PCI-SIG ENGINEERING CHANGE NOTICE

TITLE:	Alternative Routing-ID Interpretation (ARI)
DATE:	August 7, 2006; last updated June 4, 2007
AFFECTED DOCUMENTS:	PCI Express Base Specification version 2.0; PCI Express Base Specification version 1.1 & related 1.1 ECNs
SPONSOR:	Hewlett-Packard

Part I

1. Summary of the Functional Changes

For virtualized and non-virtualized environments, a number of PCI-SIG member companies have requested that the current constraints on number of Functions allowed per multi-Function Device be increased to accommodate the needs of next generation I/O implementations. This ECR specifies a new method to interpret the Device Number and Function Number fields within Routing IDs, Requester IDs, and Completer IDs, thereby increasing the number of Functions that can be supported by a single Device.

This ECR specifies how these interpretations are enabled while insuring interoperability with existing 1.1- and 2.0-based implementations. The changes in this ECR are shown in the context of the 2.0 Base specification, which is essentially the 1.1 Base specification with 1.1-based ECNs and errata incorporated.

This ECR impacts multi-Function devices at an Upstream Port, Root Ports, and Switches.

2. Benefits as a Result of the Changes

Alternative Routing-ID Interpretation (ARI) enables next generation I/O implementations to support an increased number of concurrent users of a multi-Function device while providing the same level of isolation and controls found in existing implementations. While ARI obviously benefits the virtualized operating environments where each Function can be uniquely assigned to a guest OS, ARI also benefits non-virtualized environments where, e.g. due to increased process improvements, a large number of I/O Functions can be integrated into a single Device.

3. Assessment of the Impact

ARI is optional normative functionality that is applicable to multi-Function devices at Upstream Ports (“ARI Devices”), and Downstream Ports (Root Ports or Switches). An ARI Device interprets its directly associated IDs (Routing, Requester, and Completer) as having an 8-bit Function Number instead of the traditional 3-bit Function Number. An ARI Device has its associated Device Number implied to be 0 rather than specified by an ID field.

ARI is managed via a set of capability and control register bits within the Device Capabilities 2 register of the PCI Express Capability structure and a new ARI Capability structure. Software detects ARI Devices and configures the Downstream Port immediately above each ARI Device to forward TLPs using the Alternative Routing-ID Interpretation. Given the optional normative nature, the technology (with a few exceptions) is backward compatible with existing hardware and software, i.e. if the capabilities are not enabled, the hardware and software operate in compliance with the existing PCI Express 1.1 and 2.0 Base Specifications.

With some existing Link controls, the required logic for ARI Devices is simplified somewhat compared to earlier Base specifications. Instead of requiring “voting” logic between the different Functions (up to 256 of them), certain Link controls are managed solely by the settings in Function 0, and the corresponding Link controls in the other Functions are ignored by hardware. This may impact backward compatibility with some hypothetical software, but is expected to be compatible with software that configures all Functions with the same Link settings as generally recommended.

4. Analysis of the Hardware Implications

ARI requires new hardware and is therefore optional normative. Hardware that is not ARI capable will treat ARI-capable hardware per the existing PCI Express Base Specification and be fully interoperable.

For Routing IDs, Requester IDs, and Completer IDs associated with each ARI Device, the traditional 5-bit Device Number and 3-bit Function Number fields are instead interpreted as a single 8-bit Function Number field. Thus with ARI Devices, the Function Number can used to address up to 256 Functions.

When ARI Forwarding is enabled in an ARI Downstream Port, the logic that determines when to turn a Type 1 Configuration Request into a Type 0 Configuration Request no longer enforces a restriction on the traditional Device Number field in the Requester ID being 0. For the ARI Device immediately below the Downstream Port, what was the Device Number field is now used instead as part of an 8-bit Function Number field.

When ARI Forwarding is not enabled in ARI Downstream Ports, ID-based Routing is performed per the existing PCI Express Base Specification. Restrictions on the number of Functions and Device Number usage are per the existing PCI Express Base Specification.

5. Analysis of the Software Implications

ARI requires new software to enable its functionality and thus is optional normative. Software that does not comprehend the new functionality will interoperate with ARI capable hardware per the existing PCI Express Base Specification. With an ARI Device, ARI functionality is managed via a new ARI Capability structure. With an ARI Downstream Port, ARI functionality is managed via the Device Capabilities 2 and Device Control 2 registers of the PCI Express Capability structure.

Part II

Detailed Description of the change

Modify Terms and Acronyms as shown

ARI	Alternative Routing-ID Interpretation. Applicable to Requester IDs and Completer IDs as well as Routing IDs.
ARI Downstream Port	A Switch Downstream Port or Root Port that supports ARI Forwarding.
ARI Device	A multi-Function Device associated with an Upstream Port, capable of supporting up to 256 Functions.
Completer ID	The combination of a Completer's Bus Number, Device Number, and Function Number that uniquely identifies the Completer of the Request. With an ARI Completer ID, bits traditionally used for the Device Number field are used instead to expand the Function Number field, and the Device Number is implied to be 0.
Extended Function	Within an ARI Device, a Function whose Function Number is greater than 7. Extended Functions are accessible only after ARI-aware software has enabled ARI Forwarding in the Downstream Port immediately above the ARI Device.
Function Group	Within an ARI Device, a configurable set of Functions that are associated with a single Function Group Number. Function Groups can optionally serve as the basis for VC arbitration or access control between multiple Functions within the ARI Device.
Requester ID	The combination of a Requester's Bus Number, Device Number, and Function Number that uniquely identifies the Requester. With an ARI Requester ID, bits traditionally used for the Device Number field are used instead to expand the Function Number field, and the Device Number is implied to be 0.

Modify Section 2.2.2. as shown

2.2.2. TLPs with Data Payloads - Rules

...

- ❑ The Transmitter of a TLP with a data payload must not allow the data payload length as given by the TLP's Length [ ] field to exceed the length specified by the value in the Max_Payload_Size field of the Transmitter's Device Control register taken as an integral number of DW (see Section 7.8.4).
  - For ARI Devices, the Max_Payload_Size is determined solely by the setting in Function 0. The Max_Payload_Size settings in other Functions are ignored.
  - For an Upstream Port associated with a non-ARI multi-Function device whose Max_Payload_Size settings are identical across all Functions, a transmitted TLP's data payload must not exceed the common Max_Payload_Size setting.
  - For an Upstream Port associated with a non-ARI multi-Function device whose Max_Payload_Size settings are not identical across all Functions, a transmitted TLP's data payload must not exceed a Max_Payload_Size setting whose determination is implementation specific.

...

- ❑ The size of the data payload of a Received TLP as given by the TLP's Length [ ] field must not exceed the length specified by the value in the Max_Payload_Size field of the Receiver's Device Control register taken as an integral number of DW (see Section 7.8.4).
  - Receivers must check for violations of this rule. If a Receiver determines that a TLP violates this rule, the TLP is a Malformed TLP
    - ♦ This is a reported error associated with the Receiving Port (see Section 6.2)
  - For ARI Devices, the Max_Payload_Size is determined solely by the setting in Function 0. The Max_Payload_Size settings in other Functions are ignored.
  - For an Upstream Port associated with a non-ARI multi-Function device whose Max_Payload_Size settings are identical across all Functions, the Receiver is required to check the TLP's data payload size against the common Max_Payload_Size setting.
  - For an Upstream Port associated with a non-ARI multi-Function device whose Max_Payload_Size settings are not identical across all Functions, the Receiver is required to check the TLP's data payload against a Max_Payload_Size setting whose determination is implementation specific.

Modify Section 2.2.4.2. as shown

2.2.4.2. ID Based Routing Rules

- ❑ ID routing is used with Configuration Requests, optionally with Vendor_Defined Messages (see Section 2.2.8.6), and with Completions
- ❑ ID routing uses the Bus, Device, and Function Numbers (as applicable) to specify the destination for the TLP:
  - For non-ARI Routing IDs, the Bus, Device, and (3-bit) Function Number to TLP header mapping is shown in Table 2-7.
  - For ARI Routing IDs, the Bus and (8-bit) Function Number to TLP header mapping is shown in Table 2-7a.

- Two ID routing formats are specified, one used with a 4 DW header (see Figure 2-7) and one used with a 3 DW header (see Figure 2-8)
  - Header field locations are the same for both formats, and are given in Table 2-7

Table 2-7: Header Field Locations for (non-ARI) ID Routing

Field	Header Location
Bus Number[7:0]	Bits 7:0 of Byte 8
Device Number[4:0]	Bits 7:3 of Byte 9
Function Number[2:0]	Bits 2:0 of Byte 9

Table 2-7a: Header Field Locations for ARI ID Routing

Field	Header Location
Bus Number[7:0]	Bits 7:0 of Byte 8
Function Number[7:0]	Bits 7:0 of Byte 9

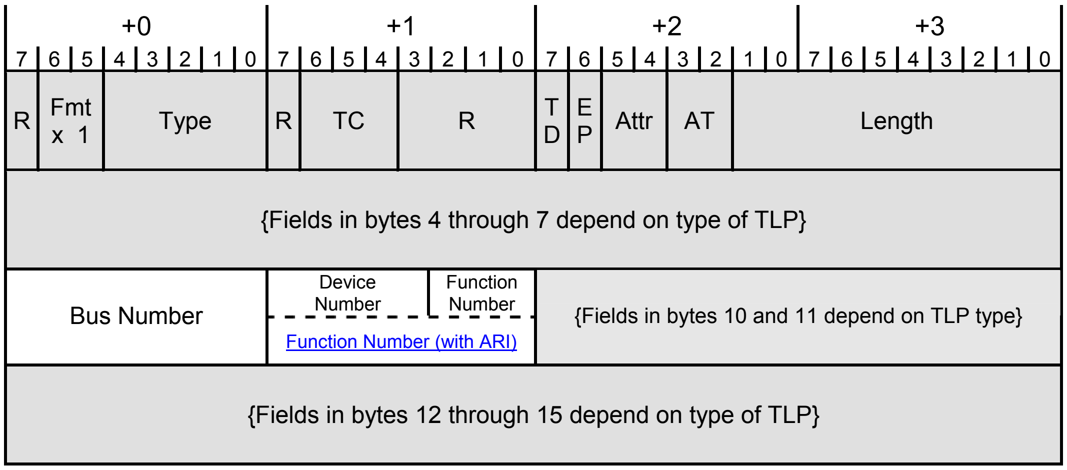
+0
+1
+2
+3
7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0
R
Fmt x 1
Type
R
TC
R
TD
EP
Attr
AT
Length
{Fields in bytes 4 through 7 depend on type of TLP}
Bus Number
Device Number
Function Number
Function Number (with ARI)
{Fields in bytes 10 and 11 depend on TLP type}
{Fields in bytes 12 through 15 depend on type of TLP}

Figure 2-7: ID Routing with 4 DW Header

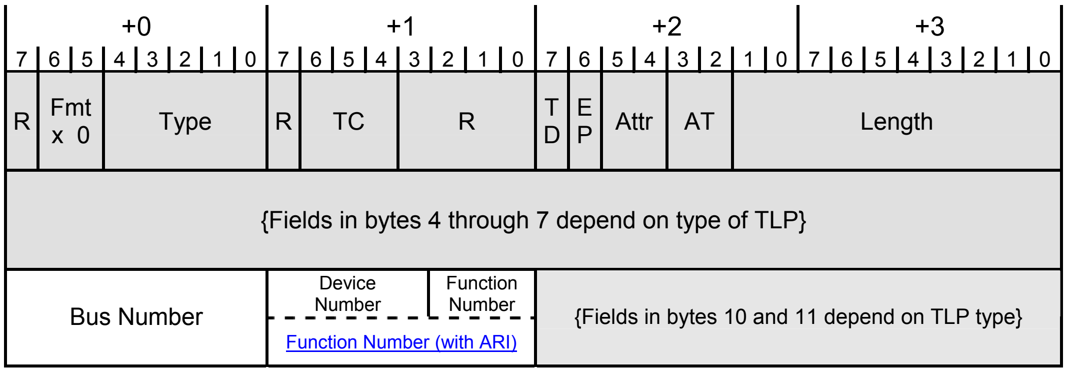

Figure 2-8: ID Routing with 3 DW Header

Modify Section 2.2.6.2. as shown

2.2.6.2. Transaction Descriptor – Transaction ID Field

The Transaction ID field consists of two major sub-fields: Requester ID and Tag as shown in Figure 2-11.

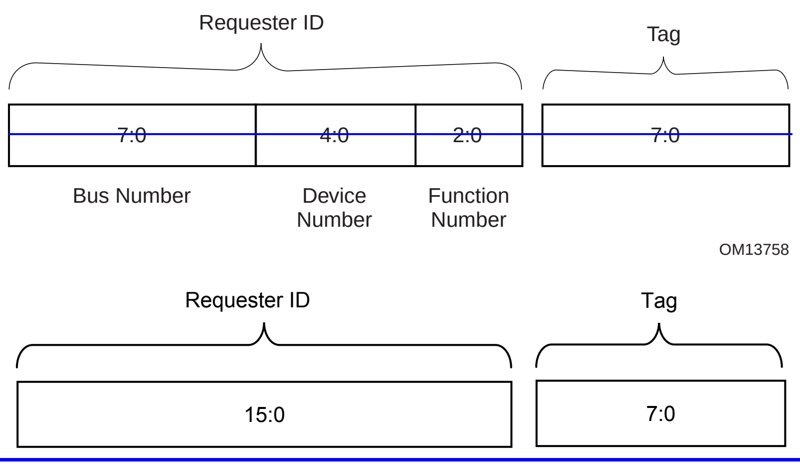

Figure 2-11: Transaction ID

…

- ❑ Functions must capture the Bus and Device Numbers[1] supplied with all Type 0 Configuration Write Requests completed by the Function and supply these numbers in the Bus and Device Number fields of the Requester ID[2] for all Requests initiated by the Device/Function.

  Exception: The assignment of Bus and Device Numbers to the Devices within a Root Complex, and Device Numbers to the downstream ports within a switch, may be done in an implementation specific way.

  Note that the Bus Number and Device Number[3] may be changed at run time, and so it is necessary to re-capture this information with each and every Configuration Write Request.

…

- ❑ Each Function associated with a Device must be designed to respond to a unique Function Number for Configuration Requests addressing that Device.
  Note: Each non-ARI Device may contain up to eight Functions. Each ARI Device may contain up to 256 Functions.

[1] In ARI Devices, Functions are only required to capture the Bus Number. ARI Devices are permitted to retain the captured Bus Number on either a per-Device or a per-Function basis. If the captured Bus Number is retained on a per-Device basis, all Functions are required to update and use the common Bus Number.

[2] An ARI Requester ID does not contain a Device Number field. See Section 2.2.4.2.

[3] With ARI Devices, only the Bus Number can change.

Modify Section 2.2.8.1. as shown

2.2.8.1. *INTx Interrupt Signaling - Rules*

…

- ❑ The Length field is reserved.
- ❑ With Assert_INTx/Deassert_INTx Messages, the Function Number field in the Requester ID must be 0. Note that the Function Number field is a different size for non-ARI and ARI Requester IDs.

…

Table 2-13: INTx Mechanism Messages

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support[4]				~~Req ID~~[5]	Description/Comments
			RC	Ep	Sw	Br		
Assert_INTA	0010 0000	100	All:				~~BD~~	Assert INTA virtual wire
			r		tr			Note: These Messages are used for PCI 3.0 compatible INTx emulation.
			As Required:					
				t		t		
Assert_INTB	0010 0001	100	All:				~~BD~~	Assert INTB virtual wire
			r		tr			
			As Required:					
				t		t		
Assert_INTC	0010 0010	100	All:				~~BD~~	Assert INTC virtual wire
			r		tr			
			As Required:					
				t		t		

[4] Abbreviations:
RC = Root Complex
Sw = Switch (only used with "Link" routing)
Ep = Endpoint
Br = PCI Express (primary) to PCI/PCI-X (secondary) Bridge
r = Supports as Receiver
t = Supports as Transmitter

Note that Switches must support passing Messages on all legal routing paths. Only Messages specifying Local (0100b) routing or a reserved field value are terminated locally at the Receiving Port on a Switch.

[5] ~~The Requester ID includes sub-fields for Bus Number, Device Number, and Function Number. Some Messages are not associated with specific Functions in a component, and for such Messages this field is Reserved; this is shown in this column using a code. Some messages can be used in more than one context and, therefore, more than one code may be listed. The codes in this column are:~~

~~BD = Bus Number and Device Number included; Function Number is Reserved~~
~~BDF = Bus Number, Device Number, and Function Number are included~~

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support[4]				~~Req ID[5]~~	Description/Comments
			RC	Ep	Sw	Br		
Assert_INTD	0010 0011	100	All:				~~BD~~	Assert INTD virtual wire
			r		tr			
			As Required:					
				t		t		
Deassert_INTA	0010 0100	100	All:				~~BD~~	De-assert INTA virtual wire
			r		tr			
			As Required:					
				t		t		
Deassert_INTB	0010 0101	100	All:				~~BD~~	De-assert INTB virtual wire
			r		tr			
			As Required:					
				t		t		
Deassert_INTC	0010 0110	100	All:				~~BD~~	De-assert INTC virtual wire
			r		tr			
			As Required:					
				t		t		
Deassert_INTD	0010 0111	100	All:				~~BD~~	De-assert INTD virtual wire
			r		tr			
			As Required:					
				t		t		

Add following the existing Implementation Note at the end of Section 2.2.8.1.

IMPLEMENTATION NOTE

Virtual Wire Mapping for INTx Interrupts From ARI Devices

The implied Device Number for an ARI Device is 0. When ARI-aware software (including BIOS and operating system) enables ARI Forwarding in the Downstream Port immediately above an ARI Device in order to access its Extended Functions, software must comprehend that the Downstream Port will use Device Number 0 for the virtual wire mappings of INTx interrupts coming from all Functions of the ARI Device. If non-ARI-aware software attempts to determine the virtual wire mappings for Extended Functions, it can come up with incorrect mappings by examining the traditional Device Number field and finding it to be non-0.

Modify Section 2.2.8.2. as shown

2.2.8.2. Power Management Messages

…

- ❑ The Length field is reserved.
- ❑ With PM_Active_State_Nak Messages, the Function Number field in the Requester ID must contain the Function Number of the Downstream Port that sent the Message, or else 000b for compatibility with earlier revisions of this specification.
- ❑ With PME_TO_Ack Messages, the Function Number field in the Requester ID must be reserved, or else for compatibility with earlier revisions of this specification must contain the Function Number of one of the Functions associated with the Upstream Port. Note that the Function Number field is a different size for non-ARI and ARI Requester IDs.

…

Table 2-15: Power Management Messages

Name	**Code[7:0] (b)**	**Routing r[2:0] (b)**	**Support**				**~~Req ID~~**	**Description/Comments**
			RC	**Ep**	**Sw**	**Br**		
PM_Active_State_Nak	0001 0100	100	t	r	tr	r	~~BDF[6]~~	Terminate at Receiver
PM_PME	0001 1000	000	All:				~~BDF~~	Sent Upstream by PME-requesting component. Propagates Upstream.
			r		tr	t		
			If PME supported:					
				t				
PME_Turn_Off	0001 1001	011	t	r		r	~~BDF~~	Broadcast Downstream
PME_TO_Ack	0001 1011	101	r	t		t	~~BD[7]~~	Sent Upstream by Endpoint. Sent Upstream by Switch when received on all Downstream Ports.
			(Note: Switch handling is special)					

6 ~~Also permitted to be BD for compatibility with earlier revisions of this specification.~~

7 ~~Also permitted to be BDF for compatibility with earlier revisions of this specification.~~

Modify Section 2.2.8.3. as shown

2.2.8.3. Error Signaling Messages

…

❑ The Length field is reserved.

❑ With Error Signaling Messages, the Function Number field in the Requester ID must indicate which Function is signaling the error. Note that the Function Number field is a different size for non-ARI and ARI Requester IDs.

…

Table 2-16: Error Signaling Messages

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support				~~Req ID~~	Description/Comments
			RC	Ep	Sw	Br		
ERR_COR	0011 0000	000	r	t	tr	t	~~BD~~ BDF	This Message is issued when the component or device detects a correctable error on the PCI Express interface.
ERR_NONFATAL	0011 0001	000	r	t	tr	t	~~BD~~ BDF	This Message is issued when the component or device detects a Non-fatal, uncorrectable error on the PCI Express interface.
ERR_FATAL	0011 0011	000	r	t	tr	t	~~BD~~ BDF	This Message is issued when the component or device detects a Fatal, uncorrectable error on the PCI Express interface.

Modify Section 2.2.8.4. as shown

2.2.8.4. Locked Transactions Support

…

❑ The Length field is reserved.

❑ With Unlock Messages, the Function Number field in the Requester ID is reserved.

…

Table 2-17 Unlock Message

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support				~~Req ID~~	Description/Comments
			RC	Ep	Sw	Br		
Unlock	0000 0000	011	t	r	tr	r	~~BD~~	Unlock Completer

Modify Section 2.2.8.5. as shown

2.2.8.5. Slot Power Limit Support

…

Table 2-18: Set_Slot_Power_Limit Message

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support				~~Req ID~~	Description/Comments
			RC	Ep	Sw	Br		
Set_Slot_Power_Limit	0101 0000	100	t	r	tr	r	~~BDF~~	Set Slot Power Limit in Upstream Port

Modify Section 2.2.8.6. as shown

2.2.8.6. Vendor_Defined Messages

…

❑ The Vendor_Defined Messages (see Table 2-19) use the header format shown in Figure 2-18.

- The Requester ID is implementation specific.
- If the Route by ID routing is used, bytes 8 and 9 form a 16-bit field for the destination ID

…

Table 2-19: Vendor_Defined Messages

Name	Code[7:0] (b)	Routing r[2:0] (b)	Support				~~Req ID~~	Description/Comments
			RC	Ep	Sw	Br		
Vendor_Defined Type 0	0111 1110	000, 010, 011, 100	See Note 1.				~~See Note 2.~~	Triggers detection of UR by Completer if not implemented.
Vendor_Defined Type 1	0111 1111	000, 010, 011, 100	See Note 1.				~~See Note 2.~~	Silently discarded by Completer if not implemented.

Note 1: Transmission by Endpoint/Root Complex/Bridge is implementation specific. Switches must forward received Messages using Routing[2:0] field values of 000b, 010b, and 011b.

~~Note 2: Implementation specific.~~

Modify Section 2.2.8.7. as shown

2.2.8.7. *Ignored Messages*

…

Table 2-20: Ignored Messages

Name	**Code[7:0] (b)**	**Routing r[2:0] (b)**	**Support**				**~~Req ID~~**	**Description/Comments**
			RC	**Ep**	**Sw**	**Br**		
Ignored Message	0100 0001	100						
Ignored Message	0100 0011	100						
…	…	…						

Modify Section 2.2.9. as shown

2.2.9. Completion Rules

…

- ❑ The Completer ID[15:0] is a 16-bit value that is unique for every PCI Express Function within a Hierarchy (see Figures 2-20 and 2-20a)

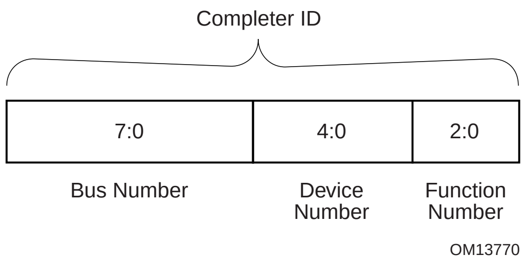

Figure 2-20: (Non-ARI) Completer ID

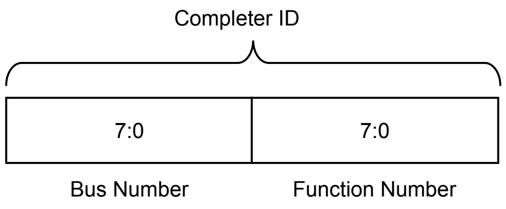

Figure 2-20a: ARI Completer ID

- ❑ Functions must capture the Bus and Device Numbers[8] supplied with all Type 0 Configuration Write Requests completed by the Function, and supply these numbers in the Bus and Device Number fields of the Completer ID[9] for all Completions generated by the Device/Function.

---

[8] With ARI Devices, Functions are only required to capture the Bus Number. ARI Devices are permitted to retain the captured Bus Number on either a per-Device or a per-Function basis. See Section 2.2.6.2.

[9] An ARI Completer ID does not contain a Device Number field. See Section 2.2.4.2.

…

❑ In some cases, a Completion with the UR status may be generated by a multi-Function device without associating the Completion with a specific Function within the device – in this case, the Function Number field[10] is Reserved.

Modify Section 2.6.1.2. as shown

2.6.1.2. FC Information Tracked by Receiver

…

❑ For non-infinite PD and CPLD types, when the number of available credits is less than Max_Payload_Size, an UpdateFC FCP must be scheduled for Transmission each time one or more units of that type are made available by TLPs processed

- For ARI Devices, the Max_Payload_Size is determined solely by the setting in Function 0. The Max_Payload_Size settings in other Functions are ignored.
- For a non-ARI multi-Function device whose Max_Payload_Size settings are identical across all Functions, the common Max_Payload_Size setting or larger must be used.
- For a non-ARI multi-Function device whose Max_Payload_Size settings are not identical across all Functions, the selected Max_Payload_Size setting is implementation specific, but it is recommended to use the largest Max_Payload_Size setting across all Functions.

…

IMPLEMENTATION NOTE

Flow Control Update Frequency

…

$$\frac{(Max\_Payload\_Size + TLPOverhead) * UpdateFactor}{LinkWidth} + InternalDelay$$

where:

Max_Payload_Size	The value in the Max_Payload_Size field of the Device Control register. For a multi-Function device, it is recommended that the smallest Max_Payload_Size setting across all Functions[11] is used.

…

[10] Note: with an ARI Completer ID, the Function Number field is 8 bits.

[11] For ARI Devices, the Max_Payload_Size is determined solely by the setting in Function 0, and thus the settings in the other Functions should be ignored.

Modify Section 3.5.2.1. as shown

3.5.2.1. LCRC and Sequence Number Rules (TLP Transmitter)

…

$$\left(\frac{(Max\_Payload\_Size + TLPOverhead) * AckFactor}{LinkWidth} + InternalDelay\right) * 3$$

where

Max_Payload_Size	is the value in the Max_Payload_Size field of the Device Control register. For ARI Devices, the Max_Payload_Size is determined solely by the setting in Function 0. For a non-ARI multi-Function device whose Max_Payload_Size settings are identical across all Functions, the common Max_Payload_Size setting must be used. For a non-ARI multi-Function device whose Max_Payload_Size settings are not identical across all Functions, the selected Max_Payload_Size setting is implementation specific, but it is recommended to use the largest Max_Payload_Size setting across all Functions.

Modify Section 3.5.3.1. as shown

3.5.3.1. LCRC and Sequence Number Rules (TLP Receiver)

…

$$\frac{(Max\_Payload\_Size + TLPOverhead) * AckFactor}{LinkWidth} + InternalDelay + Tx\_L0s\_Adjustment$$

where

Max_Payload_Size	is the value in the Max_Payload_Size field of the Device Control register. For ARI Devices, the Max_Payload_Size is determined solely by the setting in Function 0. For a non-ARI multi-Function device whose Max_Payload_Size settings are identical across all Functions, the common Max_Payload_Size setting must be used. For a non-ARI multi-Function device whose Max_Payload_Size settings are not identical across all Functions, the selected Max_Payload_Size setting is implementation specific, but it is recommended to use the smallest Max_Payload_Size setting across all Functions.

Modify Section 5.3.2. as shown

5.3.2. PM Software Control of the Link Power Management State

…

The following rules relate to PCI-PM compatible power management:

…

- ❑ The Upstream Port of a non-ARI multi-Function device must not initiate a Link state transition to L1 (on behalf of PCI-PM) until all of its Functions have been programmed to a non-D0 D-state.
- ❑ The Upstream Port of an ARI Device must not initiate a Link state transition to L1 (on behalf of PCI-PM) until at least one of its Functions has been programmed to a non-D0 state, and all of its Functions are either in a non-D0 state or the $D0_{uninitialized}$ state.

Modify Section 5.4.1. as shown

5.4.1. Active State Power Management (ASPM)

<following Implementation Note on Isochronous Traffic and ASPM>

For ARI Devices, ASPM Control is determined solely by the setting in Function 0, regardless of Function 0's D-state. The ASPM Control settings in other Functions are ignored by the component.

An Upstream Port with a non-ARI multi-Function device may be programmed with different values in its respective ~~Active_PM_En~~ ASPM Control registers of each Function. The policy for such a component will be dictated by the most active common denominator among all D0 Functions according to the following rules:

Modify Section 6.13.1.2. as shown

6.13.1.2. ACS Functions in Multi-Function Devices

…

- o ACS P2P Egress Control: implementation is optional; is based on Function Numbers or Function Group Numbers; controls peer-to-peer Requests between different Functions within a multi-Function device.

  ACS P2P Egress Control is subject to interaction with ACS P2P Request Redirect and ACS Direct Translated P2P mechanisms (if implemented). Refer to Section 6.13.3 for more information.

  Each Function within a multi-Function device that supports ACS P2P Egress Control can be selectively enabled to block peer-to-peer communication with other Functions or Function Groups[12] within the device. This is configured on a per Function basis.

[12] ACS Function Groups capability is optional for ARI Devices that implement ACS P2P Egress Controls.

~~Conceptually, the Functions are interconnected through a transparent embedded switch, and access control uses logic similar to that in a Switch but validates whether the Function field within the Requester ID of a Request is allowed or not.~~

With ACS P2P Egress Control in a multi-Function devices, controls in the "sending" Function determine if the Request is blocked, and if so, the "sending" Function handles the ACS Violation error per Section 6.13.4.

When ACS Function Groups are enabled in an ARI Device, ACS P2P Egress Controls are enforced on a per Function Group basis instead of a per Function basis. See Section 6.14.

Modify Section 6.13.3. as shown

6.13.3. ACS Peer-to-Peer Control Interactions

…

Table 6-8: ACS P2P Request Redirect and ACS P2P Egress Control Interactions

Control Bit E (b)	Control Bit R (b)	Egress Control Vector Bit for the Associated Egress Switch Port, Root Port, Function, or Function Group	Required Handling for Peer-to-Peer Requests
0	0	X – Don't care	Route directly to peer-to-peer target
0	1	X – Don't Care	Redirect upstream
1	0	1	Handle as an ACS Violation
1	0	0	Route directly to peer-to-peer target
1	1	1	Redirect upstream
1	1	0	Route directly to peer-to-peer target

Add new Section 6.14.

6.14. Alternative Routing-ID Interpretation (ARI)

Routing IDs, Requester IDs, and Completer IDs are 16-bit identifiers traditionally composed of three fields: an 8-bit Bus Number, a 5-bit Device Number, and a 3-bit Function Number. With ARI, the 16-bit field is interpreted as two fields instead of three: an 8-bit Bus Number and an 8-bit Function Number – the Device Number field is eliminated. This new interpretation enables an ARI Device to support up to 256 Functions [0..255] instead of 8 Functions [0..7].

ARI is controlled by a new set of optional capability and control register bits. These provide:

- Software the ability to detect whether a component supports ARI.

- Software the ability to configure a ARI Downstream Port so the logic that determines when to turn a Type 1 Configuration Request into a Type 0 Configuration Request no longer enforces a restriction on the traditional Device Number field being 0.

- Software the ability to configure an ARI Device to assign each Function to a Function Group. Controls based on Function Groups may be preferable when finer granularity controls based on individual Functions are not required.

    - If Multi-Function VC arbitration is supported and enabled, arbitration can optionally be based on Function Groups instead of individual Functions.

    - If ACS P2P Egress Controls are supported are enabled, access control can optionally be based on Function Groups instead of individual Functions.

The following illustrates an example flow for enabling these capabilities and provides additional details on their usage:

1. Software enumerates the PCI Express hierarchy and determines whether the ARI capability is supported.

    a. For an ARI Downstream Port, the capability is communicated through the Device Capabilities 2 register.

    b. For an ARI Device, the capability is communicated through the ARI Capability structure.

    c. ARI has no impact on the base enumeration algorithms used in platforms today.

2. Software enables ARI functionality in each component.

    a. In an ARI Downstream Port immediately above an ARI Device, software sets the ARI Forwarding Enable bit in the Device Control 2 register. Setting this bit ensures the logic that determines when to turn a Type 1 Configuration Request into a Type 0 Configuration Request no longer enforces a restriction on the traditional Device Number field being 0.

    b. In an ARI Device, Extended Functions are always implicitly enabled. However, once ARI-aware software enables ARI Forwarding in the Downstream Port immediately above the ARI Device, ARI-aware software must discover and configure the Extended Functions.

    c. If an ARI Device implements a Multi-Function VC Capability structure with Function arbitration, and also implements MFVC Function Groups, ARI-aware software categorizes Functions into Function Groups.

        i. Each Function is assigned to a Function Group represented by a Function Group Number.

        ii. A maximum of 8 Function Groups can be configured.

iii. Within the Multi-Function VC Arbitration Table, a Function Group Number is used in place of a Function Number in each arbitration slot.

1. Arbitration occurs on a Function Group basis instead of an individual Function basis.

2. All other aspects of Multi-Function VC arbitration remain unchanged. See section 7.17.10 for additional details.

iv. Function arbitration within each Function Group is implementation-specific.

d. If an ARI Device supports ACS P2P Egress Control, access control can be optionally implemented on a Function Group basis.

e. To improve the enumeration performance and create a more deterministic solution, software can enumerate Functions through a linked list of Function Numbers. The next linked list element is communicated through each Function's ARI Control Register.

i. Function 0 acts as the head of a linked list of Function Numbers. Software detects a non-zero Function Number within the ARI Control Register as the next Function within the linked list. Software issues a configuration probe using the Bus Number captured by the Device and the Function Number derived from the ARI Control Register to locate the associated Function's configuration space.

ii. Function Numbers may be sparse and non-sequential in their consumption by an ARI Device.

With an ARI Device, the Phantom Functions Supported field within each Function's Device Capabilities register (see Section 7.8.3, Table 7-11) must be set to 00b to indicate that Phantom Functions are not supported. The Extended Tag Field Enable bit (see Section 7.8.4, Table 7-12) can still be used to enable each Function to support up to 256 outstanding Requests.

Figure 6-xx shows an example system topology with 2 ARI Devices, one below a Root Port and one below a Switch. For access to Extended Functions in ARI Device X, Root Port A must support ARI Forwarding and have it enabled by software. For access to Extended Functions in ARI Device Y, Switch Downstream Port D must support ARI Forwarding and have it enabled by software. With this configuration, it is recommended that software not enable ARI Forwarding in Root Port B or Switch Downstream Port C.

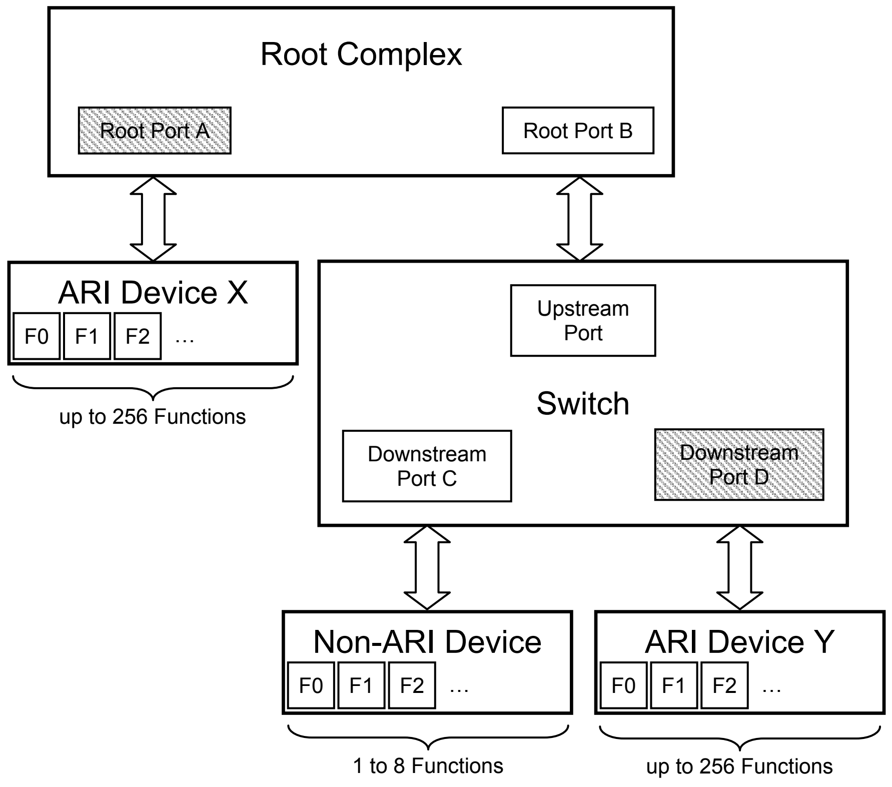

Figure 6-xx Example System Topology With ARI Devices

IMPLEMENTATION NOTE

ARI Forwarding Enable Being Set Inappropriately

It is strongly recommended that software in general Set the ARI Forwarding Enable bit in a Downstream Port only if software is certain that the device immediately below the Downstream Port is an ARI Device. If the bit is Set when a non-ARI Device is present, the non-ARI Device can respond to Configuration Space accesses under what it interprets as being different Device Numbers, and its Functions can be aliased under multiple Device Numbers, generally leading to undesired behavior.

Following a hot-plug event below a Downstream Port, it is strongly recommended that software Clear the ARI Forwarding Enable bit in the Downstream Port until software determines that a newly added component is in fact an ARI Device.

IMPLEMENTATION NOTE

ARI Forwarding Enable Setting At Firmware/OS Control Handoff

It is strongly recommended that firmware not have the ARI Forwarding Enable bit Set in a Downstream Port upon control handoff to an operating system unless firmware knows that the operating system is ARI-aware. With this bit Set, a non-ARI-aware operating system might be able to discover and enumerate Extended Functions in an ARI Device below the Downstream Port, but such an operating system would generally not be able to manage Extended Functions successfully, since it would interpret there being multiple Devices below the Downstream Port instead of a single ARI Device. As one example of many envisioned problems, the interrupt binding for INTx virtual wires would not be consistent with what the non-ARI-aware operating system would expect.

Modify Section 7.2.2. as shown

7.2.2. PCI Express Enhanced Configuration Access Mechanism (ECAM)

...

Table 7-1: Enhanced Configuration Address Mapping

Memory Address[13]	**PCI Express Configuration Space**
A[(20 + n – 1):20]	Bus Number $1 \le n \le 8$
A[19:15]	Device Number
A[14:12]	Function Number
A[11:8]	Extended Register Number
A[7:2]	Register Number
A[1:0]	Along with size of the access, used to generate Byte Enables

Note: for Requests targeting Extended Functions in an ARI Device, A[19:12] represents the (8-bit) Function Number, which replaces the (5-bit) Device Number and (3-bit) Function Number fields above.

[13] This address refers to the byte-level address from a software point of view.

Modify Section 7.3.1. as shown

7.3.1. Device Number

~~As in conventional PCI and PCI-X, all~~ With non-ARI Devices, PCI Express components are restricted to implementing a single Device Number on their primary interface (Upstream Port), but may implement up to eight independent Functions within that Device Number. Each internal Function is selected based on decoded address information that is provided as part of the address portion of Configuration Request packets.

~~Switches and Root Complexes~~ Downstream Ports that do not have ARI Forwarding enabled must associate only Device 0 with the device attached to the Logical Bus representing the Link from ~~a Switch Downstream Port or a Root~~ the Port. Configuration Requests targeting the Bus Number associated with a Link specifying Device Number 0 are delivered to the device attached to the Link; Configuration Requests specifying all other Device Numbers (1-31) must be terminated by the Switch Downstream Port or the Root Port with an Unsupported Request Completion Status (equivalent to Master Abort in PCI). Non-ARI Devices must not assume that Device Number 0 is associated with their Upstream Port, but must capture their assigned Device Number as discussed in Section 2.2.6.2. Non-ARI Devices must respond to all Type 0 Configuration Read Requests, regardless of the Device Number specified in the Request.

Switches, and components wishing to incorporate more than eight Functions at their Upstream Port, may implement one or more "virtual switches," represented by multiple Type 1 (PCI-PCI Bridge) configuration space headers as illustrated in Figure 7-2. These virtual switches serve to allow fan-out beyond eight Functions. Since Switch Downstream Ports may appear on any Device Number, in this case all address information fields (Bus, Device, and Function Numbers) must be completely decoded to access the correct register. Any configuration access targeting an unimplemented bus, device, or Function must return a Completion with Unsupported Request Completion Status.

With an ARI Device, its Device Number is implied to be 0 rather than specified by a field within an ID. The traditional 5-bit Device Number and 3-bit Function Number fields in its associated Routing IDs, Requester IDs, and Completer IDs are interpreted as a single 8-bit Function Number. See Section 6.14. Any Type 0 Configuration Request targeting an unimplemented Function in an ARI Device must be handled as an Unsupported Request.

If an ARI Downstream Port has ARI Forwarding enabled, the logic that determines when to turn a Type 1 Configuration Request into a Type 0 Configuration Request no longer enforces a restriction on the traditional Device Number field being 0.

Modify Section 7.3.2. as shown

7.3.2. Configuration Transaction Addressing

PCI Express Configuration Requests use the following addressing fields:

- Bus Number – PCI Express maps logical PCI Bus Numbers onto PCI Express Links such that PCI 3.0 compatible configuration software views the configuration space of a PCI Express Hierarchy as a PCI hierarchy including multiple bus segments.
- Device Number – Device Number association is discussed in Section 7.3.1. When an ARI Device is targeted and the Downstream Port immediately above it is enabled for ARI Forwarding, the Device Number is implied to be 0, and the traditional Device Number field is used instead as part of an 8-bit Function Number field. See Section 6.14.
- Function Number – PCI Express also supports multi-Function devices using the same discovery mechanism as PCI 3.0. With ARI Devices, discovery and enumeration of Extended Functions require ARI-aware software. See Section 6.14.

Modify Section 7.8.3. as shown

7.8.3. Device Capabilities Register

…

Table 7-11: Device Capabilities Register

Bit Location	Register Description	Attributes
	…..	
4:3	**Phantom Functions Supported** – This field indicates the support for use of unclaimed Function Numbers to extend the number of outstanding transactions allowed by logically combining unclaimed Function Numbers (called Phantom Functions) with the Tag identifier (see Section 2.2.6.2 for a description of Tag Extensions). With every Function in an ARI Device, the Phantom Functions Supported field must be set to 00b. The remainder of this field description applies only to non-ARI multi-Function devices. …	RO

Modify Section 7.8.4. as shown

7.8.4. Device Control Register (Offset 08h)

…

Table 7-12: Device Control Register

Bit Location	Register Description	Attributes
…	**…**	…
7:5	**Max_Payload_Size** – This field sets maximum TLP payload size for the Function. … … System software is not required to program the same value for this field for all the Functions of a multi-Function device. Refer to Section 2.2.2 for important guidance. For ARI Devices, Max_Payload_Size is determined solely by the setting in Function 0. The settings in the other Functions always return whatever value software programmed for each, but otherwise are ignored by the component. …	RW

Modify Section 7.8.6. as shown

7.8.6. Link Capabilities Register (Offset 0Ch)

…

Table 7-14: Link Capabilities Register

Bit Location	Register Description	Attributes
…	**…**	…
18	**Clock Power Management** – For Upstream Ports, a value of 1b in this bit indicates … … For a multi-Function device, each Function indicates its capability independently. Power Management configuration software must only permit reference clock removal if all Functions of the multi-Function device indicate a 1b in this bit. For ARI Devices, all Functions must indicate the same value in this bit. …	RO

Modify Section 7.8.7. as shown

7.8.7. Link Control Register (Offset 10h)

…

Table 7-15: Link Control Register

Bit Location	Register Description	Attributes
1:0	**Active State Power Management (ASPM) Control** – This field controls … … For multi-Function devices (including ARI Devices), it is recommended that software program the same value for this field in all Functions. ~~Only~~ For non-ARI multi-Function devices, only capabilities enabled in all Functions are enabled for the component as a whole. For ARI Devices, ASPM Control is determined solely by the setting in Function 0, regardless of Function 0's D-state. The settings in the other Functions always return whatever value software programmed for each, but otherwise are ignored by the component.	RW
…	**…**	…
6	**Common Clock Configuration** – When Set, this bit indicates that this component and … … For non-ARI multi-Function devices, software must program the same value for this field in all Functions. If not all Functions are Set, then the component must as a whole assume that its reference clock is not common with the upstream component. For ARI Devices, Common Clock Configuration is determined solely by the setting in Function 0. The settings in the other Functions always return whatever value software programmed for each, but otherwise are ignored by the component.	RW
…	**…**	…

Bit Location	Register Description	Attributes
8	**Enable Clock Power Management** – Applicable only for Upstream Ports and with form factors that … … For a non-ARI multi-Function device, power-management-configuration software must only Set this bit if all Functions of the multi-Function device indicate a 1b in the Clock Power Management bit of the Link Capabilities Register. The component is permitted to use the CLKREQ# signal to power manage Link clock only if this bit is Set for all Functions. For ARI Devices, Clock Power Management is enabled solely by the setting in Function 0. The settings in the other Functions always return whatever value software programmed for each, but otherwise are ignored by the component.	RW

Insert following Table 7-15

IMPLEMENTATION NOTE

Software Compatibility With ARI Devices

With the ASPM Control field, Common Clock Configuration bit, and Enable Clock Power Management bit in the Link Control register, there are potential software compatibility issues with ARI Devices since these controls operate strictly off the settings in Function 0 instead of the settings in all Functions.

With compliant software, there should be no issues with the Common Clock Configuration bit, since software is required to set this bit the same in all Functions.

With the Enable Clock Power Management bit, there should be no compatibility issues with software that sets this bit the same in all Functions. However, if software does not set this bit the same in all Functions, and relies on each Function having the ability to prevent Clock Power Management from being enabled, such software may have compatibility issues with ARI Devices.

With the ASPM Control field, there should be no compatibility issues with software that sets this bit the same in all Functions. However, if software does not set this bit the same in all Functions, and relies on each Function in D0 state having the ability to prevent ASPM from being enabled, such software may have compatibility issues with ARI Devices.

Modify Section 7.8.15. as shown

7.8.15. Device Capabilities 2 Register (Offset 24h)

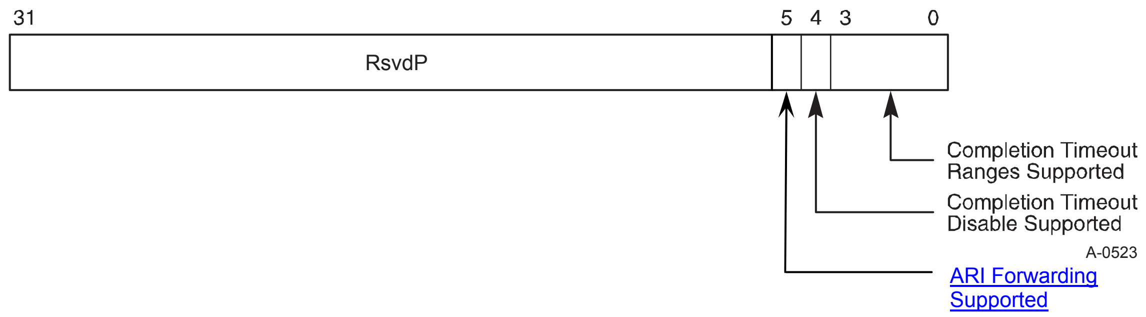
31
5
4
3
0
RsvdP
Completion Timeout Ranges Supported
Completion Timeout Disable Supported
A-0523
ARI Forwarding Supported

Figure 7-24 Device Capabilities 2 Register

Table 7-23 Device Capabilities 2 Register

Bit Location	Register Description	Attributes
…	**…**	…
5	**ARI Forwarding Supported –** Applicable only to Switch Downstream Ports and Root Ports; must be 0b for other Function types. This bit must be set to 1b if a Switch Downstream Port or Root Port supports this optional capability. See Section 6.14. for additional details.	RO

Modify Section 7.8.16. as shown

7.8.16. Device Control 2 Register (Offset 28h)

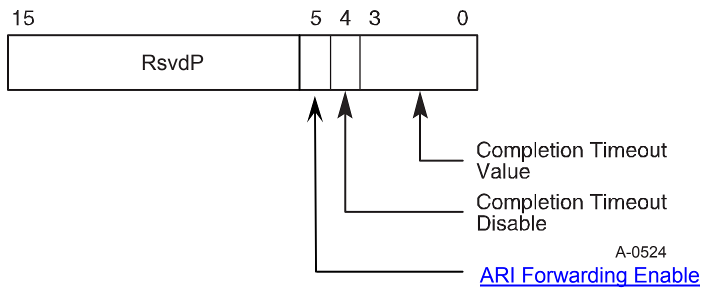

Figure 7-25 Device Control 2 Register

Table 7-24 Device Control 2 Register

Bit Location	Register Description	Attributes
…	**…**	…
5	**ARI Forwarding Enable –** When set, the Downstream Port disables its traditional Device Number field being 0 enforcement when turning a Type 1 Configuration Request into a Type 0 Configuration Request, permitting access to Extended Functions in an ARI Device immediately below the Port. See Section 6.14. Default value of this bit is 0b. Must be hardwired to 0b if the ARI Forwarding Supported bit is 0b.	RW

Modify Section 7.16.4. as shown

7.16.4. Egress Control Vector (Offset 08h)

…

For Root Ports and Switch Downstream Ports, each bit in the bit-array always corresponds to a Port Number. Otherwise, for Functions[14] within a multi-Function device, each bit in the bit-array corresponds to ~~a~~ one or more Function Numbers, or a Function Group Number. For example, access to Function 2 is controlled by bit number 2 in the bit-array. For both Port Number cases and Function Number cases, the bit corresponding to the Function that implements this Extended Capability structure must be hardwired to 0b[15].

If an ARI Device implements ACS Function Groups, its Egress Control Vector Size is required to be a power-of-2 between 8 and 256, and all of its implemented Egress Control Vector bits must be R/W. With ARI Devices, multiple Functions can be associated with a

[14] Including Switch Upstream Ports.

[15] For ARI Devices, the bit must be R/W. See subsequent description.

single bit, so for each Function, its associated bit determines how Requests from it targeting other Functions (if any) associated with the same bit are handled.

If ACS Function Groups are enabled in an ARI Device, the first 8 Egress Control Vector bits in each Function are associated with Function Group Numbers instead of Function Numbers. In this case, access control is enforced between Function Groups instead of Functions, and any implemented Egress Control Vector bits beyond the first 8 are unused.

Independent of whether an ARI Device implements ACS Function Groups, its Egress Control Vector Size is not required to cover the entire Function Number range of all Functions implemented by the Device. If ACS Function Groups are not enabled, Function Numbers are mapped to implemented Egress Control Vector bits by taking the modulo of the Egress Control Vector Size, which is constrained to be a power-of-2.

…

Table 7-66: Egress Control Vector

Bit Location	Register Description	Attributes
N-1:0	**Egress Control Vector** – An N-bit bit-array configured by software. When a given bit is set, peer-to-peer Requests targeting the associated Port, Function, or Function Group are blocked or redirected (if enabled) (see Section 6.13.3). Default value of this field is 0.	RW

Modify Section 7.18.10. as shown

7.18.10. Function Arbitration Table

The Function Arbitration Table register in the MFVC capability structure takes the same form as the Port Arbitration Table register in the VC capability structure (see Section 7.11.10).

The Function Arbitration Table register is a read-write register array that is used to store the WRR or time-based WRR arbitration table for Function Arbitration for the VC resource. It is only present when one or more asserted bits in the Function Arbitration Capability field indicate that the multi-Function device supports a Function Arbitration scheme that uses a programmable arbitration table. Furthermore, it is only valid when one of the above mentioned bits in the Function Arbitration Capability field is selected by the Function Arbitration Select field.

The Function Arbitration Table represents one Function arbitration period. Each table entry containing a Function Number or Function Group[16] Number corresponds to a phase within a Function Arbitration period. The table entry size requirements are as follows:

- The table entry size for non-ARI devices must support enough values to specify all implemented Functions plus at least one value that does not correspond to an

[16] If an ARI Device supports MFVC Function Groups capability and ARI-aware software enables it, arbitration is based on Function Groups instead of Functions. See Section 7.23.

implemented Function. For example, a table with 2-bit entries can be used by a multi-Function device with up to three Functions.

- ❑ The table entry size for ARI Devices must be either 4 bits or 8 bits.
    - ♦ If MFVC Function Groups are enabled, each entry maps to a single Function Group. Arbitration between multiple Functions within a Function Group is implementation specific, but must guarantee forward progress.
    - ♦ If MFVC Function Groups are not enabled and 4-bit entries are implemented, a given entry maps to all Functions whose Function Number modulo 8 matches its value. Similarly, if 8-bit entries are implemented, a given entry maps to all Functions whose Function Number modulo 128 matches its value. If a given entry maps to multiple Functions, arbitration between those Functions is implementation specific, but must guarantee forward progress.

A Function Number or Function Group Number written to a table entry indicates that the phase within the Function Arbitration period is assigned to the selected Function or Function Group (the Function Number or Function Group Number must be a valid one).

- ❑ When the WRR Function Arbitration is used for a VC of the Egress Port of the multi-Function device, at each arbitration phase the Function Arbiter serves one transaction from the Function or Function Group indicated by the Function Number or Function Group Number of the current phase. When finished, it immediately advances to the next phase. A phase is skipped, i.e., the Function Arbiter simply moves to the next phase without delay if the Function or Function Group indicated by the phase does not contain any transaction for the VC.
- ❑ When the Time-based WRR Function Arbitration is used for a VC of the Egress Port of the multi-Function device, at each arbitration phase aligning to a virtual timeslot, the Function Arbiter serves one transaction from the Function or Function Group indicated by the Function Number or Function Group Number of the current phase. It advances to the next phase at the next virtual timeslot. A phase indicates an "idle" timeslot, i.e., the Function Arbiter does not serve any transaction during the phase, if
    - the phase contains the Number of a Function or a Function Group that does not exist, or
    - The Function or Function Group indicated by the phase does not contain any transaction for the VC.

The Function Arbitration Table Entry Size field in the Port VC Capability register determines the table entry size. The length of the table is determined by the Function Arbitration Select field as shown in Table 7-76.

When the Function Arbitration Table is used by the default Function Arbitration for the default VC, the default values for the table entries must contain at least one entry for each of active Functions or Function Groups in the multi-Function device to ensure forward progress for the default VC for the multi-Function device's Upstream Port. The table may contain RR or RR-like fair Function Arbitration for the default VC.

Add new Section 7.23.

7.23. ARI Capability

ARI is an optional capability. This capability must be implemented by each Function in an ARI Device. It is not applicable to a Root Port, a Switch Downstream Port, a Root Complex Integrated Endpoint, or a Root Complex Event Collector.

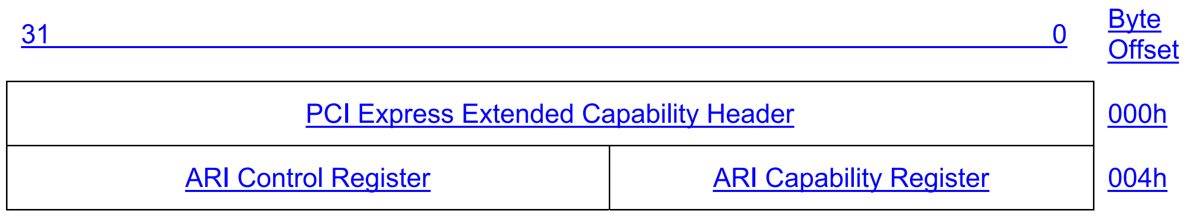

Figure 7-xx ARI Capability

7.23.1. ARI Capability Header (Offset 00h)

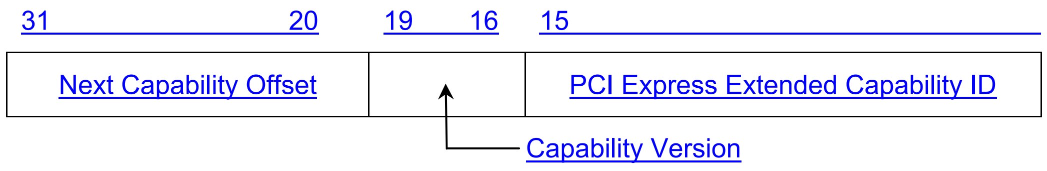

Figure 7-xx ARI Capability Header

Table 7-xx ARI Capability Header

Bit Location	**Register Description**	**Attributes**
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the extended capability. PCI Express Extended Capability ID for the ARI Capability is 000Eh.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of capabilities.	RO

7.23.2. ARI Capability Register (Offset 04h)

15 8	7 2	1	0
Next Function Number	RsvdP	A	M

Figure 7-xx ARI Capability Register

Table 7-xx ARI Capability Register

Bit Location	**Register Description**	**Attributes**
0	**MFVC Function Groups Capability (M) –** Applicable only for Function 0; must be 0b for all other Functions. If 1b, indicates that the ARI Device supports Function Group level arbitration via its Multi-Function Virtual Channel (MFVC) Capability structure.	RO
1	**ACS Function Groups Capability (A) –** Applicable only for Function 0; must be 0b for all other Functions. If 1b, indicates that the ARI Device supports Function Group level granularity for ACS P2P Egress Control via its ACS Capability structures.	RO
15:8	**Next Function Number –** This field indicates the Function Number of the next higher numbered Function in the Device, or 00h if there are no higher numbered Functions. Function 0 starts this linked list of Functions.	RO

7.23.3. ARI Control Register (Offset 06h)

15 7	6 4	3 2	1	0
RsvdP	Function Group	RsvdP	A	M

Figure 7-xx ARI Control Register

Table 7-xx ARI Control Register

Bit Location	Register Description	Attributes
0	**MFVC Function Groups Enable (M) –** Applicable only for Function 0; must be hardwired to 0b for all other Functions. When set, the ARI Device must interpret entries in its Function Arbitration Table as Function Group Numbers rather than Function Numbers. Default value of this field is 0b. Must be hardwired to 0b if the MFVC Function Groups Capability bit is 0b.	RW
1	**ACS Function Groups Enable (A) –** Applicable only for Function 0; must be hardwired to 0b for all other Functions. When set, each Function in the ARI Device must associate bits within its Egress Control Vector with Function Group Numbers rather than Function Numbers. Default value of this field is 0b. Must be hardwired to 0b if the ACS Function Groups Capability bit is 0b.	RW
6:4	**Function Group –** Assigns a Function Group Number to this Function. Default value of this field is 0b. Must be hardwired to 0b if in Function 0, the MFVC Function Groups Capability bit and ACS Function Groups Capability bit are both 0b.	RW