SDC 20

BY Developers FOR Developers

Storage Developer Conference
September 22-23, 2020

Introducing Smart Data Acceleration Interface (SDXI)

A new SNIA TWG to standardize the interface for memory-to-memory data movement and acceleration

Shyam Iyer,
Distinguished Member of Technical Staff
Dell
Co-Founder & Interim Chair SNIA SDXI TWG
shyam.iyer@dell.com

Richard A. Brunner,
CTO, Principal Engineer
VMware
Co-founder SNIA SDXI TWG
rbrunner@vmware.com

Agenda

The problem and the need for a solution

Introducing SDXI

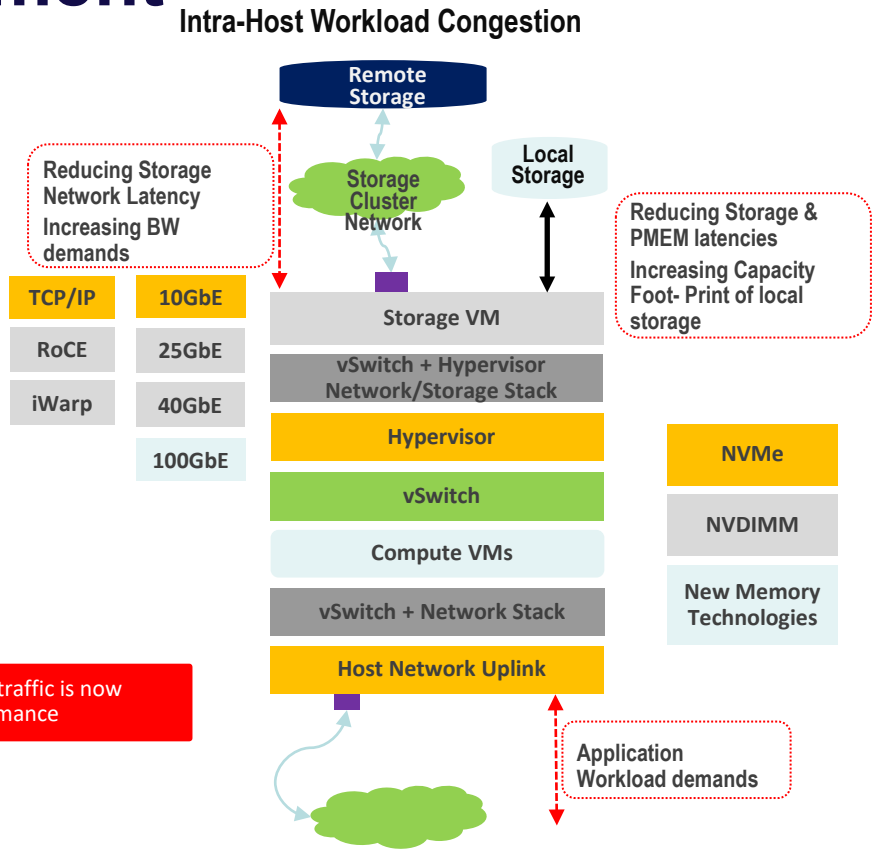
Introduction to SDXI Concepts

Trends

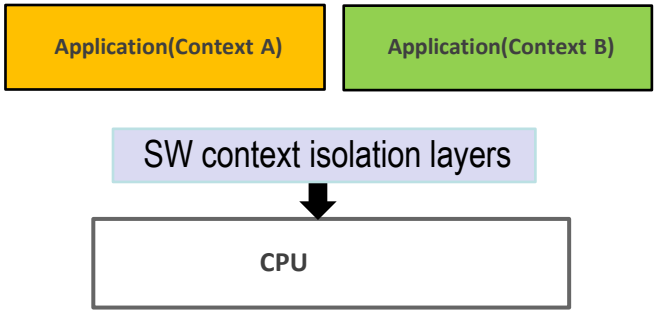
- Need to increase core counts to enable Compute scaling
- Compute density is on the rise
- Converged and Hyperconverged Storage appliances are enabling new workloads on server class systems
  - Data locality is important
- Single threaded performance is under pressure.
- I/O intensive workloads can take away compute CPU cycles available.
- Network and Storage workloads can take compute cycles
- Data Movement, Encryption, Decryption, Compression

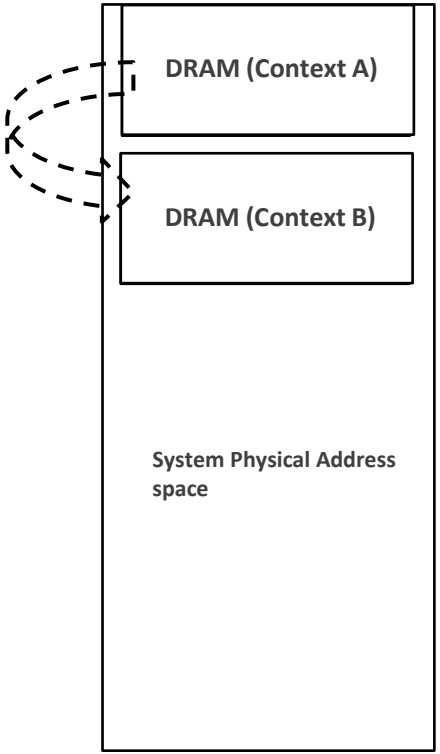
Case Study: Need for Accelerated Intra-host Data Movement

- Each intra-host exchange can comprise multiple memory buffer copies (or transformations)
  - Generally implemented with layers of software stacks:
  - Kernel-to-I/O can leverage I/O-specific hardware memory copy
  - But, SW-to-SW usually relies on per-core synchronous software (CPU-only) memory copies



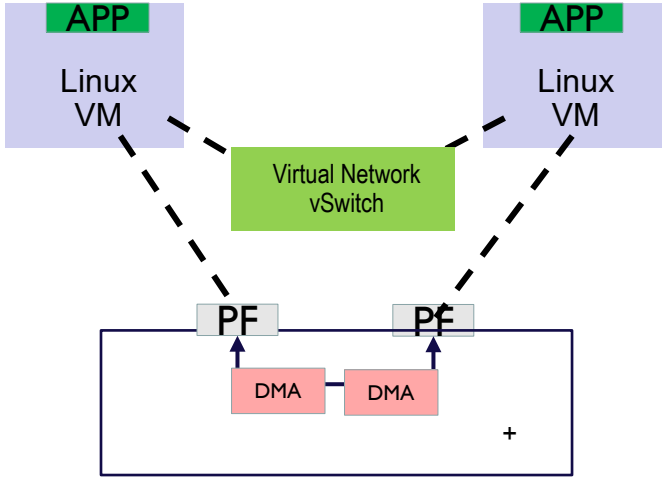
Current data movement standard:



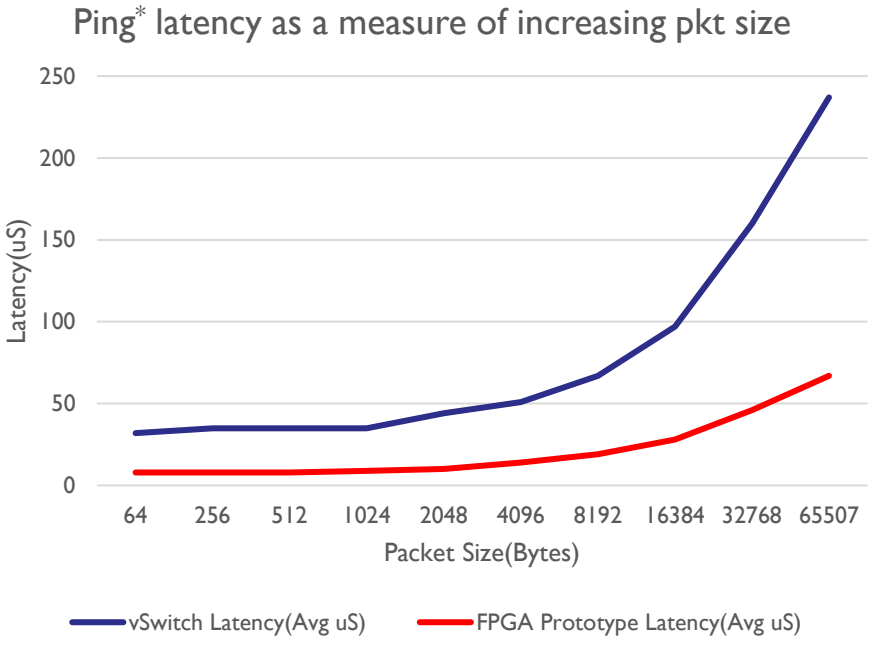


- Software uses stable CPU ISA for memory copies
  - Takes away from application performance
  - Software overhead to provide context isolation

Offload DMA engines ?



- Fast DMA offload engines are
  - HW Vendor-specific implementations
  - HW Vendor specific drivers, APIs
  - Direct access by user level software is difficult
  - Limited Usage Models



+ = Dell FPGA Prototype, (Back2Back DMA with two PCIe PFs, Circa 2016)

* = ping latency was measured with the option "ping -f"

DELL Technologies vmware

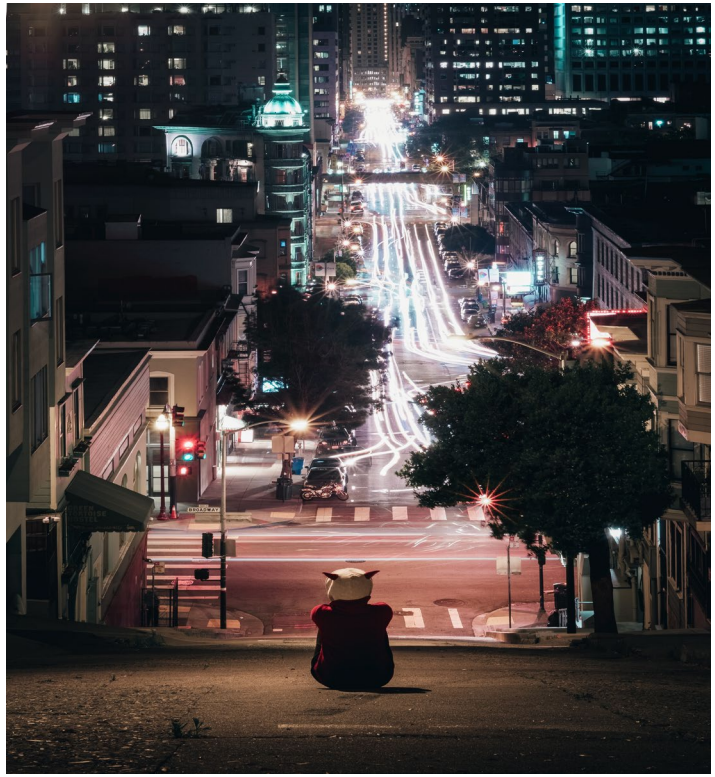
Solution Requirements

1. Need to offload I/O from Compute CPU cycles
2. Need Architectural Stability
3. Enable VM acceleration but,
   - Help migration from existing SW Stacks
4. Create abstractions in Control Path for scale and management
5. Enable performance in data path with offloads

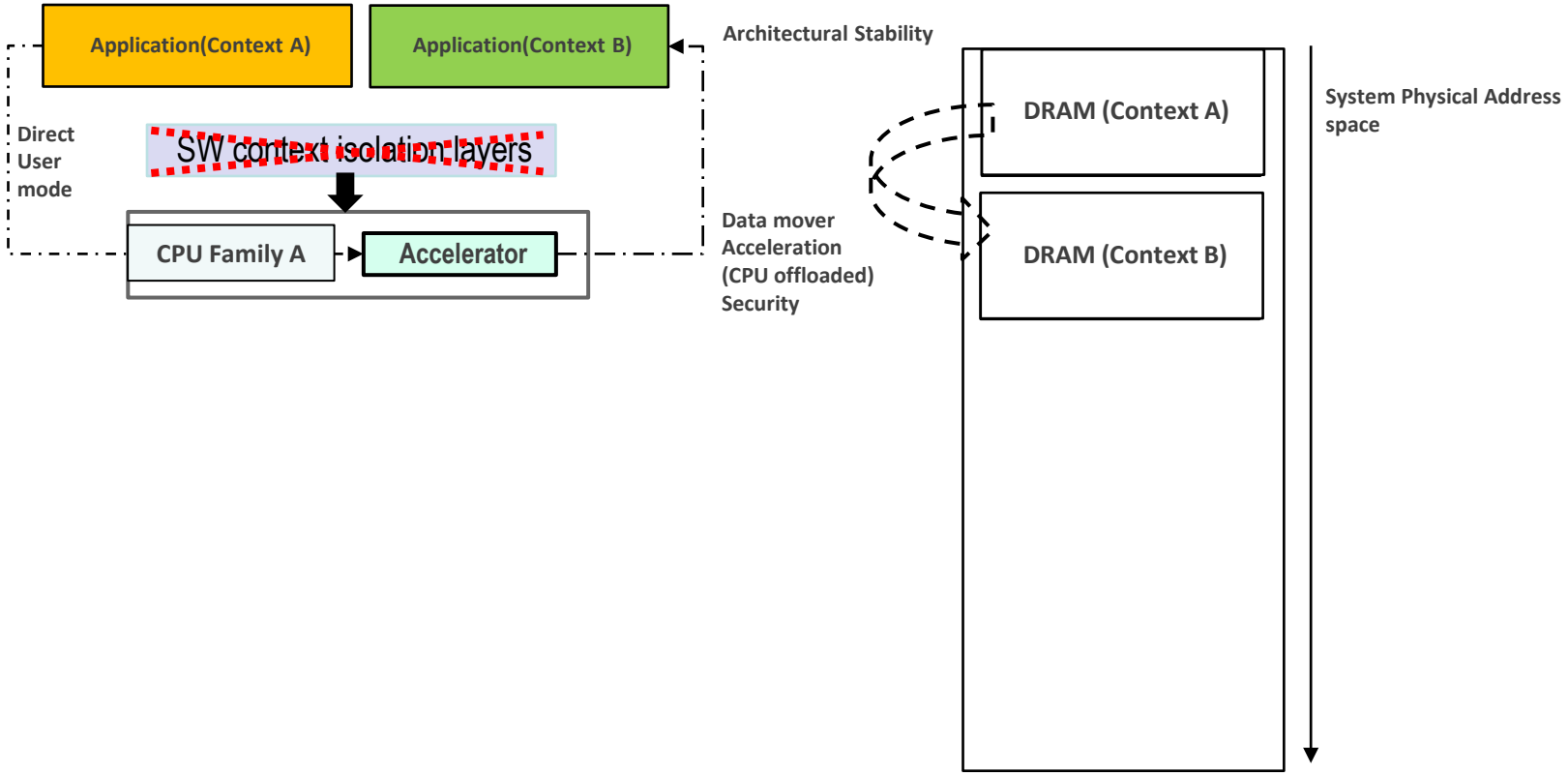
Emerging Server & Storage Architectures

1. Memory-centric architectures.
2. New memory interconnects.
   a. CXL
   b. Gen-Z
3. Varied memory types.
4. Heterogenous architectures are becoming main stream.
5. The need to democratize data movement.

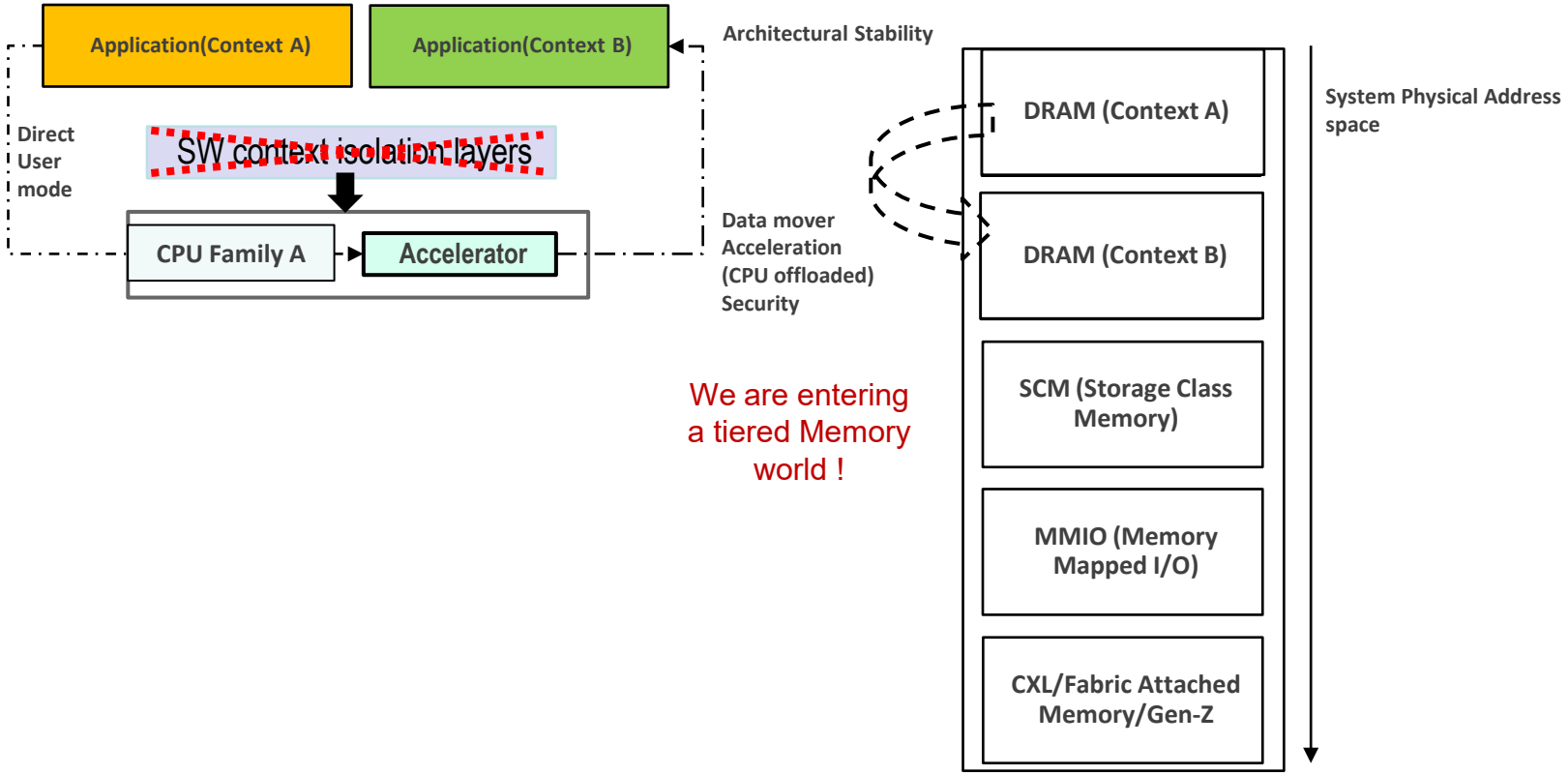
Looking into the horizon …



Emerging Needs: New Memory Architectures

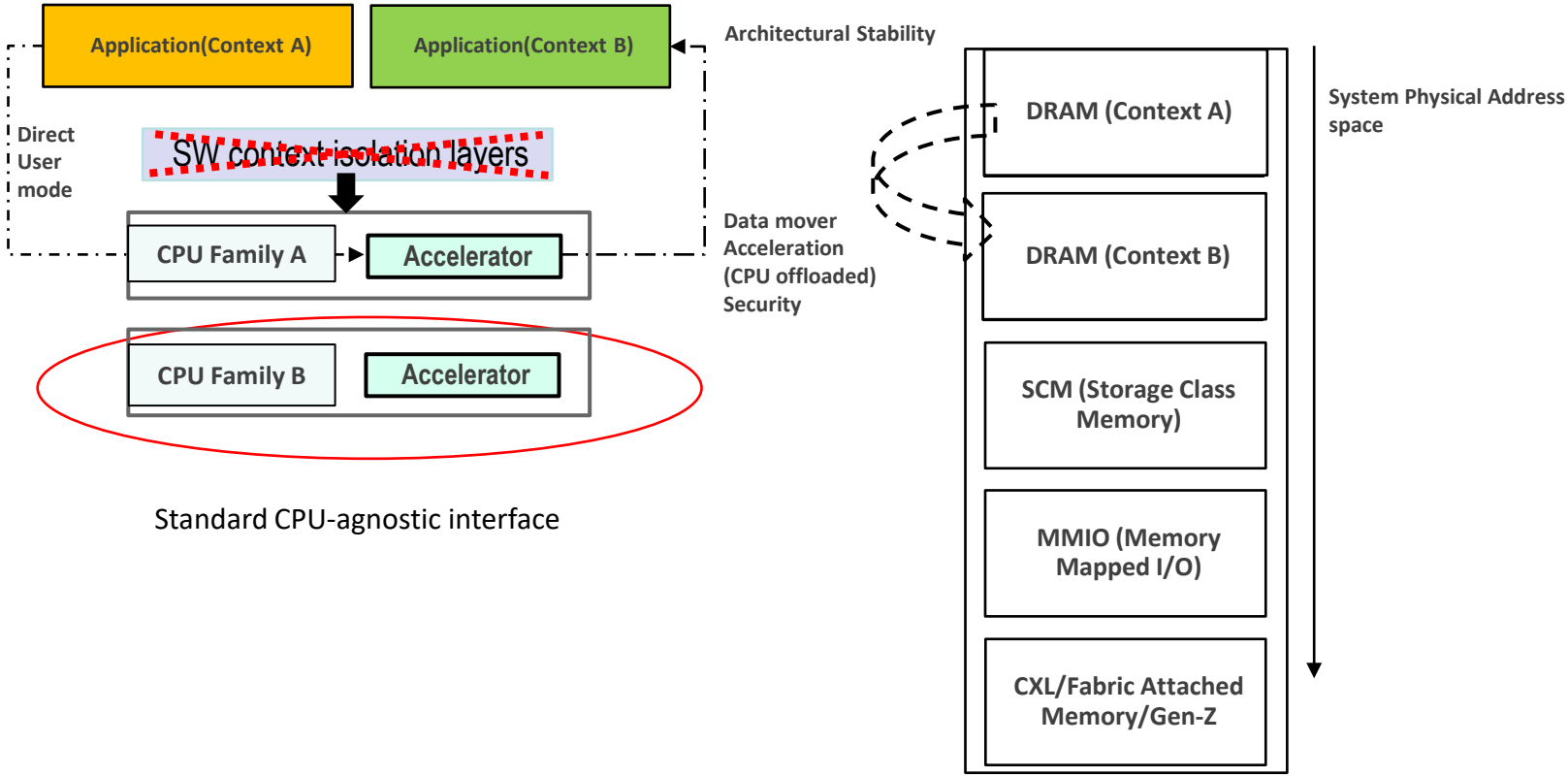


Emerging Needs: New Memory Architectures



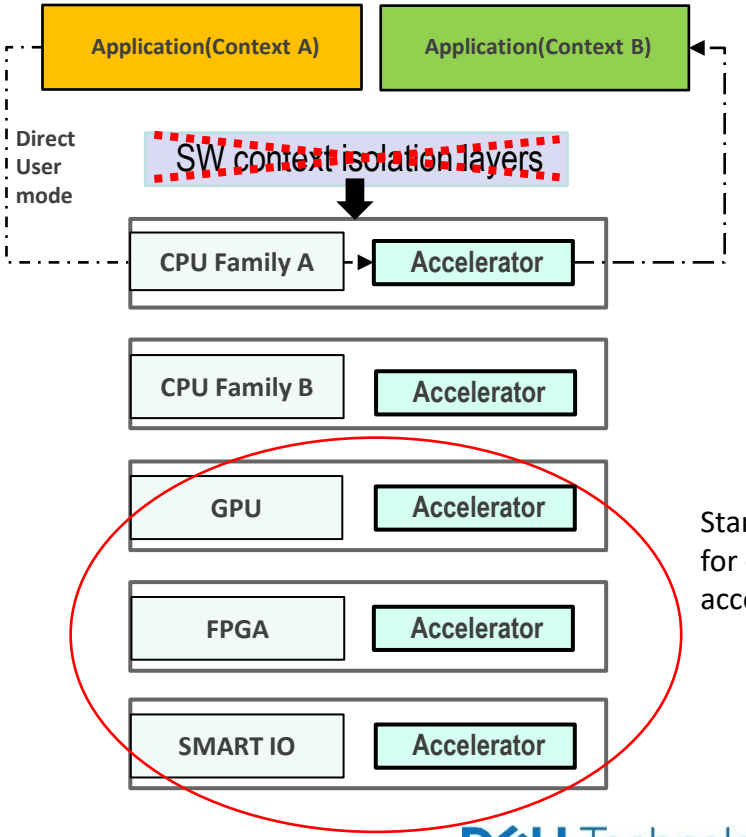
We are entering a tiered Memory world !

Architectural Stability

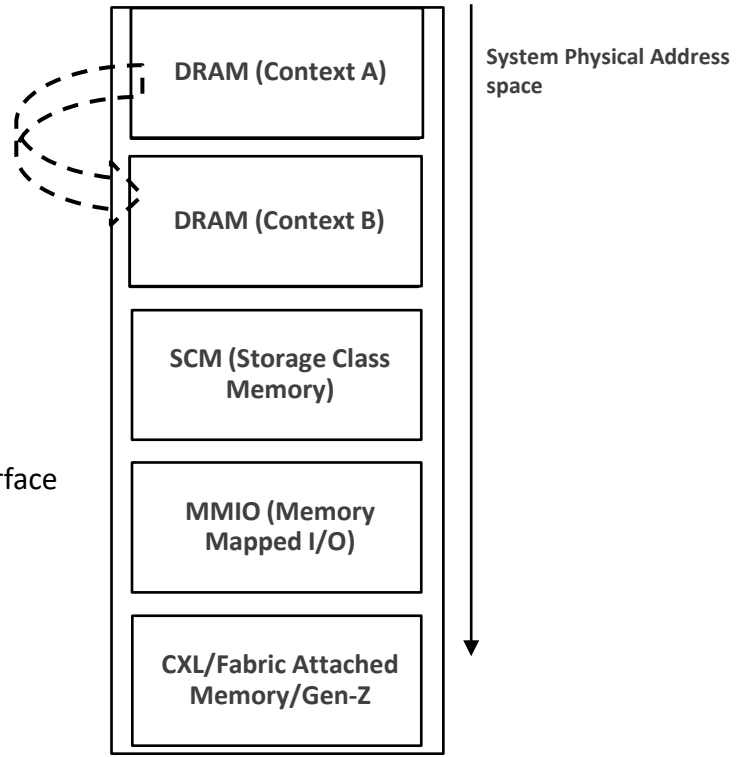


Standard CPU-agnostic interface

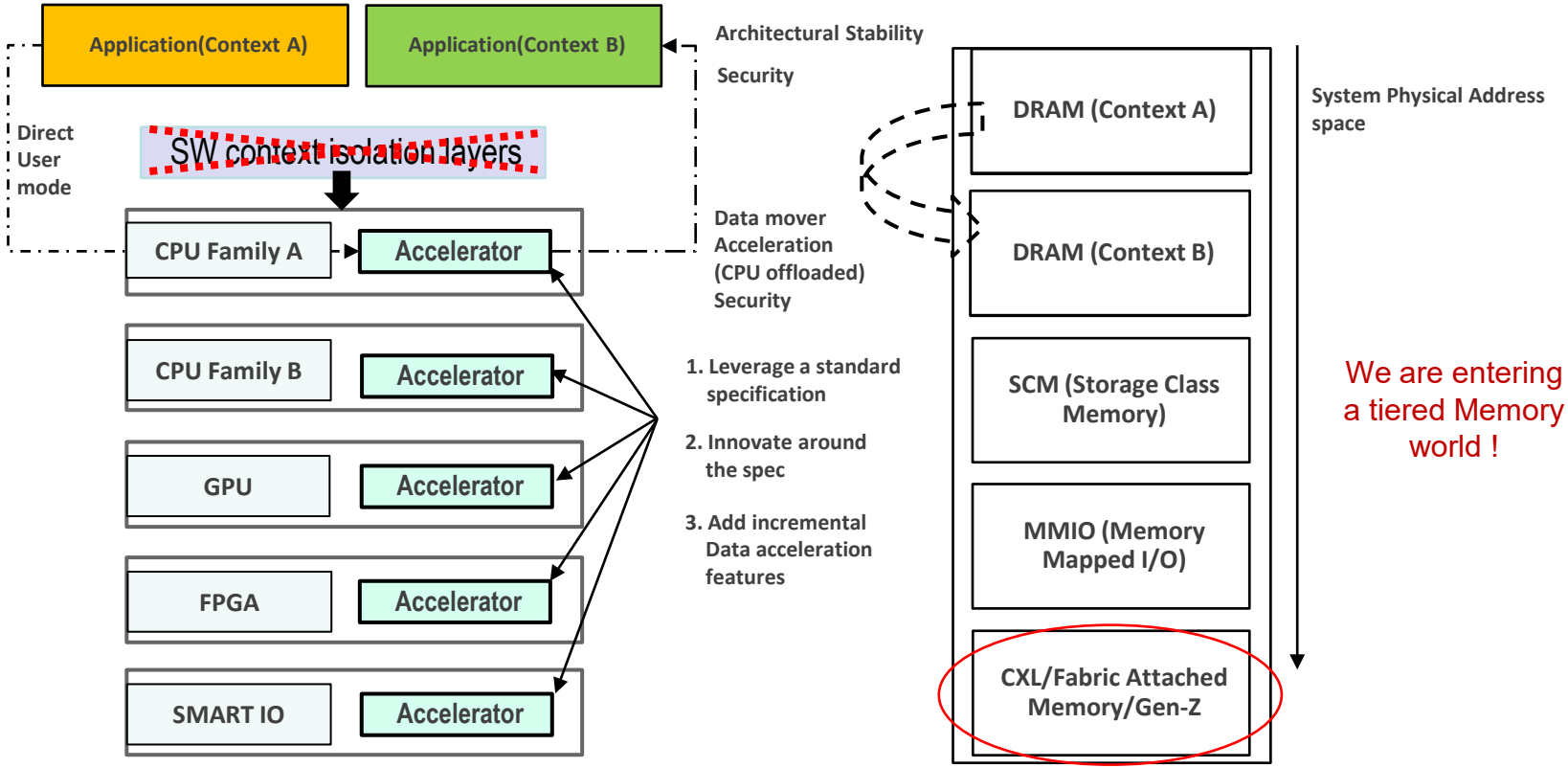
Enabling Accelerators



Standard interface for different accelerators



The need for an industry standard



We are entering a tiered Memory world !

Agenda

The problem and the need for a solution

Introducing SDXI

Introduction to SDXI Concepts

Acknowledge

- Philip Ng
  - AMD Sr. Fellow, Co-founder, Co-author for the SDXI TWG
- AMD, Dell, VMware are contributing the starting spec.

Introducing SNIA SDXI TWG

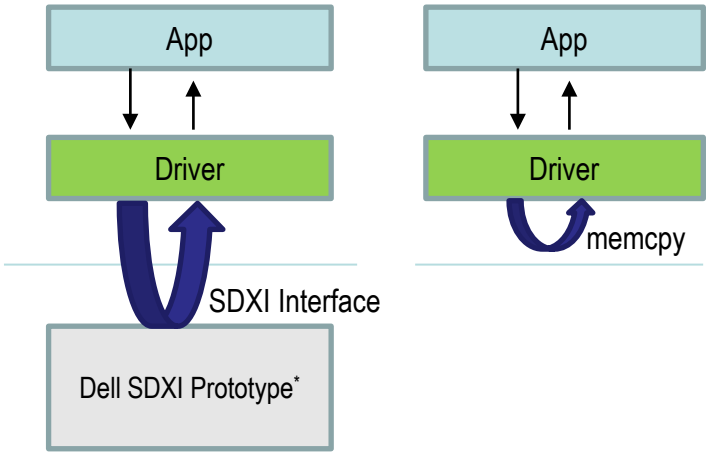
SDXI Charter

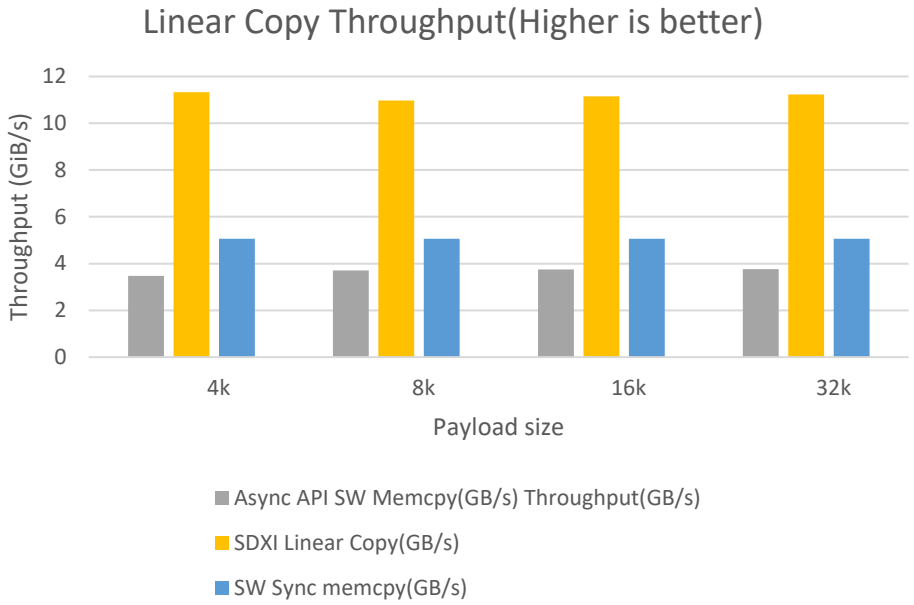
- Develop and Standardize a Memory to Memory Data Movement and Acceleration interface that is -
  - Extensible
  - Forward-compatible
  - Independent of I/O interconnect technology

Design Tenets

- Data movement between different address spaces.
  - Includes user address spaces, different virtual machines
- Data movement without mediation by privileged software.
  - Once a connection has been established.
- Allows abstraction or virtualization by privileged software.
- Capability to quiesce, suspend, and resume the architectural state of a per-address-space data mover.
  - Enable “live” workload or virtual machine migration between servers.
- Enables forwards and backwards compatibility across future specification revisions.
  - Interoperability between software and hardware
- Incorporate additional offloads in the future leveraging the architectural interface.
- Concurrent DMA model.

Dell Vetted Out The Design ...





Enabling the Ecosystem is key to success !

*= (Gen3 PCIe FPGA Implementation)
Contact for more details: shyam<dot>iyer<at-the-rate>dell<dot>com

TWG External Collaboration

- Calling other SNIA TWGs/Forums
  - PM Persistent Memory work group
  - CS (Computational Storage Technical work group)
  - Network Storage
  - CMSI (Compute, Memory, Storage Initiative)
- Calling other Standards groups
  - PCISIG, CXL, OFA, UEFI, Gen-Z etc

SDXI TWG Program of Work

- Contributed spec being reviewed by SNIA base v1.0 architecture..
- Post v1.0 Focus
  - New data mover operations for smart acceleration
  - Data mover operations involving persistent memory targets
  - Cache coherency models for data movers
  - Security Features involving data movers
  - Connection Management architecture for data movers
- Encourage adopting companies to work towards compliant software implementations and driver models.
- Educate and encourage adoption by OS, Hypervisors, OEMs, Applications and Data Acceleration vendors

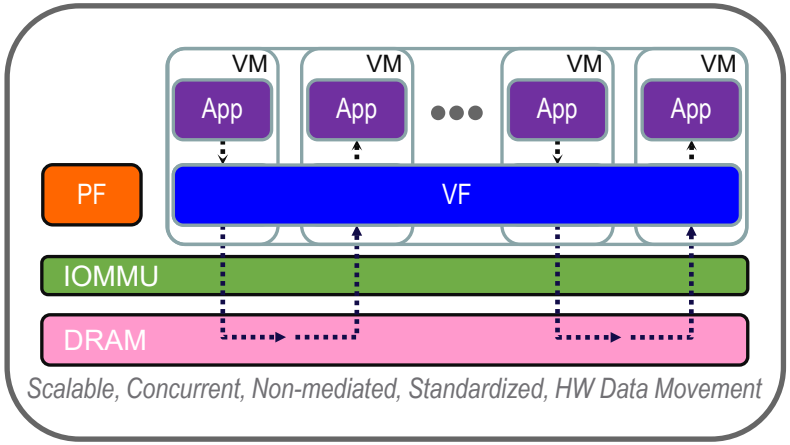
Agenda

The problem and the need for a solution

Introducing SDXI

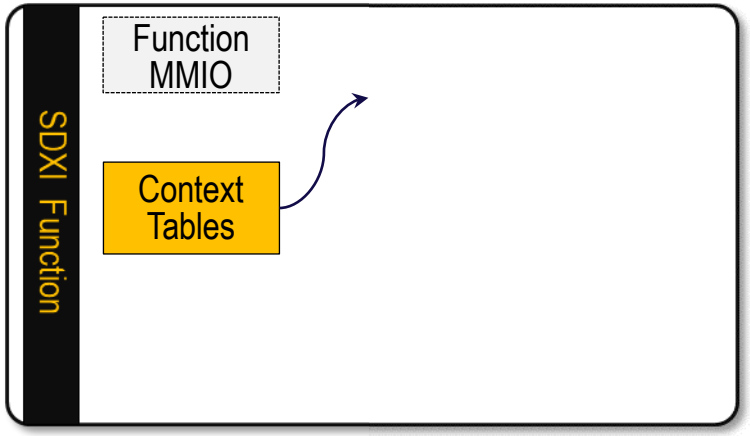
Introduction to SDXI Concepts

Accelerated, Virtualized, Standardized HW Data Movement



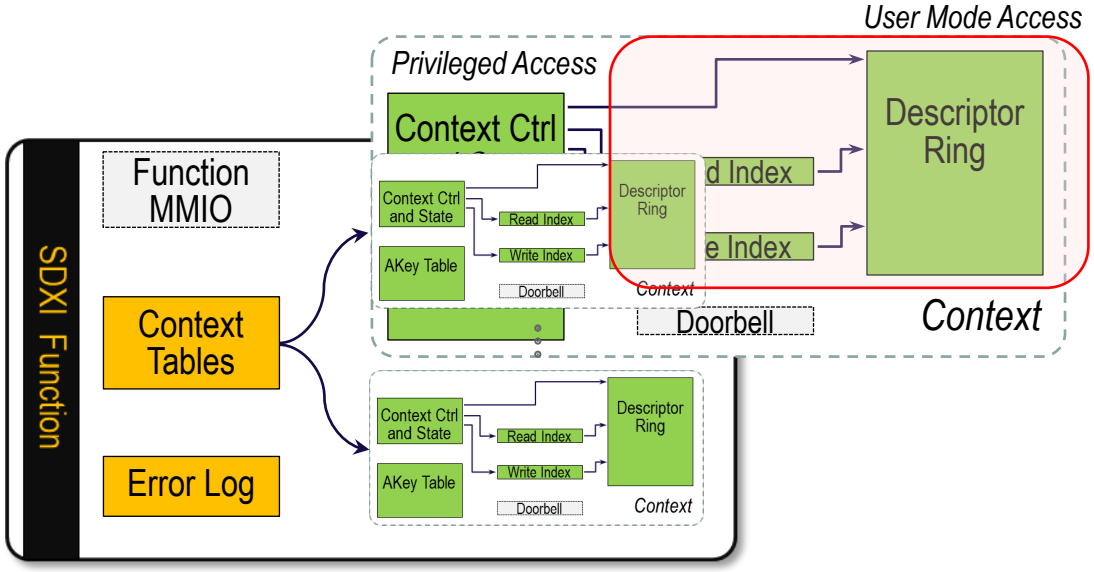
1. A standardized data mover interface independent of actual implementations & underlying I/O.
2. Data movement between different address spaces both within and across VMs.
3. Data movement without mediation by privileged software(Hypervisor).
4. An interface and architecture that can be abstracted or virtualized by privileged software.
5. Concurrent DMA model.
6. Allow “live” workload or virtual machine migration between servers.
7. Forwards and backwards compatibility.
   - Allow Hardware, software Interoperability
8. Incorporate additional offloads in the future.

SDXI Function Architecture



- Function setup & control accelerator-independent.
- One standard descriptor format.
- All SDXI context state resides in memory.
  - No special mechanisms to serialize state.
- Very easy to virtualize.

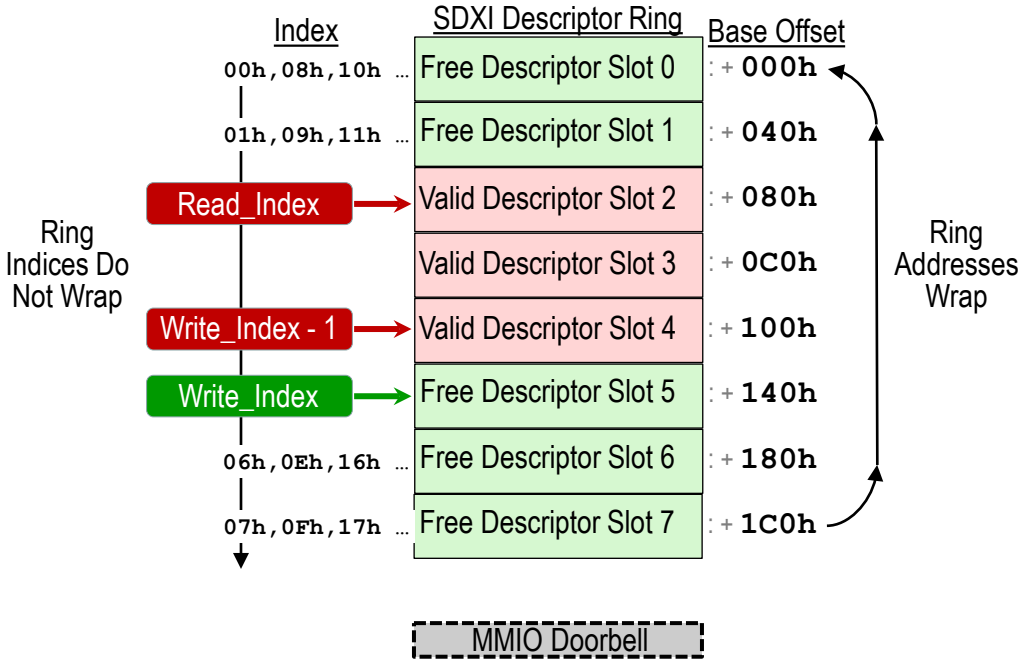
SDXI Function Architecture



- Ring state directly managed by user mode.
- Multiple contexts per function.
  - Allows easier multi-producer support within one VM.
- One way to setup, control, and quiesce contexts.
- One flexible way to log errors.

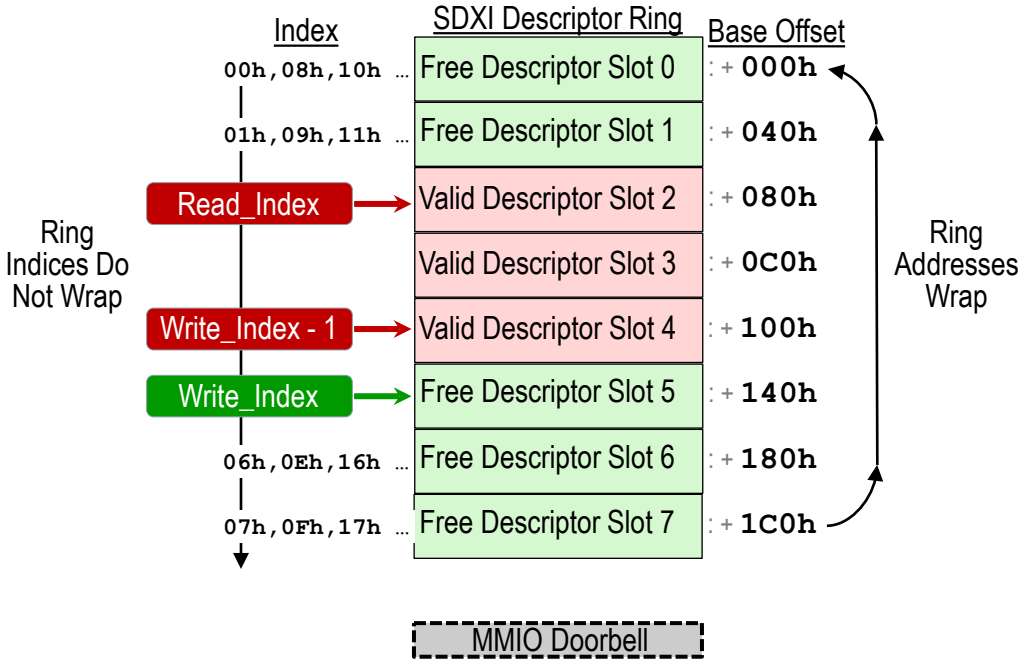
A Ring is A Ring is A Ring (1)

- No need to keep inventing user-mode circular ring.
- Ring is managed by position-independent Indices.
  - Index pointers reside in user mode memory.
- SDXI Function starts reading descriptors at Read_Index.
  - Stops reading at Write_Index.
- SW starts writing new descriptors at Write_Index.
  - Stops writing at Read_Index.



A Ring is A Ring is A Ring (2)

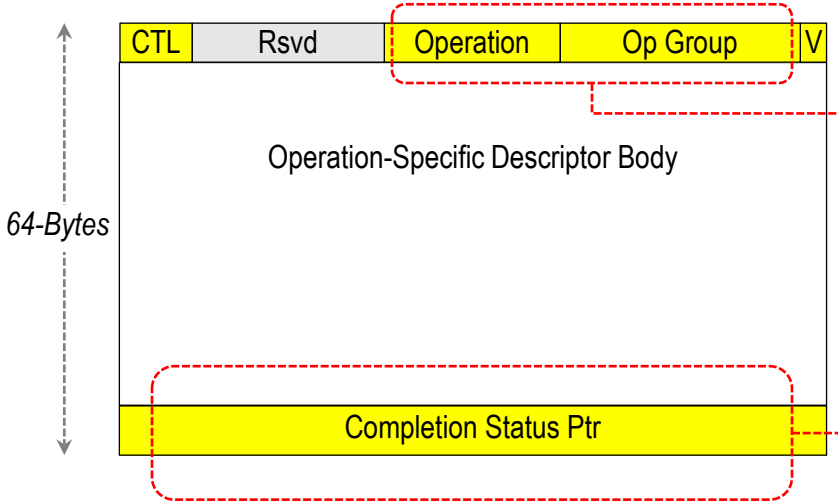
- Descriptors are processed (issued) in-order by function.
  - Executed out-of-order.
  - Completed out-of-order.
  - Read_Index is incremented after each issue step.
- Function may aggressively read valid descriptors…
  - Between Read & Write indices w/o writing Doorbell.
- But Doorbell ensures new descriptors are recognized.



SDXI Maximum parallelism of operations.
Quiescing & Serializing state at well-defined boundaries.

DELL Technologies vmware

A Standard Descriptor Format (1)



Architecturally Registered Operation Groups:

DMA Base	Administrative	Vendor-Defined
Atomic	Connect	Others …

DMA: Copy, RepCopy, WriteImm
Atomic: Bitwise Ops, Add, Sub, Swap, Min, Max, CmpSwap, etc
Admin: Start/Stop/Update/Sync, Interrupt Function & Contexts (easily virtualizable)
Connect: Connect VMs through SDXI functions (easily virtualizable)

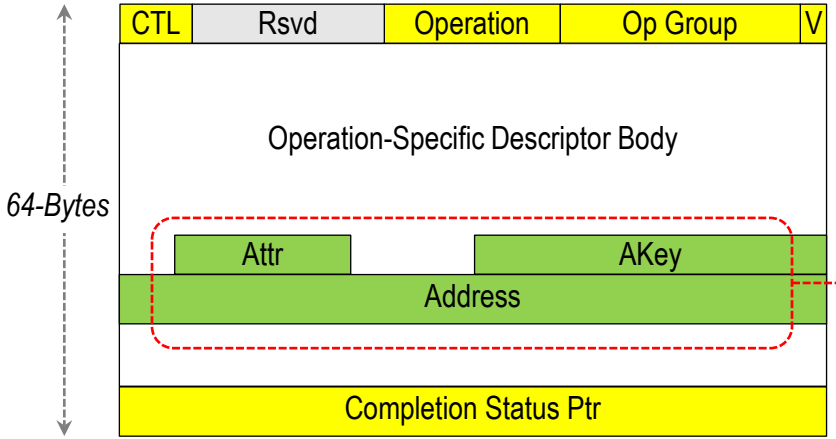
Completion Status Block

- Initialized by SW, Decremented by Function on Success
- Can be shared across groupings of descriptors
- Errors flagged in Completion Signal and Misc Status

Completion Signal
Misc Status

*Room for lots of future operations

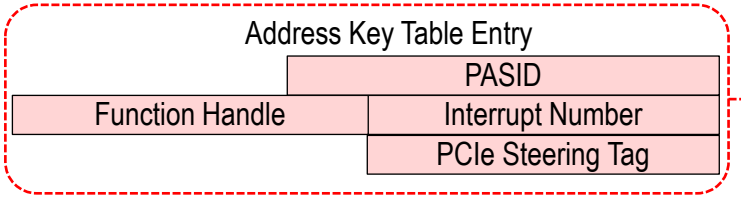
A Standard Descriptor Format (2)



A memory location is always specified as a triple:

- Address Space ID: Index to Context Address Key Table Entry
- 64-bit Address
- Cacheability Attributes

Generated Address can be HPA, HVA, GPA, GVA and always translated through IOMMU.

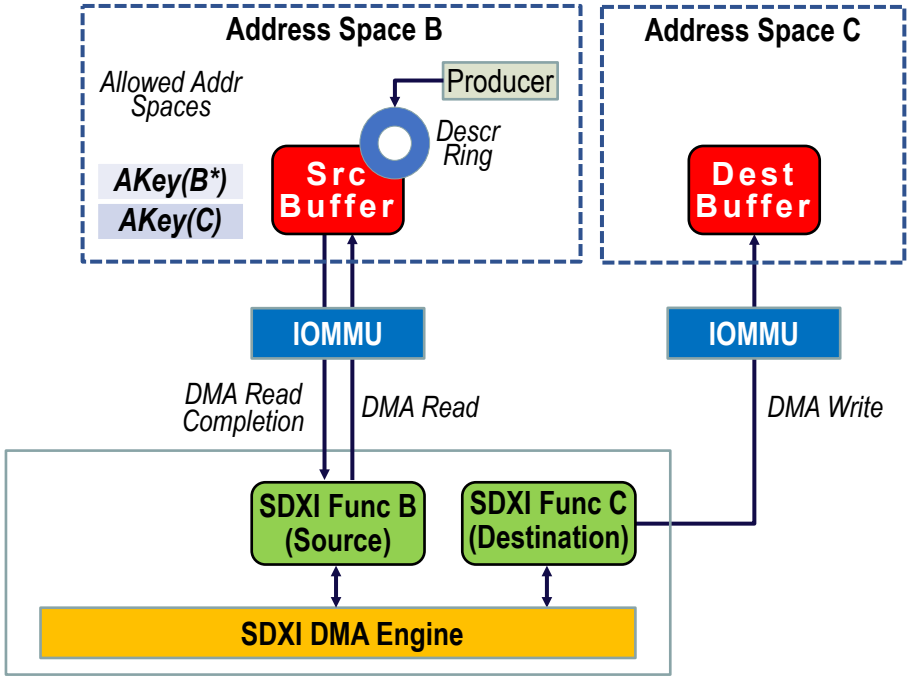


An AKey table entry encodes all valid address spaces, PASIDs and interrupts available to the function context.

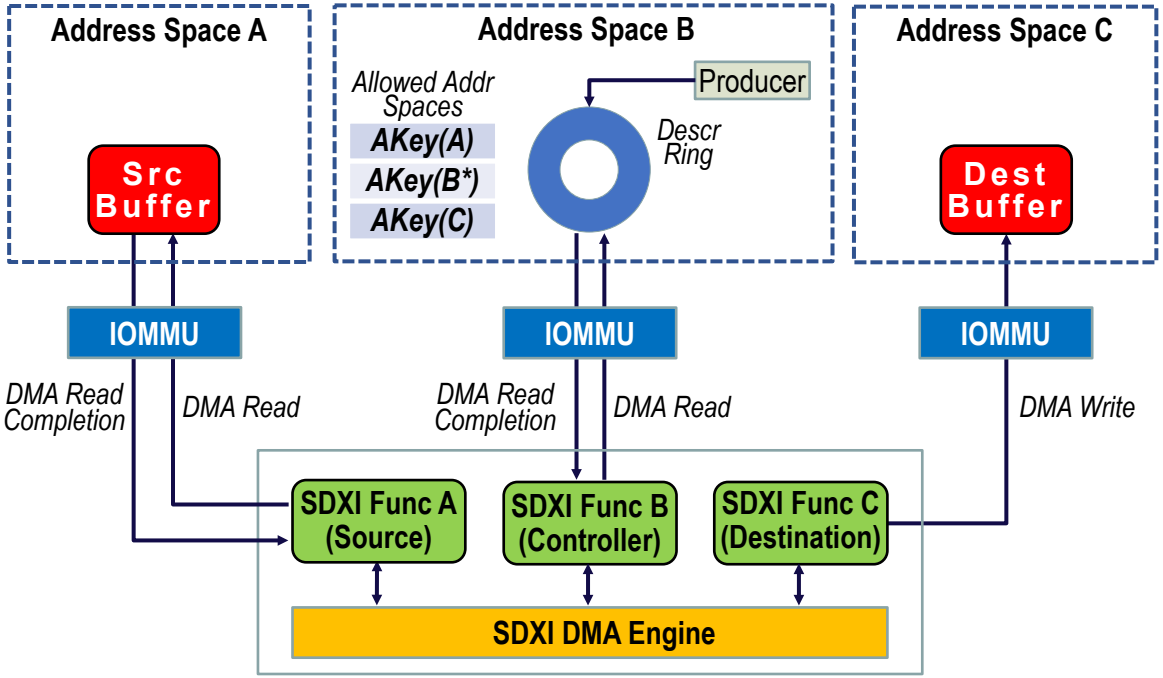
Any descriptor within a context can reference an AKey table entry.

*Room for lots of future operations

Multi-Address Space Data Movement (1)

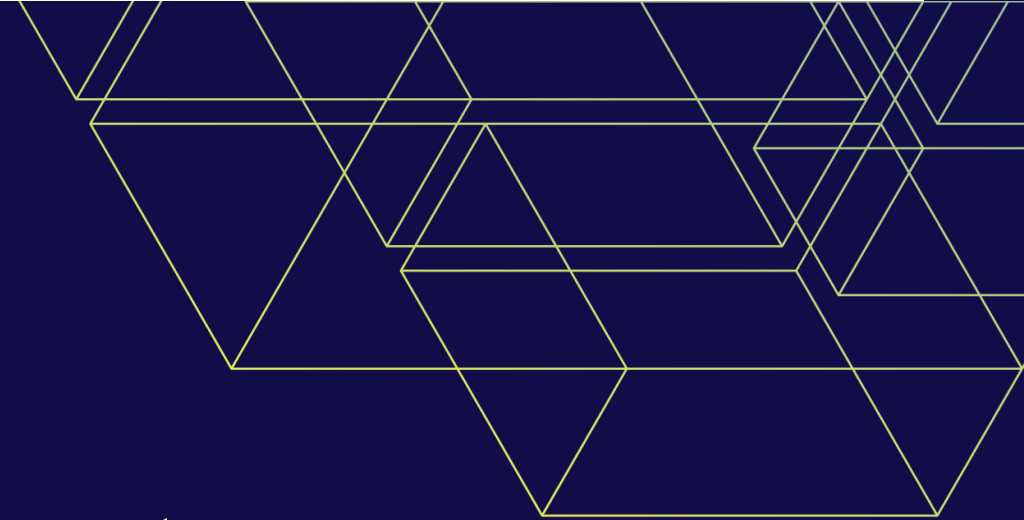


Multi-Address Space Data Movement (2)



Summary

- As CPU cores scale, the usage & demand for ever larger and faster data exchanges scales.
  - It scales among kernels, applications, VMs, and I/O devices.
  - Future Network and Storage technology will especially require this scaling.
- Solutions to provide data-movement scaling requires not only acceleration, but a standard interface that supports software reuse and virtualization.
- Dell, AMD, and VMware are contributing a proposed starting point for this interface to SNIA.
  - In this session, we have discussed key concepts of this proposed interface.
  - SNIA has authorized SDXI (Smart Data Acceleration Interface) Technical Work Group to begin work on the interface.
- We believe this interface proposal to be of broad value.
- Come join us in the TWG!



Please take a moment to rate this session.

Your feedback matters to us.