10比特标签。因此，强烈建议使用10位标签的任何组件之间的交换机支持10位标签完成器功能。请注意，支持16.0 GT/s或更高数据速率的交换机必须支持10位标记完成器功能。

对于具有10位标签请求者能力的请求者以完成者为目标的配置，其中一些完成者具有10位标记完成者能力，而另一些不具有10位标志完成者能力，请求者如何确定哪些NPR包括10位标签不在本规范范围内。

* **实现时注意——同时使用10位标记和较小的标记**

如本节前面所述，如果请求方支持向某些完成方发送10位标记请求，并支持同时向其他完成方发送较小的标记请求，则如果缺少10位标记完成方能力（10bit Tag Completer Capability）的完成方完成了任何10位标记申请，则请求方必须确保未完成的10位标记不能别名为未完成的较小标记。（个人理解，即请求方需保证未完成的10位标记在这样的完成方中不能混淆。）

一种实现方法是让请求方将其8位标签空间划分为两个区域：一个区域仅用于较小的标签（8位或5位标签），另一个区域只用于10位标签的较低8位。注意，这将强制在可用于10位标签的标签空间和较小标签之间进行权衡。

例如，如果请求方将其8位标签空间划分为仅将最低的4位用于较小的标签，则最多支持16个未完成的较小标签，并将10位标签空间减少3\*16个值，支持768-48=720个未完成10位标签。许多其他分区选项也是可能的，所有这些选项都可以减少未处理请求的总数。通常，为较小的标签保留N个值会将10位标签空间减少3\*N个值，并且较小的标签加上10位标签的总数最终为768-2\*N。