CoreWeave

ATS vs IOMMU-regroup:

A journey to optimize GDRDMA in cloud scenario

Liang Yan, Sr. Software Engineer, Havock

Havock: Kernel Hardware - Acceleration - Virtualization - Operating Systems - Containerization - Kubelet

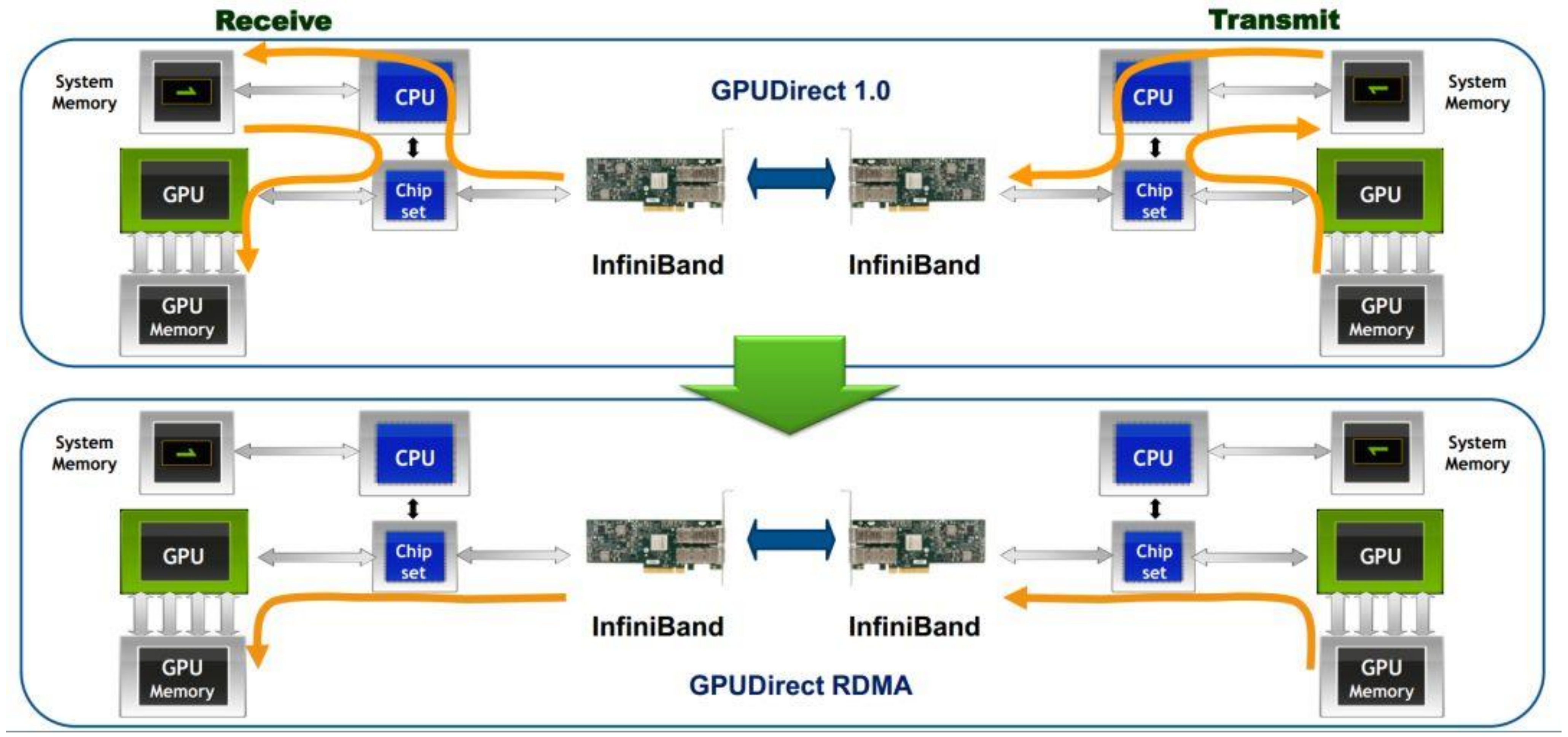
Table of Contents

Background

Background

- RDMA Perftest
  - ib_write_bw –use_cuda
- GDR
  - GPU Direct RDMA
  - ACS/ATS
  - IOMMU

GPU Direct RDMA



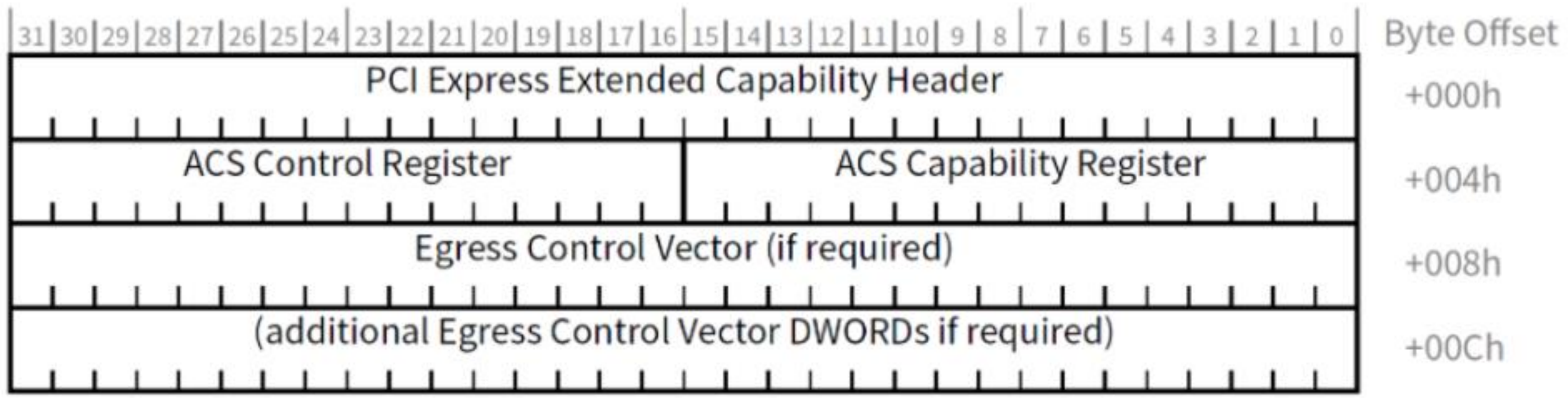
ACS/ATS

ACS is a protocol based on pcie capability:

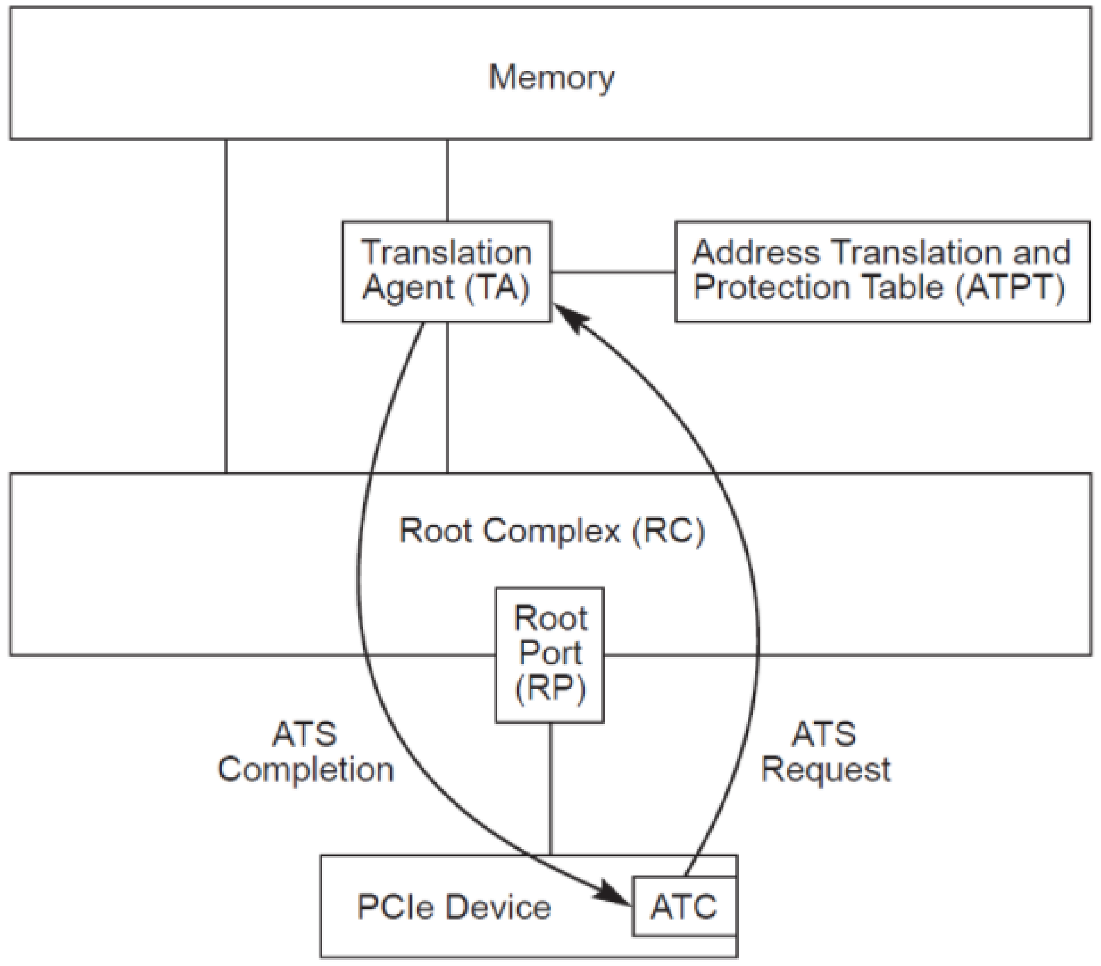
PCIe allows for peer-to-peer transactions without needing to pass through the Root Complex. This provides better performance (at the expense of security), so ACS was introduced to prevent peer-to-peer access.

ATS is an IOMMU unit, TA on IOMMU, ATC on EP

MMU	:	TLB
IOMMU	:	ATS



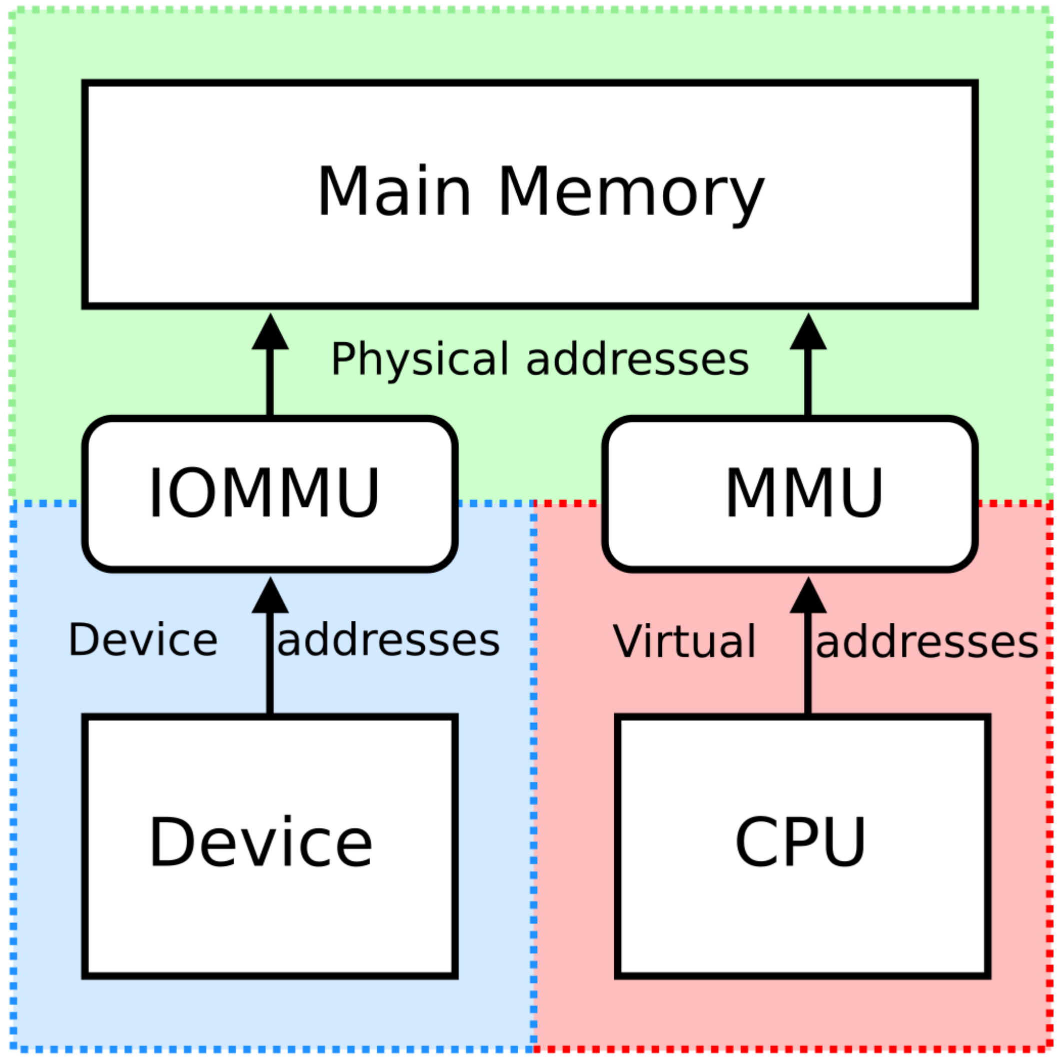
31 … 16	15 … 0	Byte Offset
PCI Express Extended Capability Header		+000h
ACS Control Register	ACS Capability Register	+004h
Egress Control Vector (if required)		+008h
(additional Egress Control Vector DWORDs if required)		+00Ch



IOMMU

IOMMU

IOMMU needs ACS to isolate VFs from SRIOV, IOMMU group.
IOMMU needs ATS to accelerate page table translation between devices



Where the story begins

Where the story begins

Same PCIe Switch	Enable ACS	Disable ACS
Baremetal	~85 Gb/s	~120 Gb/s
VM	~85 Gb/s	~85 Gb/s

Disable ACS	Same PCIe Switch	Intra-NUMA	Inter-NUMA
Baremetal	~120 Gb/s	~255 Gb/s	~250 Gb/s
VM	~85 Gb/s	~255 Gb/s	~250 Gb/s

Same PCIe Switch	Enable ACS	Disable ACS
Baremetal	~85 Gb/s	~390 Gb/s
VM	~85 Gb/s	~85 Gb/s

Disable ACS	Same PCIe Switch	Intra-NUMA	Inter-NUMA
Baremetal	~390 Gb/s	~255 Gb/s	~250 Gb/s
VM	~85 Gb/s	~255 Gb/s	~250 Gb/s

Where the story begins

Hardware perspective

- PCIe topology, Dell vs SMC
- PCIe Switch PEX890xx
- GPU H100
- Mellanox CX-7

Software perspective

- NVIDIA peermem
- CONFIG_PCI_P2PDMA
- VFIO-noiommu
- IOMMU-regroup

Where the story begins

Software perspective

NVIDIA peermem

- nv_peer_memory -> nvidia-peermen
- Dependent io_peer_mem

Where the story begins

Software perspective

CONFIG_PCI_P2PDMA:

- 3.10 first introduced
- 6.2 into userspace
- Storage: NVME over RDMA/GDS

Requirement:

• Memory:	ZONE_DEVICE
• CPU:	host_bridge
• Device:	Controller Memory Buffer/NVRAM

Cons:

Hardware compatibility(cxl 3.0)
Bypass page-cache
Vender difference

Software perspective

VFIO-NOIOMMU

- CONFIG_VFIO_NOIOMMU
- Option "`enable_unsafe_noiommu_mode`"
- CAP_SYS_RAWIO privileges

Software perspective

IOMMU Group:

- pcie_acs_override=downstream,multifunction"
- iommu=pt

IOMMU regroups

- disable_acs_redirect: Disable acs redirect for devices, and these devices will be in one group.
- The ACS Request P2P Request Redirect, P2P Completion Redirect and P2P Egress Control bits are disabled which is sufficient to always allow passing P2P traffic uninterrupted.

Software perspective

Conclusion

Enable ACS
Enable IOMMU
Enable ATS:
- Enable ATS from both RC and EP
- Set ACS directed translated P2P bit = 1

Same PCIe Switch	Disable ATS	Enable ATS
Baremetal	~390 Gb/s	~390 Gb/s
VM	~85 Gb/s	~390 Gb/s

The story continues

GH200 nvlink-c2c

- Device Topo: IOMMU
  - NUMA
  - GDRDMA
- Cache Coherency: ATS
  - SCF
  - UVM

Q&A

Thanks all for attending!

Reference

Reference:

- https://github.com/NVIDIA/open-gpu-kernel-modules
- https://github.com/linux-rdma/perftest
- https://liujunming.top/2022/04/02/Introduction-to-GPUDirect-RDMA
- https://docs.nvidia.com/dgx/dgxh100-user-guide/introduction-to-dgxh100.html
- https://developer.nvidia.com/blog/nvidia-grace-hopper-superchip-architecture-in-depth/
- https://en.wikipedia.org/wiki/Input%E2%80%93output_memory_management_unit