



PCI-SIG ENGINEERING CHANGE NOTICE

TITLE:	Request Dependencies
DATE:	November 13, 2003
AFFECTED DOCUMENT:	PCI Express Base Specification Revision 1.0a
SPONSOR:	Mark Hummel; Advanced Micro Devices

Part I

1. Summary of the Functional Changes

Clarifies rules for forwarding packets between differing virtual channels by root complexes, switches and endpoints

2. Benefits as a Result of the Changes

The additional rules provide a framework for system designers to ensure deadlock free system operation.

3. Assessment of the Impact

All changes are backward compatible with current specification.

4. Analysis of the Hardware Implications

None.

5. Analysis of the Software Implications

None.

Part II

Detailed Description of the change

Edit & add text as shown:

2.4.1. Transaction Ordering Rules

Table 2-23 defines the ordering requirements for PCI Express Transactions. The rules defined in this table apply uniformly to all types of Transactions on PCI Express including Memory, I/O, Configuration, and Messages. The ordering rules defined in this table apply within a single Traffic Class (TC; see Section 2.5. Virtual Channel (VC) Mechanism). There is no ordering requirement among transactions with different TC labels. Note that this also implies that there is no ordering required between traffic that flows through different Virtual Channels since transactions with the same TC label are not allowed to be mapped to multiple VCs on any PCI Express Link.

...

- Combining of Memory Read Requests, and/or Completions for different Requests is prohibited.
- The No Snoop bit does not affect the required ordering behavior.
- For Endpoints and Bridges acceptance of a posted or non-posted request must not depend upon the transmission of a posted or non-posted request in the same Traffic Class.
- Acceptance of a posted request must not depend upon the transmission of a completion in the same Traffic Class.
- Completions issued for non-posted requests must be returned in the same Traffic Class as the corresponding non-posted request.
- Acceptance of a completion must not depend upon the transmission of a posted or non-posted request or a completion in the same Traffic Class.
- Root Complexes that support peer-to-peer operation and Switches must enforce these transaction ordering rules for all forwarded traffic.

To ensure deadlock free operation devices should not forward traffic from one virtual channel to another. The specification of constraints used to avoid deadlock in systems where devices forward or translate transactions between virtual channels is outside the scope of this document (see Appendix D for a discussion of relevant issues).

Add appendix:

D. Request Dependencies

This document does not specify the allowed resource dependencies between TLPs using different Traffic Classes. Such dependencies can create potential deadlock if devices make different assumptions about what is allowed and what is not.

Resource dependencies are created when received packets are forwarded and retransmitted on either the same or a different link. Due to the fact that the forwarding/translating device has finite buffer resources, this behavior creates a dependency where the ability to receive a packet is dependent upon the ability to transmit a packet (in potentially a different VC or flow control type).

The following notation is used to create a framework to enumerate the possibilities:

$X(m) \rightarrow Y(n)$

This means packet $X(TC=m)$ gets translated/forwarded as packet $Y(FC=n)$ where n and m are between 0-7 and X and Y are either P (Posted Request), N (Non-Posted Request), or C (Completion).

The list of possible dependencies is:

$P(m) \rightarrow P(n)$

$P(m) \rightarrow N(n)$

$P(m) \rightarrow C(n)$

$N(m) \rightarrow P(n)$

$N(m) \rightarrow N(n)$

$N(m) \rightarrow C(n)$

$C(m) \rightarrow P(n)$

$C(m) \rightarrow N(n)$

$C(m) \rightarrow C(n)$

For a given system, each of these dependencies needs to be classified as legal or illegal for each of the following cases:

- Root port forwarding to own link.
- Root port forwarding to different Root Port's link.
- Endpoint or Bridge forwarding to own link.

A Switch will not modify the TLPs that flow through it, but must ensure complete independence of resources assigned to separate VCs. System software must comprehend the system dependency rules when configuring TC/VC mapping throughout the system.

One possible legal mapping might be:

	<u>RC(same port)</u>	<u>RC(diff port)</u>	<u>End Point</u>
<u>P(m)->P(n)</u>	<u>m<=n</u>	<u>m<=n</u>	<u>m<n</u>
<u>P(m)->N(n)</u>	<u>m<n</u>	<u>m<n</u>	<u>m<n</u>
<u>P(m)->C(n)</u>	<u>illegal</u>	<u>illegal</u>	<u>illegal</u>
<u>N(m)->P(n)</u>	<u>m<n</u>	<u>m<n</u>	<u>m<n</u>
<u>N(m)->N(n)</u>	<u>m<=n</u>	<u>m<=n</u>	<u>m<n</u>
<u>N(m)->C(n)</u>	<u>m=n</u>	<u>m=n</u>	<u>m=n</u>
<u>C(m)->P(n)</u>	<u>illegal</u>	<u>illegal</u>	<u>illegal</u>
<u>C(m)->N(n)</u>	<u>illegal</u>	<u>illegal</u>	<u>illegal</u>
<u>C(m)->C(n)</u>	<u>m>=n</u>	<u>m>=n</u>	<u>m>n</u>

Note that this discussion only deals with avoiding the deadlock caused by the creation of resource dependencies. It does not deal with the additional livelock issues (or lack of forward progress) caused by the system's virtual channel arbitration policies.

Some of these potential dependencies are illegal or unreachable:

- P(m)->P(n), N(m)->N(n)
 - m=n – This case is illegal and will lead to deadlock, except if either:
 - 1) A request is forwarded from one port to another of a Switch or Root Complex.
 - 2) A request is reflected on the receiving port by a Root Complex.
 - m!=n – See discussion below.
- P(m)->N(n)
 - m=n – This case is illegal and will lead to deadlock
 - m!=n – See discussion below.
- N(m)->P(n) – See discussion below.
- P(m)->C(n) – This case is illegal and will lead to deadlock.

- N(m)->C(n)
 - m=n – This case occurs during the normal servicing of a non-posted request by either a root complex or an endpoint.
 - m!=n – This case is unreachable and should never happen. Completions always use the same TC as the corresponding request.
- C(m)->P(n), C(m)->N(n) – These cases are unreachable and should never happen due to the fact that completion buffers must be preallocated.
- C(m)->C(n)
 - m=n – This case is illegal and will lead to deadlock, except when a Completion is being forwarded from one port to another of a Switch or Root Complex.
 - m!=n – This case will occur if N(n)->N(m) dependencies are allowed.

Others of these potential dependencies may be legal when comprehended as part of a specific usage model. For example, these cases:

- P(m)->P(n), N(m)->N(n), P(m)->N(n), N(m)->P(n) where m!=n

might exist where a device processes incoming requests to complete those requests by issuing modified versions of those requests using a different Traffic Class (for example remapping the first requests address and generating the new request with the resulting address). In these cases, suitable rules must be applied to prevent circular dependencies that would lead to deadlock or livelock.

Examples of devices that may find the above mappings useful:

- Bridges to complex protocols that require state to be save/restored to/from host memory, i.e. PCI Express to Infiniband bridges.
- Messaging engines that must do address translation based upon page tables stored in host memory.
- UMA graphics devices that store their frame buffer in host memory.