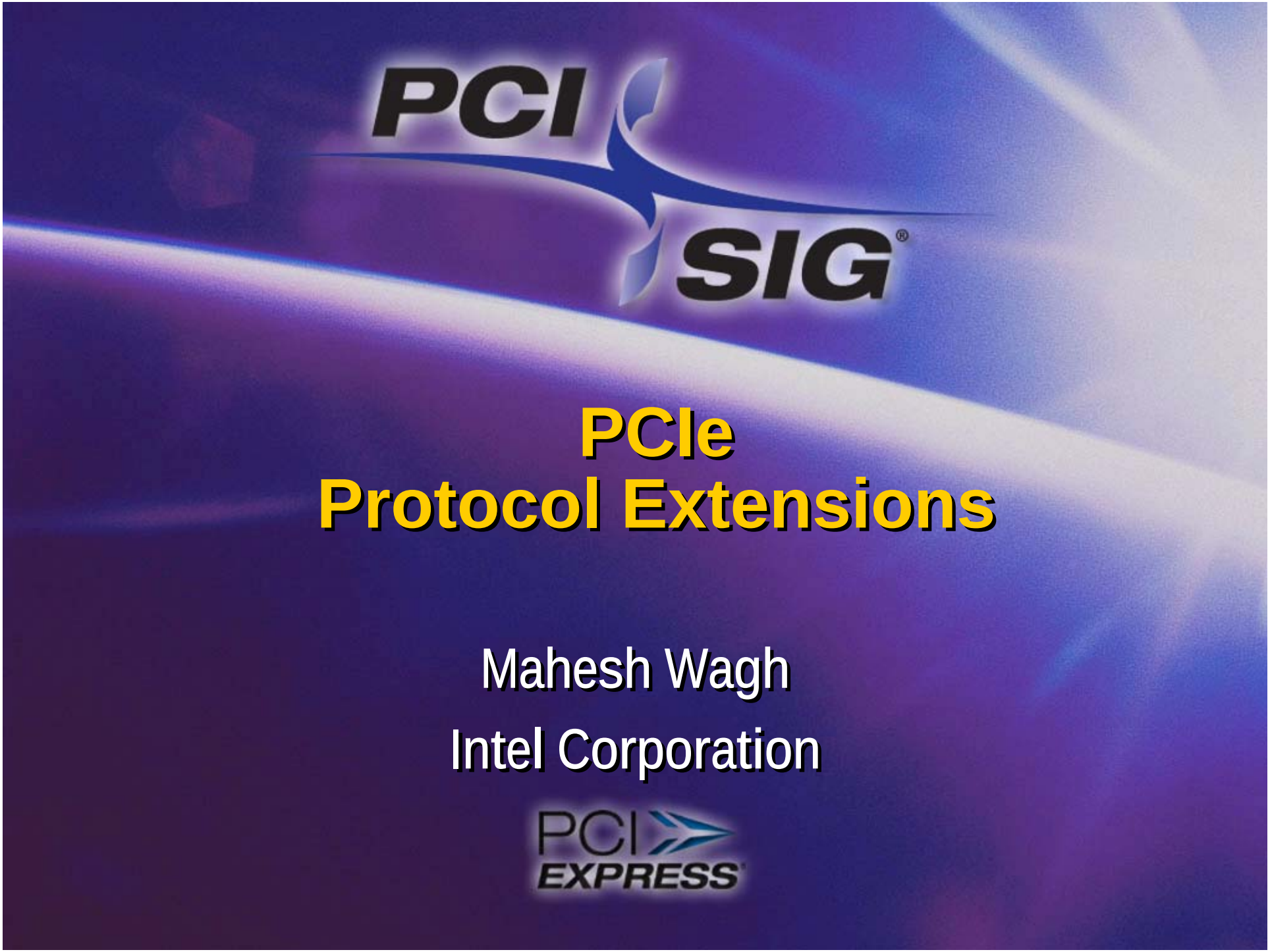
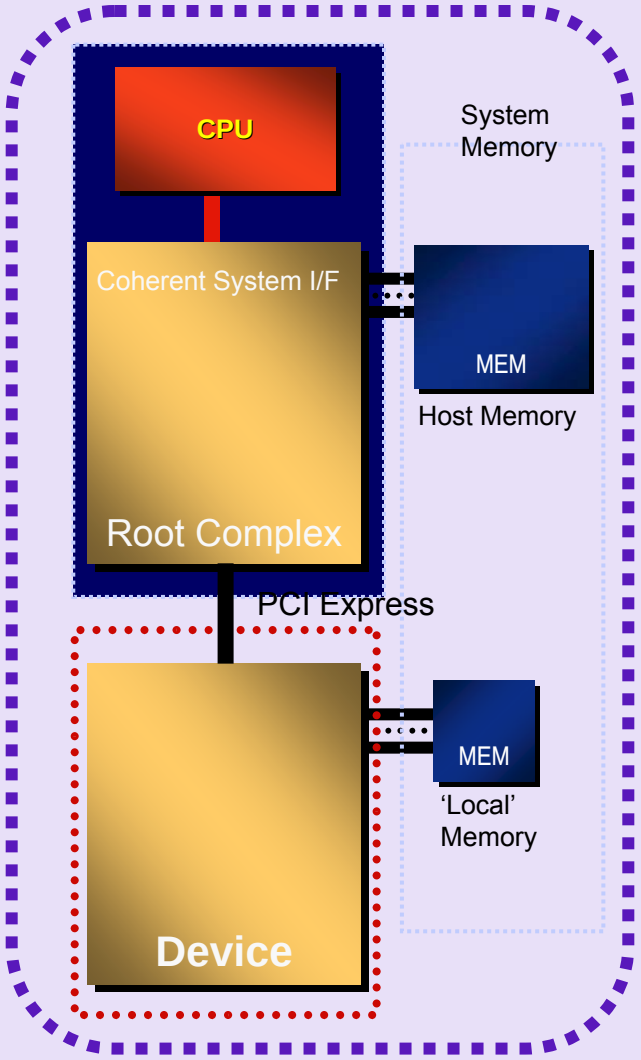

PCI
SIG
PCIe
Protocol Extensions
Mahesh Wagh
Intel Corporation
PCI
EXPRESS

Objective

- PCIe protocol working group is actively developing protocol extension ECRs to PCIe 2.0 specification
  - ✓ ECRs available for PCI SIG membership review in Q1-08
  - ✓ Plan is to roll ECRs in to PCIe 3.0 specification
- Objective of this training material is to
  - ✓ Provide an overview of protocol extensions currently under development
  - ✓ Obtain an early feedback

PCIe Protocol Extensions



Latency Reduction

- ✓ Data Re-use Hints – Mechanism for efficient and lower latency access
- ✓ Ordering – Transaction level attribute/hint to optimize ordering within RC and memory subsystem

Software Model Improvements

- ✓ Atomic Operations – Mechanisms to reduce synchronization overhead
- ✓ IO Page Faults– Mechanism to notify device of page faults and request for faulted pages to be made available

Communication Model Enhancements

- ✓ Multicast – Mechanism to transfer common data or command set from one source to multiple recipients

Power Management

- ✓ Dynamic Power Allocation - Support for dynamic power operational modes through standard configuration mechanism

Configuration Enhancements

- ✓ Resizable BAR– Mechanism for device to re-negotiate allocated BAR sizes
- ✓ Internal Error Reporting– Extend the AER to report component internal correctable/uncorrectable errors and report multiple errors


PCI
SIG
Data Re-Use Hint
(DRH)
PCI
EXPRESS

DRH Motivation

- Today's PCIe requests are coherent with respect to system memory/caches
  - ✓ Root Complex maintains coherency via "snoop"
- Current communication between device and host does not take full advantage of system capabilities
  - ✓ Use of system cache hierarchy can help reduce access latency
  - ✓ Effective use of system resources (system interconnect and memory)
- Data Re-Use hints provide an opportunity to optimize use of system fabric and improve system efficiency
  - ✓ Facilitate Data Residency/Allocation within system cache hierarchy
  - ✓ Minimize memory access latencies
  - ✓ Reduce memory & System interconnect Bandwidth & associated Power consumption

DRH - Usage Models

Data Structures of interest

- Control Data i.e. Descriptors
- Headers for protocol processing
- Payload for data copies

Usage Models

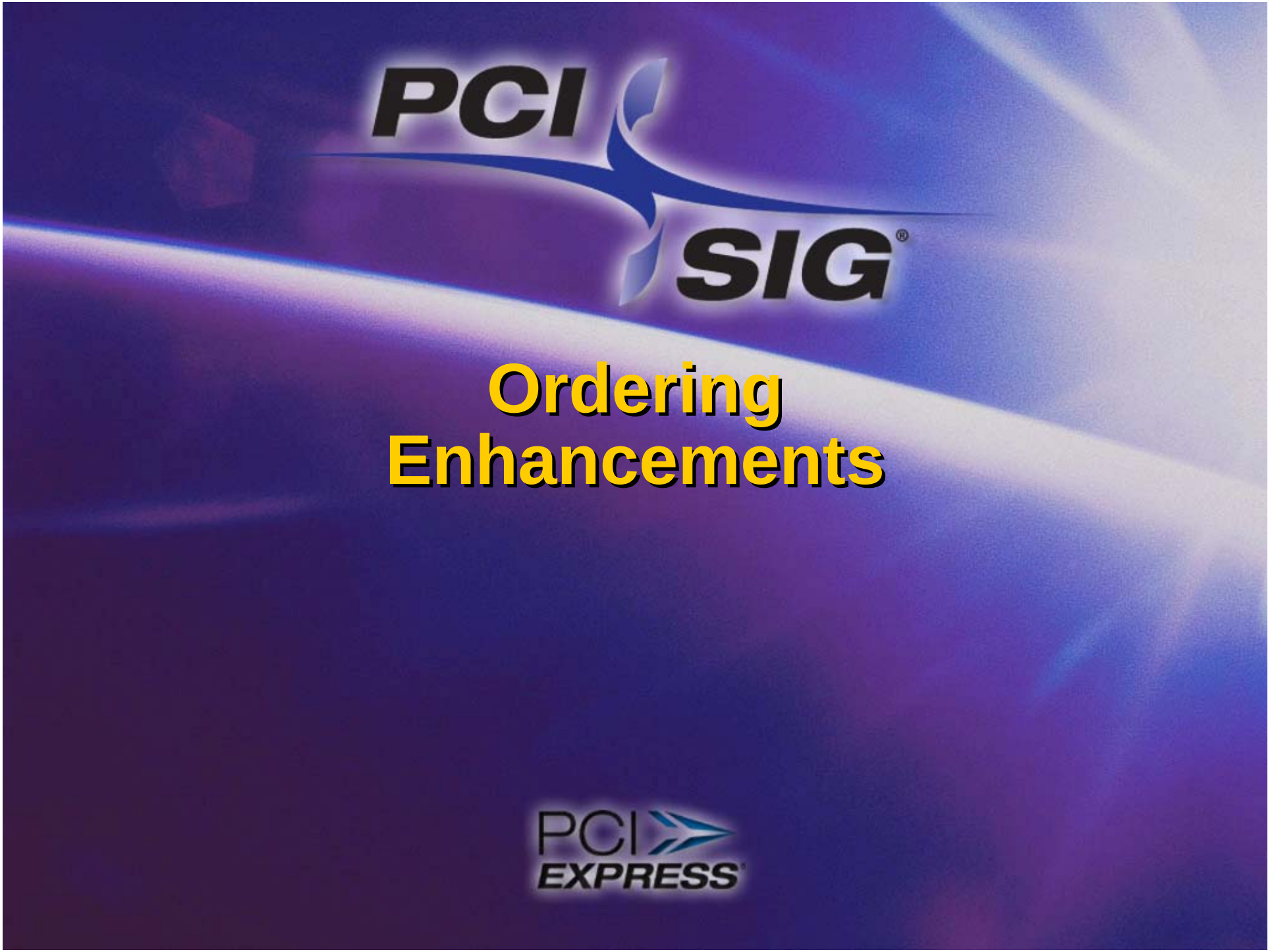
- Efficient processing of network IO in systems with variable access latencies
  - ✓ IO is increasingly becoming faster (E.g. 10G → 40G → 100G …)
- Communications adapters in HPC clusters, Database System Clusters & Computational Accelerators
  - ✓ Multi-node barriers and collective operations are often a key performance limit.
  - ✓ Particularly exchange of lock information; often key barrier to scaling
  - ✓ Delay / overhead per operation limits efficiency in some operations (not all), reducing range of applicability.

DRH - Mechanism

- Mechanism to provide processing hints on per transaction basis for requests that target memory space
  - ✓ Enable system hardware (ex: Root-Complex) to optimize on a per transaction basis
  - ✓ Applicable to Memory Read/Write and Atomic Operations
- Transaction Processing hints are opaque and implementation specific; used to provide hints for transaction processing
  - ✓ Temporal Re-Use
  - ✓ Enables association of host processing elements with processing of requests from specific devices
  - ✓ Location/Level of a targeted cache (host processor core identifier)
  - ✓ Allocation hints
  - ✓ Not limited to examples provided above
- Values may be retrieved via a central software entity e.g. a system level device driver would provide the value that can be used to target a particular cache hierarchy

DRH Summary

- Mechanism to make effective use of system fabric and improve system efficiency
  - ✓ Reduce variability in access to system memory
  - ✓ Reduce Memory & System Interconnect BW & Associated Power Consumption
- Minimal or No Ecosystem Impact
  - ✓ No change to existing software models, however software support may be required to retrieve hints from system hardware
  - ✓ Endpoints take advantage only as needed → No cost if not used
  - ✓ Root Complex allowed implementation tradeoffs
  - ✓ Minimal impact to Switches
- Architected software discovery, identification and control of capabilities
  - ✓ RC support for processing hints
  - ✓ Endpoint enabling to issue hints


PCI
SIG
Ordering
Enhancements
PCI
EXPRESS

Review: PCIe Ordering Rules

"No" entries caused by Producer/ Consumer restrictions – Potential Targets

Table 2-24: Ordering Rules Summary Table

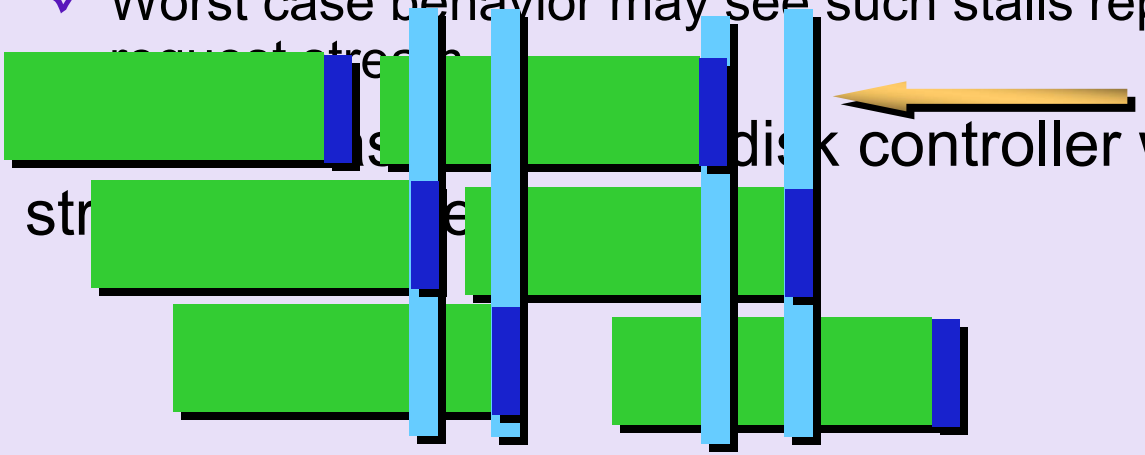
Row Pass Column?		Posted Request	Non-Posted Request		Completion	
		Memory Write or Message Request (Col 2)	Read Request (Col 3)	I/O or Configuration Write Request (Col 4)	Read Completion (Col 5)	I/O or Configuration Write Completion (Col 6)
Posted Request	or Message Request (Row A)	a) No b) Y/N	Yes	Yes	a) Y/N b) Yes	a) Y/N b) Yes
Non-Posted Request	(Row B)	No	Y/N	Y/N	Y/N	Y/N
	Configuration Write Request (Row C)	No	Y/N	Y/N	Y/N	Y/N
Completion	Completion (Row D)	a) No b)Y/N	Yes	Yes	a) Y/N b) No	Y/N
	I/O or Configuration Write Completion (Row E)	Y/N	Yes	Yes	Y/N	Y/N

"Yes" entries are required for deadlock avoidance – Not candidates for change

- Maximum theoretical flexibility: All entries are "Y/N"
- "Yes" entries required for deadlock avoidance
- Relaxed Ordering (RO) modifies A2 & D2

Motivation

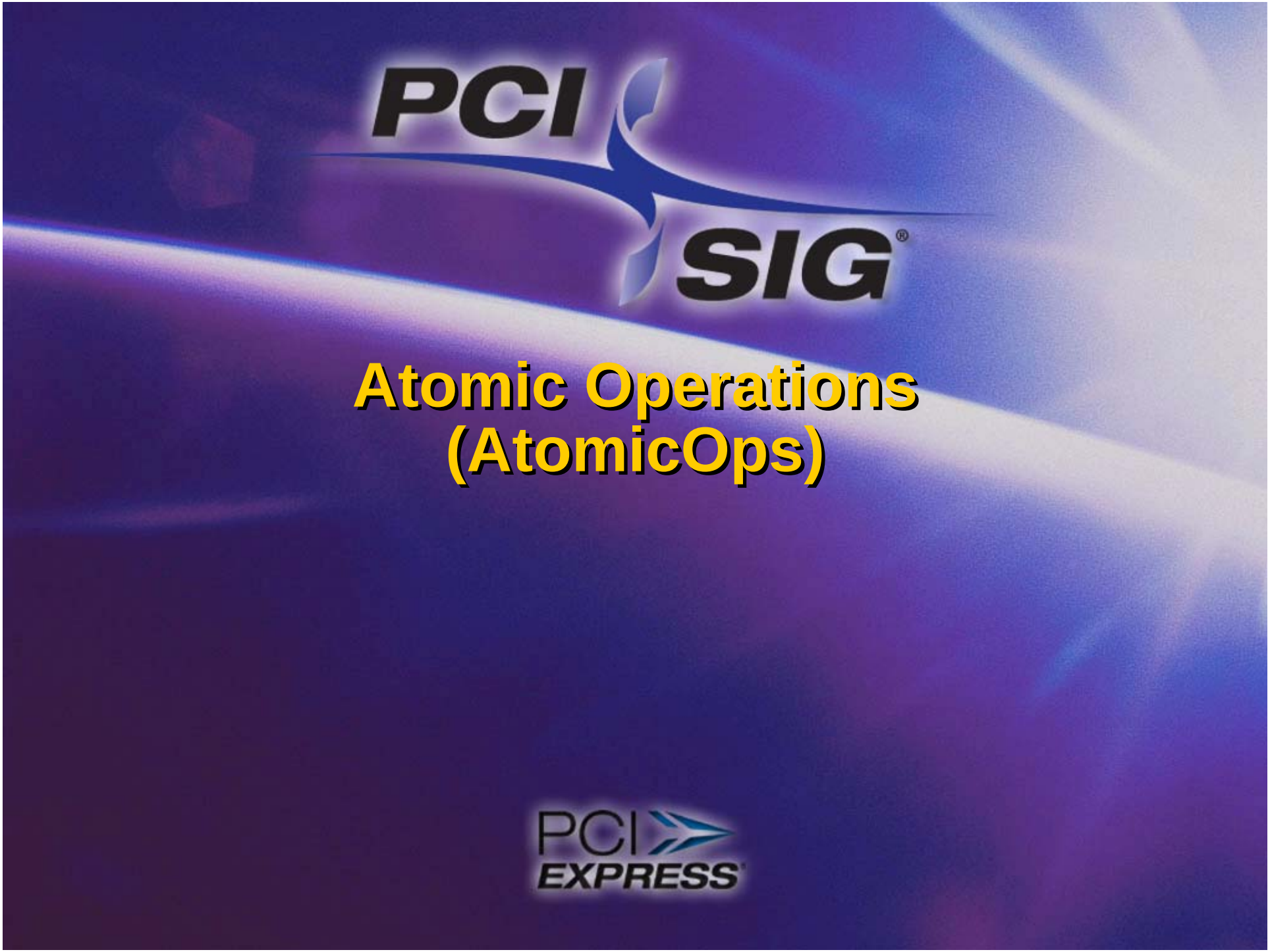
- RO works well for models where a data buffer is written once, then consumed, then recycled
  - ✓ Not OK for buffers that will be written more than once because writes are not guaranteed to complete in order issued
  - ✓ In effect, RO is “too relaxed”
- Conventional Ordering (CO) can cause significant stalls
  - ✓ Observed stalls in the 10’s to 100’s of ns are seen
  - ✓ Worst case behavior may see such stalls repeatedly for a request stream
- Consider a disk controller with multiple streams



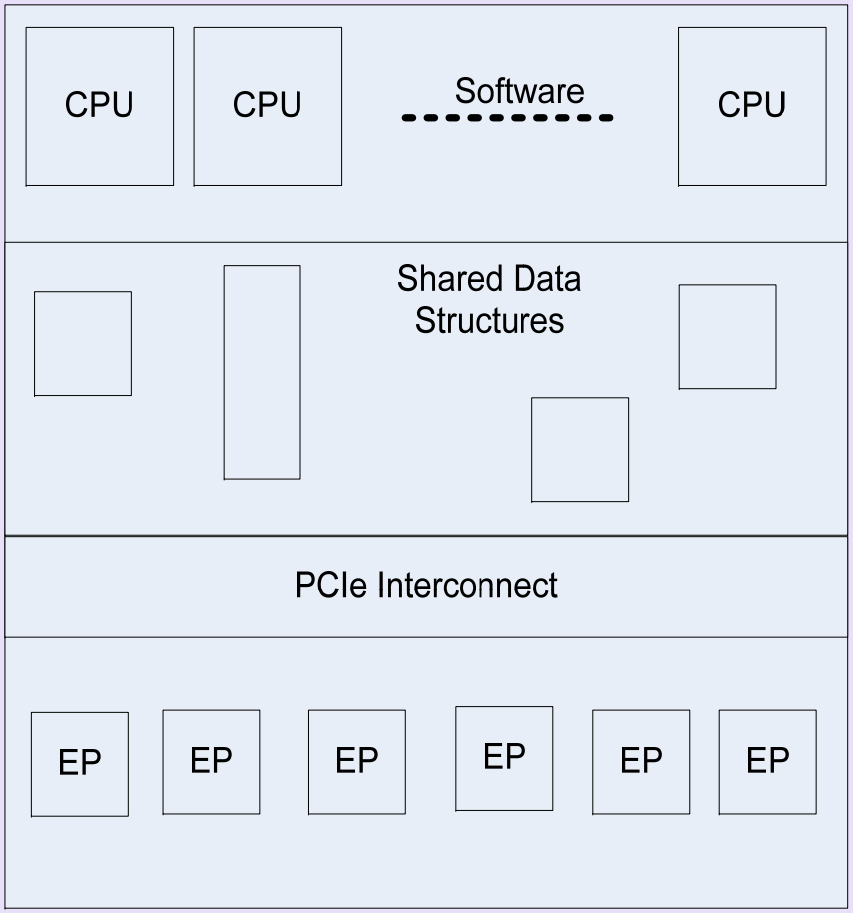
Ordering Relaxations provide large potential benefits

General Requirements for Ordering Rule improvements

- Optional – Ease of implementation & Validation
  - ✓ Ordering model must provide high ROI
  - ✓ Universally useful and accepted
  - ✓ Enabled by software through specified config register i/f
- "Hint" for Root Complex
  - ✓ Comprehend if support in other system elements is required
- Minimize software impact
  - ✓ Enabled through new end-point hardware w/o requiring modified software
- Improve performance
  - ✓ Example: Avoid needless ordering between unrelated traffic streams
  - ✓ B2 and D2 No become Y/N when Req IDs are different; optionally for same ReqID B2 becomes Y/N if read and write addresses are different
- Address RO usability problem for writes to same address
  - ✓ Writes by same requester to same address must complete in-order
- Eliminate at least the "No" for A2
  - ✓ Preferably eliminate all the column 2 "No" entries
- Only one new semantic
  - ✓ Optimal solution that gets the best ROI – minimize cost, learning curve, etc.


PCI
SIG
Atomic Operations
(AtomicOps)
PCI
EXPRESS

Motivation



- Atomic transaction support for Host update of main memory exists today
  - ✓ Useful for synchronization without interrupts
  - ✓ Rich library of proven algorithms in this area
- Benefit in extending existing inter-processor primitives for data sharing/synchronization to PCIe interconnect domain
  - ✓ Low overhead critical sections
  - ✓ Non-Blocking mechanisms for managing data structures e.g. Task lists
  - ✓ Lock-Free Statistics e.g. counter updates
- Existing application performance can generally be improved by providing synchronization enhancements
  - ✓ Faster packet arrival rates create demand for faster synchronization
- Emerging applications require PCIe synchronization enhancements to support multiple producer – multiple consumer co-ordination
  - ✓ Math, Visualization, Content Processing etc

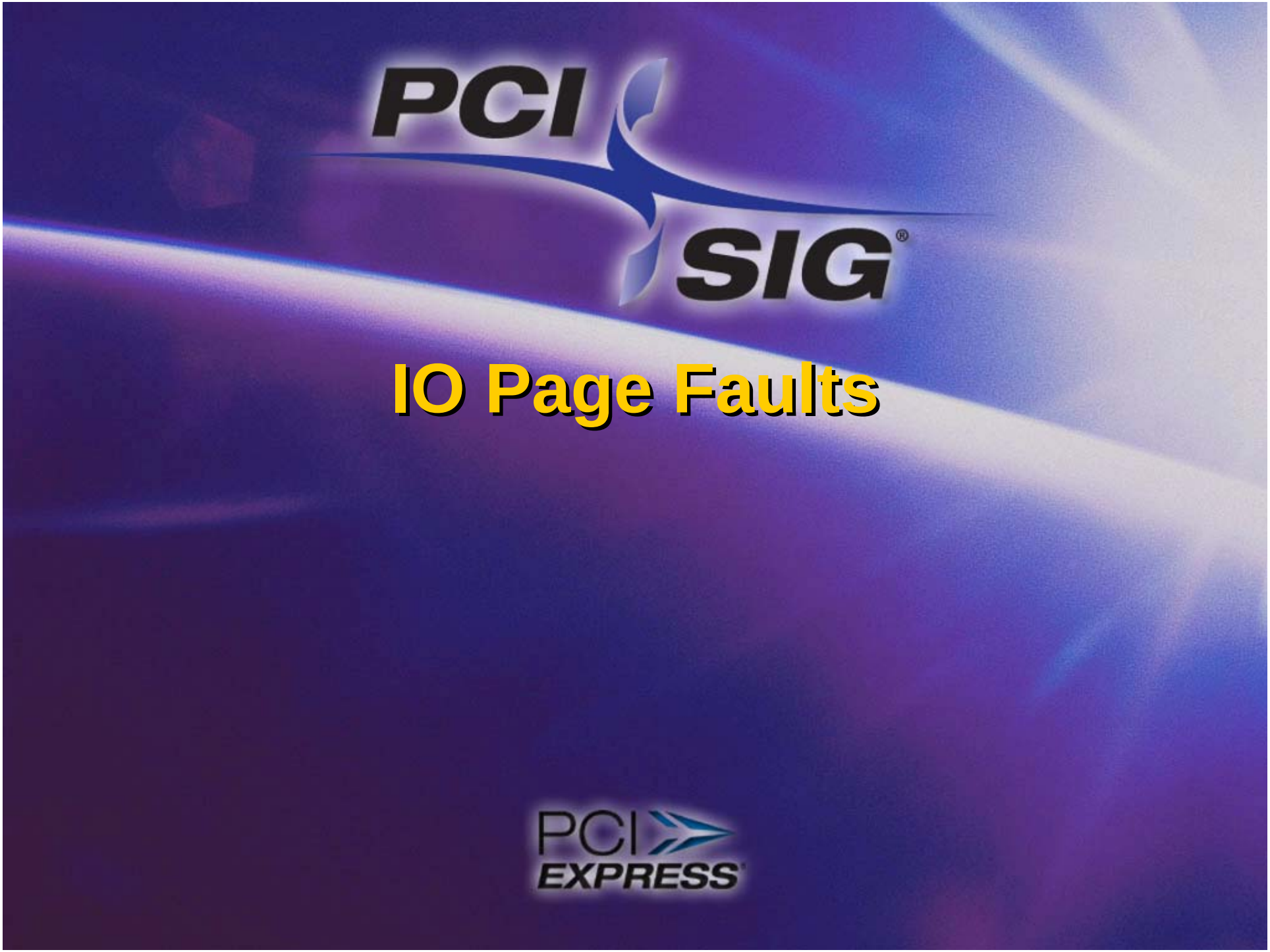
Atomic Operations

Atomic Operations	Description
FetchAdd	Data
Swap	Data(Addr) = SwapData
CAS (Compare & Swap)	Data(Addr)), Data(Addr) = SwapData

- Operation sizes supported
  - ✓ 32b and 64b operation sizes for FetchAdd and Swap
  - ✓ 32b, 64b and 128b operation sizes for CAS
- Atomic operation request address must be aligned to natural operation size
  - ✓ Atomic requests guaranteed to not cross cache line boundaries

Atomic Operations Summary

- Atomic Operations provide capability symmetric to and consistent with capabilities supported by most host CPU architectures
  - ✓ Support core synchronization mechanisms (e.g. semaphores) for CPU/device (as for CPU/CPU), without use of bus-lock type mechanism and minimal use of critical sections
- Enable reuse of existing/debugged algorithms for critical sections, data sharing and queuing implementations
  - ✓ Device side re-use of algorithms already proven on hosts
- Atomic Operations supported: compare&swap, swap and Fetchadd
  - ✓ Compare-and-Swap: Essential for implementation of critical sections and non-blocking queuing algorithms
  - ✓ Swap: Sample&Reset
  - ✓ FetchAdd: Lock-Free Statistics
- A new AtomicOp Extended Capability structure to enable software discovery of AtomicOperation capabilities supported by functions
  - ✓ New AtomicOp Routing support bit in device capabilities register 2 enables software to discover atomic routing capability for routing elements


PCI
SIG
IO Page Faults
PCI
EXPRESS

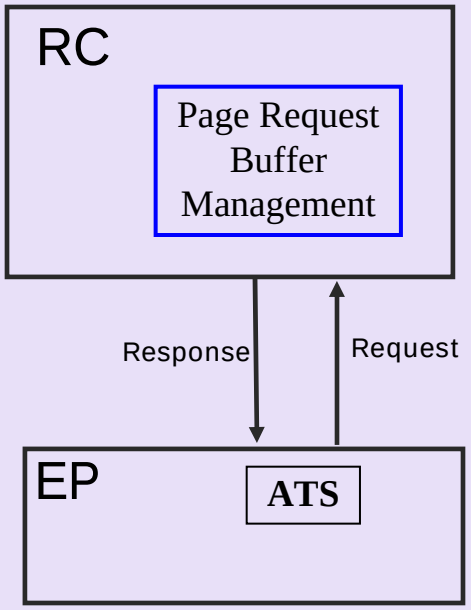
Motivation

- RC Virtualization encourages use of Guest Virtual Addresses by I/O devices
  - ✓ Virtualization increases pressure on memory resources making pinning increasingly expensive
  - ✓ Page fault not desired outcome, but preventing page faults from occurring may be more difficult than building model for supporting the rare occurrence of a page fault
- Acceleration model benefit from user-level (process) virtual address based I/O communication
  - ✓ Maintaining synchronization between “known” addresses in co-processing systems and resident pages of process memory is problematic if pinning is only available model
  - ✓ Again, memory becoming scarce commodity in virtualized systems and minimization of static process footprint is important
- Hypervisor resource over-commitment important to maintaining overall system performance
  - ✓ Balloon allocators grow and shrink guest physical memory allocations
  - ✓ Swapping of System Image (SI) to disk possible, but without page faulting, only pages not potentially used by I/O can be swapped
    - – Difficult to know for fully virtualized guests.
    - – Become more acute with IOV direct device attachment
  - ✓ Page faults result in better ability to over-commit memory
  - ✓ Yet another instance of desire to minimize static memory footprint in favor of more dynamic footprint

IO Page Fault Mechanism

Basic IO Page Fault Mechanism as follows

- Recognize
  - ✓ Is a valid translation for a page available.
  - ✓ Check IOTLB.
  - ✓ Request translation via Address Translation Services (ATS).
  - ✓ If translation request fails – Request.
- Request
  - ✓ Use the New Page Request interface to load page to RC
- Restart
  - ✓ When page has been loaded, RC notifies I/O device.
  - ✓ I/O device reinitiates 3 REs (Recognize – Request – Restart)



Page Request Interface

- Optional Normative mechanism enables a device to request residence of a specific page or set of pages into the associated system's memory
  - ✓ Devices required to employ a Translation Agent (TA) and/or Address Translation Cache (ATC) and/or Address Translation Services (ATS)
- Page Request Interface is applicable to RC and components that implement Endpoint functions
  - ✓ Completely transparent to routing elements
- Page Request is a PCIe message request initiated by endpoint functions
  - ✓ Requesting device provides address and tag
  - ✓ Allows for group of page requests into tag groups
- After the Page Request is serviced RC sends a tagged Response message back to endpoint function
  - ✓ Notify requesting function about page request load success or resource oversubscription/buffer failure conditions
- Page request interface mechanism is orthogonal to system virtualization interface
  - ✓ RC Page request buffer management implementation tradeoffs allowed


PCI
SIG
Multicast
PCI
EXPRESS

Motivation

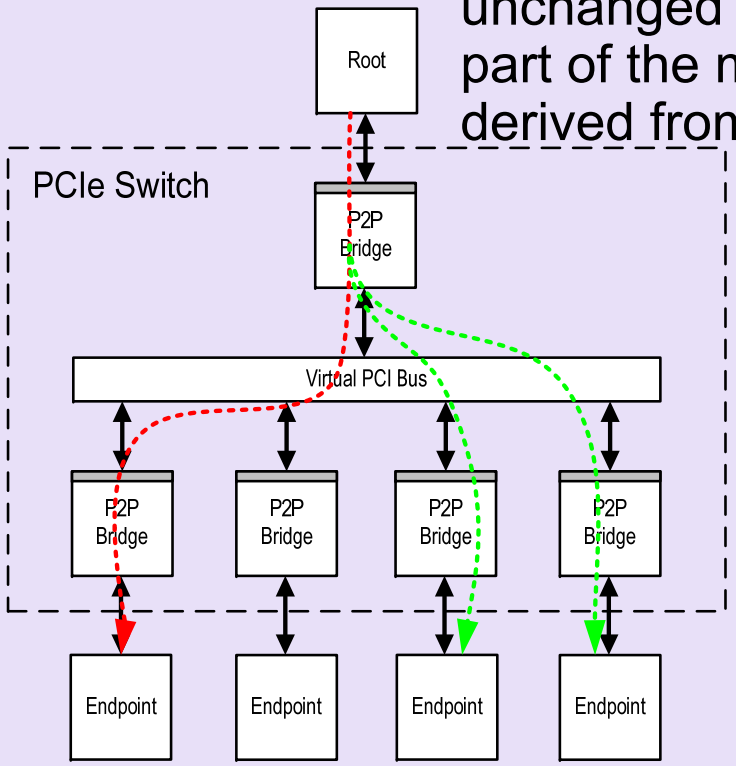
- Many applications that benefit from multicast
  - ✓ Communications (e.g. route table updates, support of IP multicast)
  - ✓ Storage (e.g., mirroring, RAID)
  - ✓ Multi-headed graphics
  - ✓ PCIe used in place of backplane Ethernet
- PCIe architecture extended to support address based multicast
  - ✓ New Multicast BAR to define multicast address space
  - ✓ New Multicast capability to configure RCRB/Switches and endpoints for multicast address decode and routing
- Support only Posted, address routed transactions (e.g., memory writes)
  - ✓ Support both RCs and EPs as both targets and initiators
  - ✓ Compatible with systems employing Address Translation Services (ATS) and Access Control Services (ACS)
  - ✓ Multicast capability permitted at any point in a PCIe hierarchy

Example 1

PCIe Standard Address Route

Multicast Address Route

- Request that hits a multicast address range, is routed unchanged to ports that are part of the multicast group derived from request address

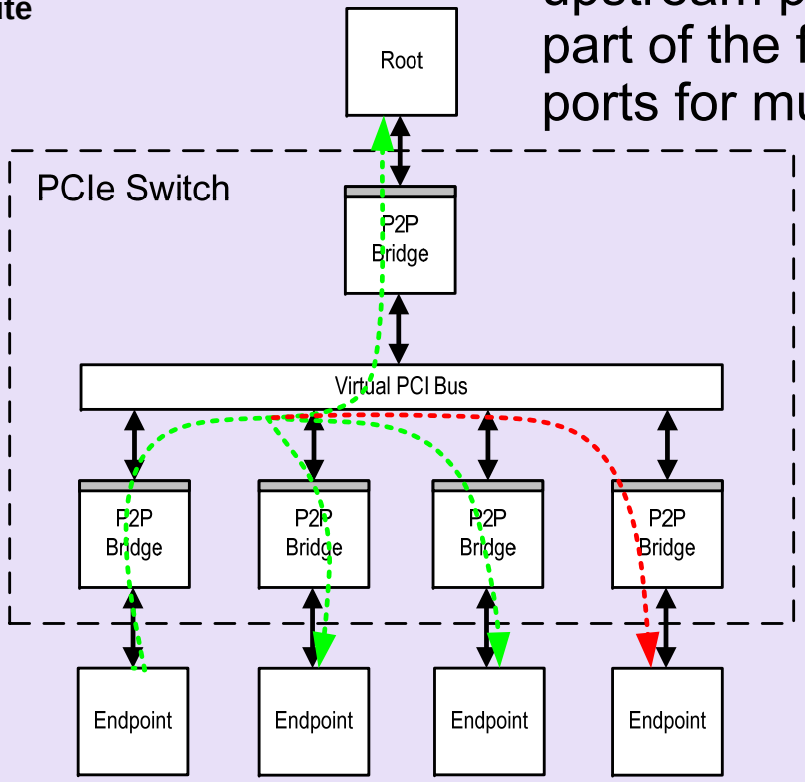


Example 2

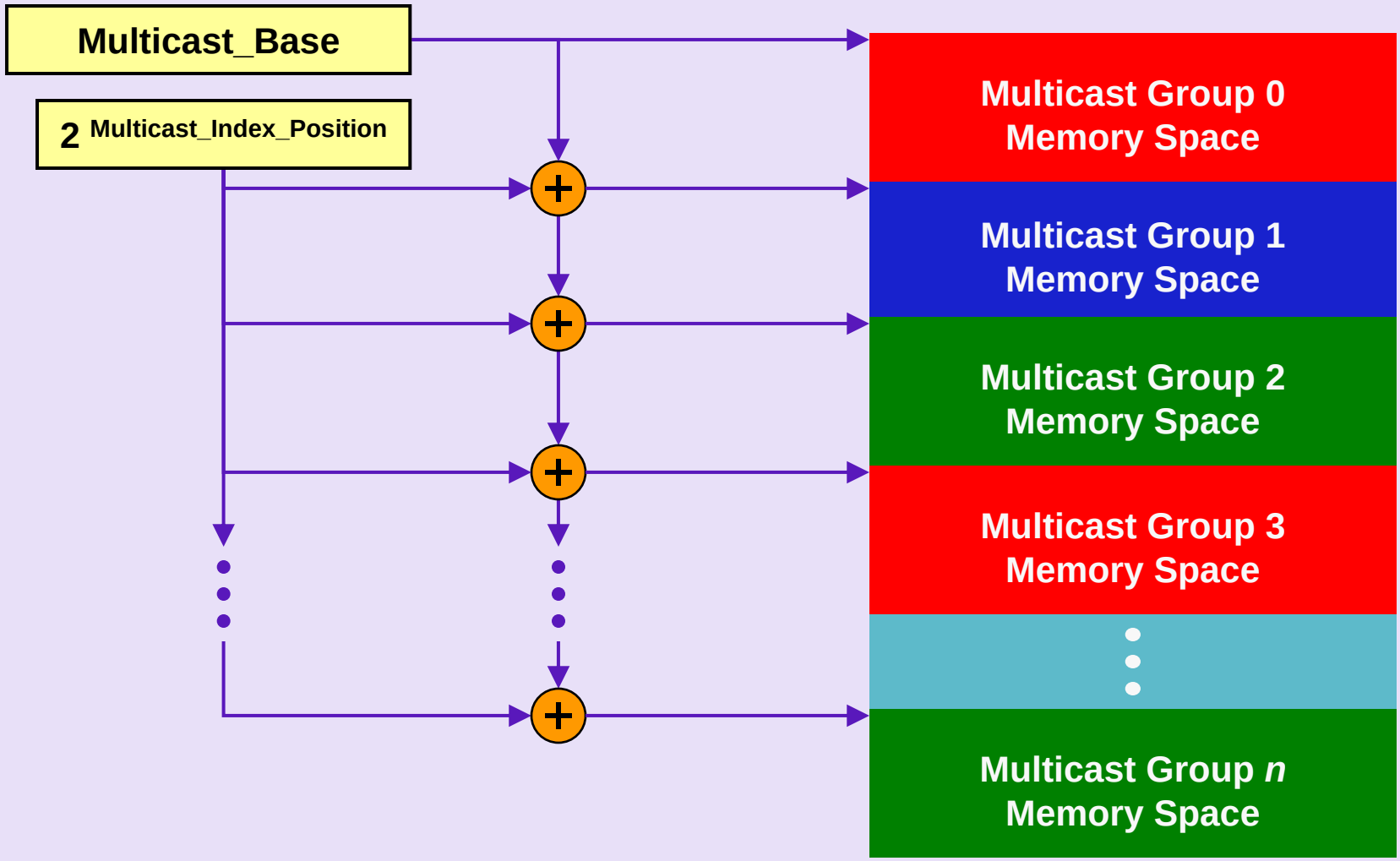
PCIe Standard Address Route

Multicast Address Route

- Address route upstream– upstream port must be part of the forwarding ports for multicast



Multicast Memory Space



Multicast Hit & Processing

- Multicast Hit if:
  - ✓ Multicast_Enable is Set and
  - ✓ TLP is Posted Memory Write and
  - ✓ Targets multicast address range
- If Switch or Root:
  - ✓ Derive the Multicast Group number and look up the associated multicast vector from Multicast capability
  - ✓ Forward TLP to ports indicated in Multicast Vector
    - – Do not forward TLP back out the ingress port – deadlock avoidance
  - ✓ To send TLPs upstream, software must set the appropriate Multicast vector bit
- If Endpoint:
  - ✓ Derive the Multicast Group number, look up multicast capability to determine if the endpoint function is a valid target
  - ✓ If permitted then accept TLP, else indicate UR

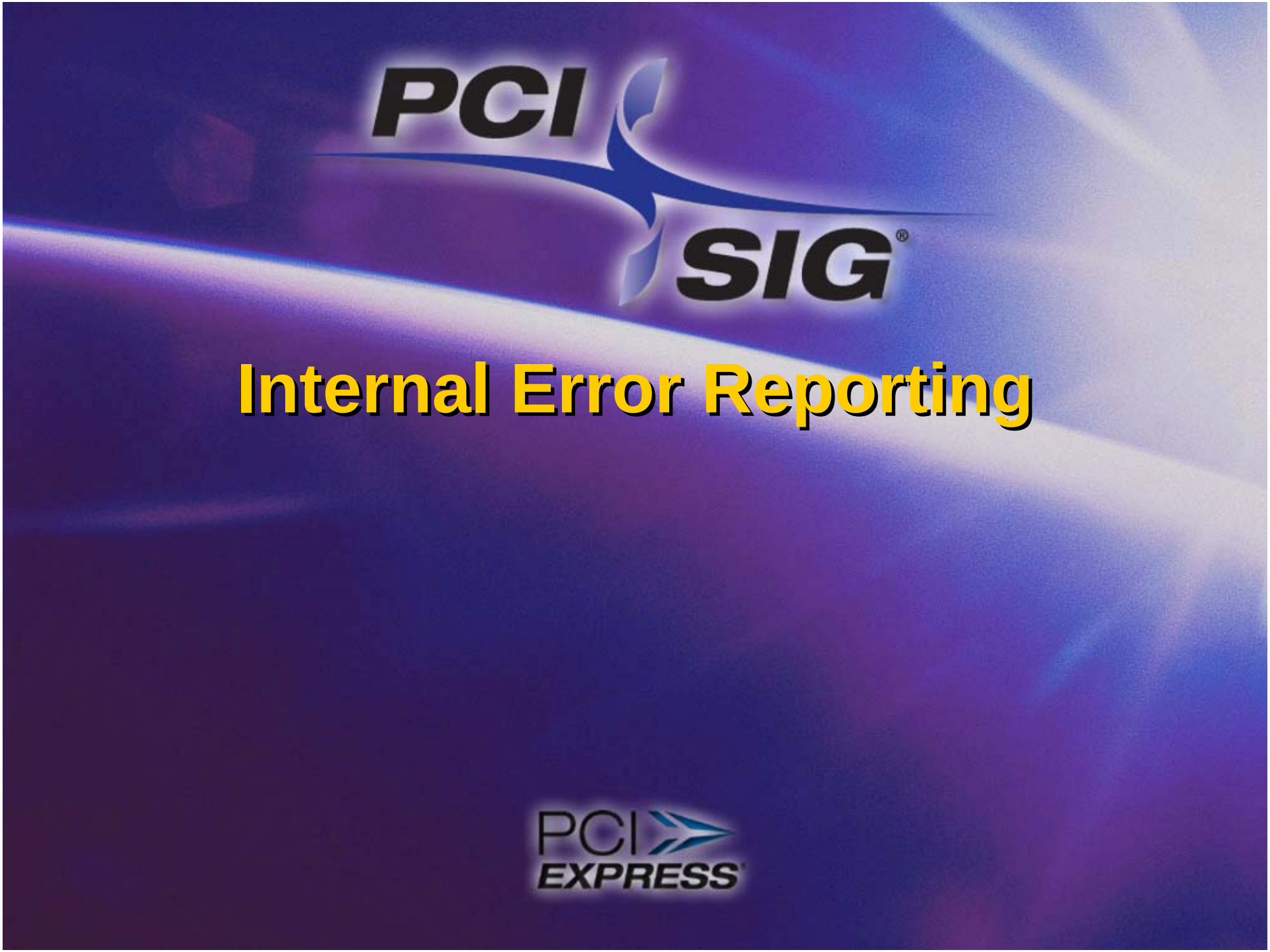

PCI
SIG
Resizable
Base Address Register
(BAR)
PCI
EXPRESS

Motivation

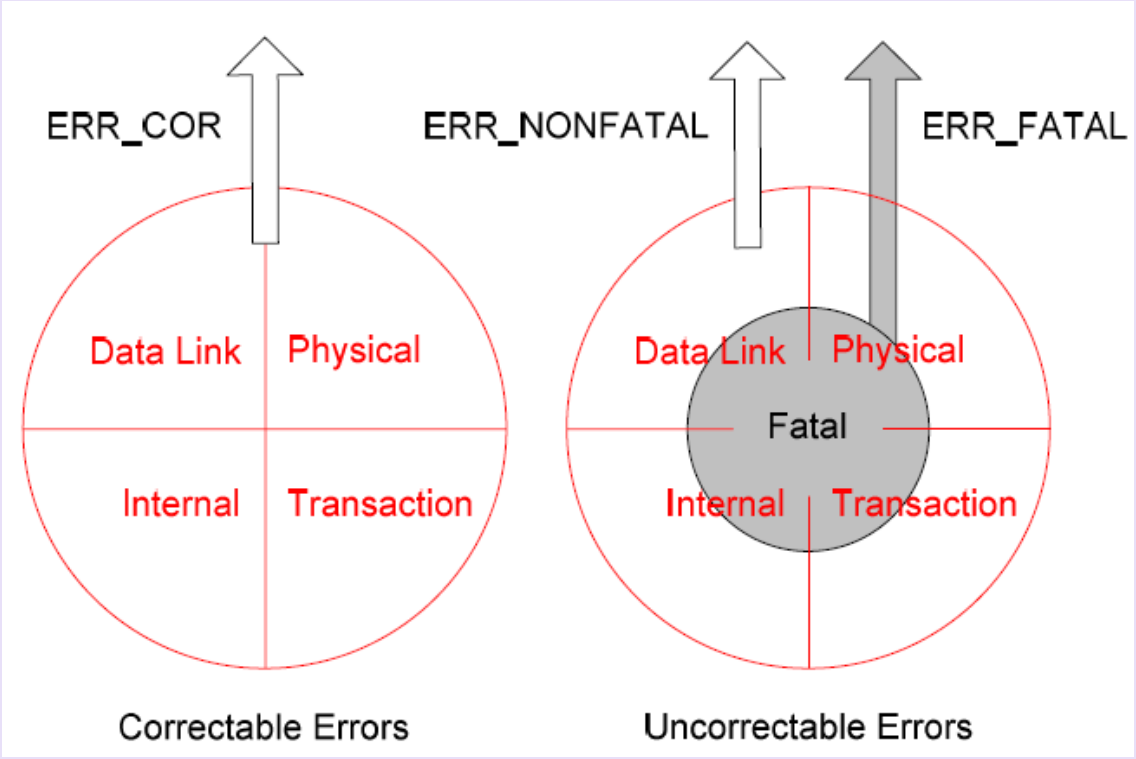
- BAR Allocation
  - ✓ Local memory is becoming quite large on some add in cards
  - ✓ More frequently, all of the memory mapped resources will not be allocated for all of the add in cards
    - – Resource is not allocated or function forced to report smaller aperture size in order to be allocated
  - ✓ 256 MB seems to be a limit in 32 bit systems
  - ✓ Current solutions are proprietary and not exposed to all software
- New mechanism allows the resource to be sized appropriately for the system it is in so that all of the resources can be allocated
- Mechanisms are invoked before BAR allocation during enumeration

Resizable BAR

- New Sizable BAR capability
  - ✓ Endpoint functions advertise various sizes for device memory mapped resource
  - ✓ Control to enable software to program size of device BAR
- Hardware utilizes this new capability to report various sizes for the device BAR to resource allocation software
- Resource allocation software will attempt to program the largest possible size for device BAR
  - ✓ System can be configured with optimal resource settings
- Backwards Compatible
  - ✓ Current resource allocation must work for software that does not comprehend this new capability


PCI
SIG
Internal Error Reporting
PCI
EXPRESS

Motivation



Error Classification

- New Internal Error classification to report internal errors associated with PCIe interface

Definitions

- Internal Error
  - ✓ An error that occurs within a component and that is not attributable to a packet or event on the PCI Express interface
  - ✓ What is reported as an internal error is implementation specific
- Correctable Internal Error
  - ✓ An internal error that hardware has masked or worked around without any loss of information or improper operation.
  - ✓ Example:
    - An ECC corrected single bit error in an internal packet buffer
- Uncorrectable Internal Error
  - ✓ An internal error that has resulted in improper operation but that does not affect proper future operation (i.e., a transient or intermittent fault).
  - ✓ Example:
    - A transient memory fault that results in a double bit error in an internal packet buffer that stores the header of a non-ECRC protected TLP.
- Uncorrectable Persistent Internal Error
  - ✓ An internal error the results in permanent improper operation.
  - ✓ Example:
    - Double bit error in an memory that stores internal control structures

Internal Error Logging

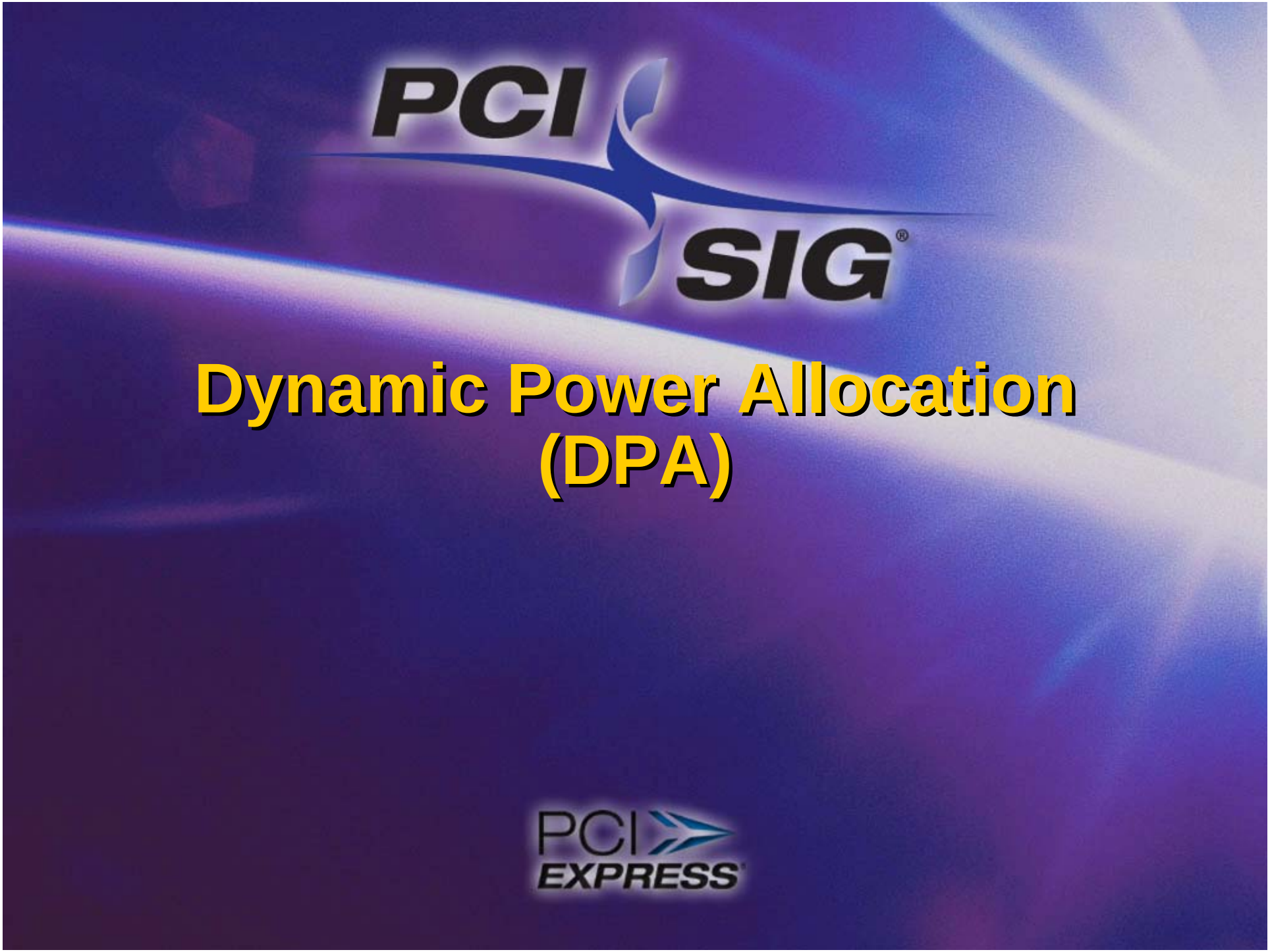
- Uncorrectable Persistent Internal Error
  - ✓ Add Uncorrectable Persistent Internal Error bit to the following
    - – AER Uncorrectable Error Status Register
    - – AER Uncorrectable Error Mask Register
    - – AER Uncorrectable Error Severity Register
  - ✓ Optionally log header
- Uncorrectable Internal Error
  - ✓ Add Uncorrectable Internal Error bit to the following
    - – AER Uncorrectable Error Status Register
    - – AER Uncorrectable Error Mask Register
    - – AER Uncorrectable Error Severity Register
  - ✓ Optionally log header
- Correctable Internal Errors
  - ✓ Add Correctable Internal Error bit to the following
    - – AER Correctable Error Status Register
    - – AER Correctable Error Mask Register
  - ✓ Header not logged

Internal Error Reporting

- Switches
  - ✓ New mechanism for PCIe switches to report internal error that can be used by OS/Hypervisor
- Roots
  - ✓ Roots behave like switches in the context of peer-to-peer traffic
  - ✓ An OS/Hypervisor can use the same reporting mechanism and may take the same action when an error can be isolated to a specific Root port as it does when an error is detected in a Switch
- Endpoints
  - ✓ Use the same mechanism defined for Switches and Roots to report internal errors in Endpoints associated with the PCIe interface
  - ✓ This mechanism is not intended to be used by Endpoints to report errors not associated with the PCIe interface (i.e., application logic). These errors should continue to be handled through proprietary methods employing device specific interrupts.

Multiple Error Handling

- Extend Advanced Error Reporting (AER) capability to allow for multiple error headers to be recorded
  - ✓ New Multiple header log capable and enable bits
  - ✓ Improved error containment and recovery
- Extend AER to define header log overflow indication
  - ✓ New Error overflow bit
  - ✓ Error that requires header log is detected but cannot be logged due to header log resource constraints or multiple header log enable bit not set and first error pointer is valid
- When multiple header log capability is supported and enabled
  - ✓ Error headers are logged in the corresponding order the errors were detected
  - ✓ When the uncorrectable status register for which the first error pointer corresponds to is cleared the first error pointer and the header log are updated to point to the next recorded header
  - ✓ When the last recorded header status bit is cleared then the first error pointer becomes invalid and value of header log is undefined


PCI
SIG
Dynamic Power Allocation
(DPA)
PCI
EXPRESS

Motivation

- Current PCIe provides standard Device & Link-level Power Management
- PCIe 2.0 adds mechanisms for dynamic scaling of Link width/speed
- Device increasingly higher consumers of system power & thermal budget (e.g. gfx)
- No architected mechanism for dynamic control of device thermal/power budgets
- New PCIe power management mechanisms to complement support on-going industry wide efforts to optimize component and platform power management to meet new customer and regulatory operating requirements

DPA

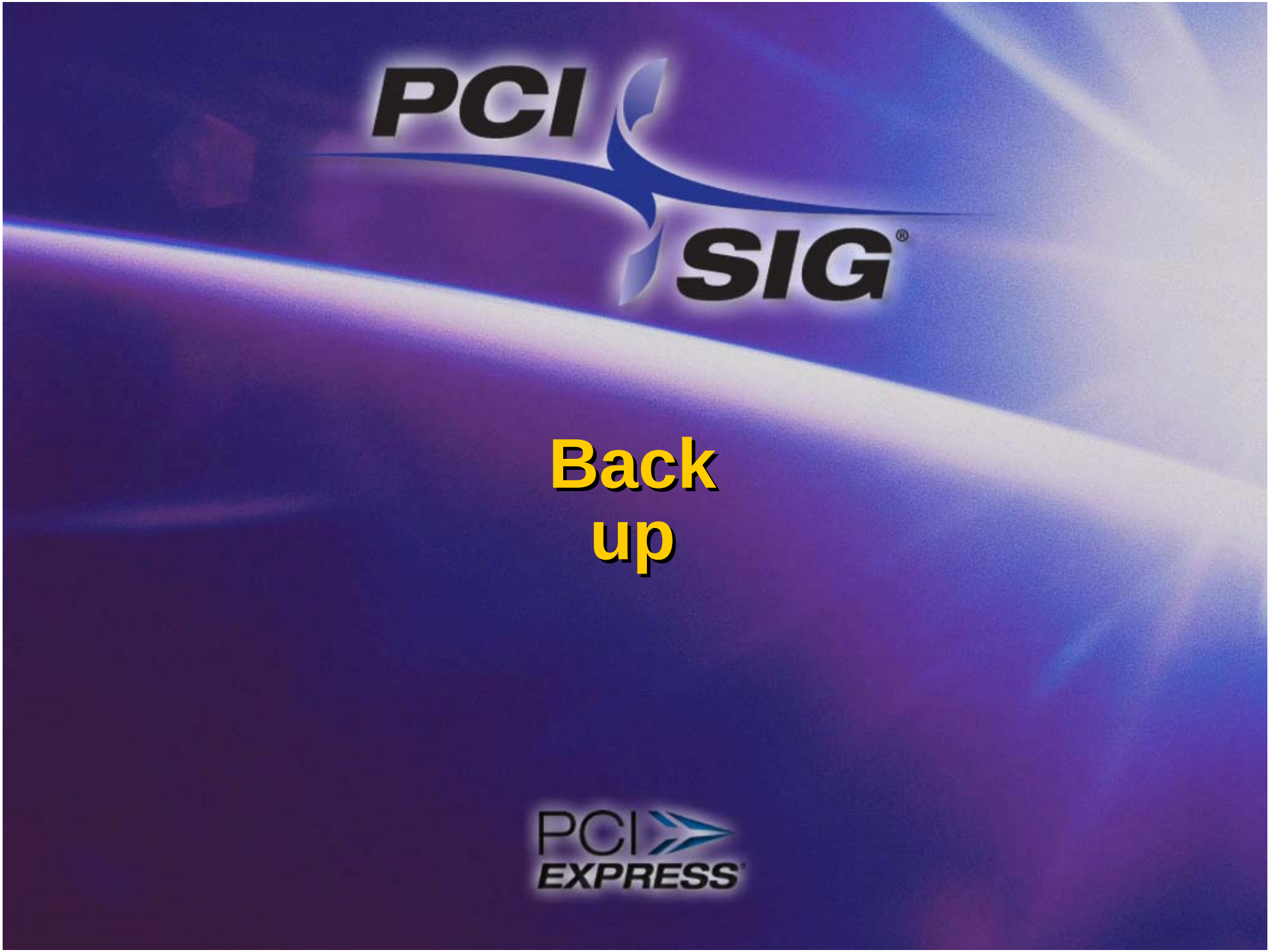
- Extend existing PCIe device PM to provide active (D0) device power management sub-states
  - ✓ Support for up to 256 sub-states per function
  - ✓ Maximum power allocation advertised per sub-state in Watts
  - ✓ Contiguously numbered sub-states 0 to N substate w/o gaps
  - ✓ Each successive sub-state reports maximum power allocation less than or equal to prior sub-state
- Software permitted to dynamically configure a function to operate at any sub-state in any sequence
  - ✓ Function must operate at or below the maximum power allocation corresponding to the selected sub-state
- New DPA capability only for endpoints
  - ✓ DPA capability to enable software to discover and actively manage endpoint function power usage

Protocol Extensions Summary

Extensions	Explanation	Benefit
Data Reuse Hints (DRH)	Request hints to enable optimized processing within host memory/cache hierarchy	Reduce access latency to system memory. Reduce System Interconnect & Memory Bandwidth & Associated Power Consumption
Ordering	New ordering semantic to improve performance	Improved performance (latency reduction)
Atomics	Atomic Read-Modify-Write mechanism	Reduced Synchronization overhead, software algorithm and data structure re-use
I/O Page Faults	Extends IO address remapping for page faults	System Memory Management optimizations
Resizable BAR	Mechanism to negotiate BAR size	System Resource optimizations
Dynamic Power Allocation (DPA)	Mechanisms to allow dynamic power/performance management for substates of D0 (active state)	Dynamic component power/thermal control, manage endpoint function power usage to meet new customer or regulatory operation requirements
Multicast	Address Based Multicast	Significant gain in efficiency compared to multiple unicasts
Internal Error Reporting	Extend AER to report component internal errors (Correctable/ uncorrectable) and multiple error logs	Enables software to implement common and interoperable error handling services. Improved error containment and recovery

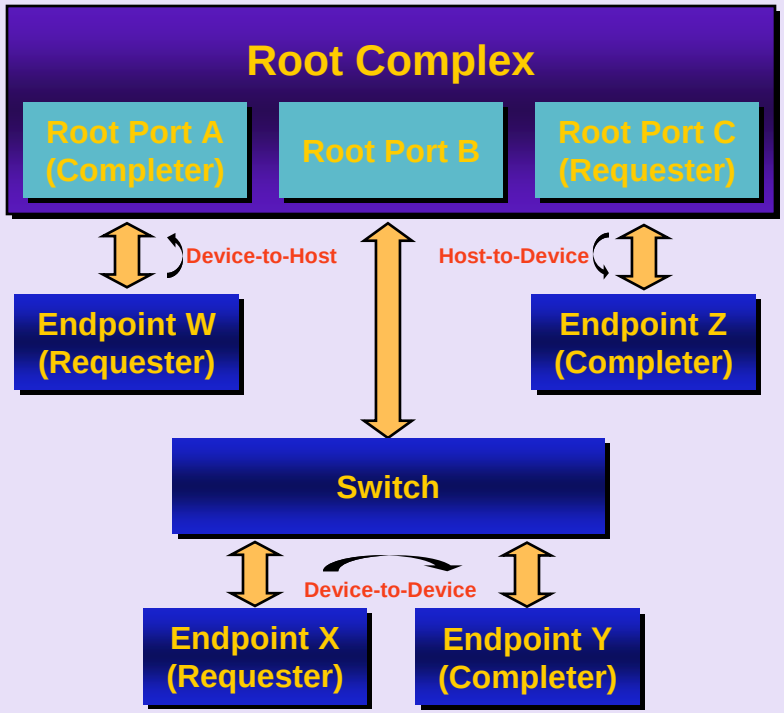
Call to Action

- **Innovate and differentiate your products with PCI Express 2.0 industry standard**
- **Review and provide feedback on PCIe protocol extensions ECRs**
- **Contribute to the evolution of PCI Express architecture**
- **Visit www.pcisig.com for PCI Express specification updates**


PCI
SIG
Back
up
PCI
EXPRESS

Communication Models

- Requester/Completer pairs for Atomics
  - ✓ **Device-to-Host**: Endpoint W to Root Port A
  - ✓ **Device-to-Device**: Endpoint X to Endpoint Y
  - ✓ **Host-to-Device**: Root Port C to Endpoint Z



- PCIe architecture readily supports common format Atomic Request & Completion TLPs serving all 3 cases above
  - ✓ Simple & symmetrical
  - ✓ Flexible; more “future-proof”