



Flash Memory Summit

# Multi-Host Sharing of NVMe Drives and GPUs Using PCIe Fabrics

Vincent Haché

Principal Applications Engineer,  
Microsemi Corporation



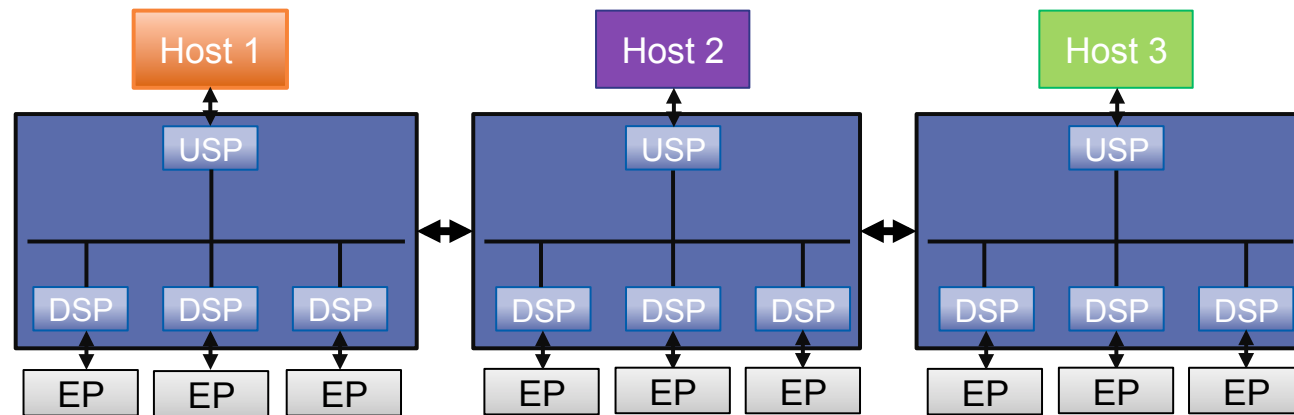
Flash Memory Summit

# Introduction

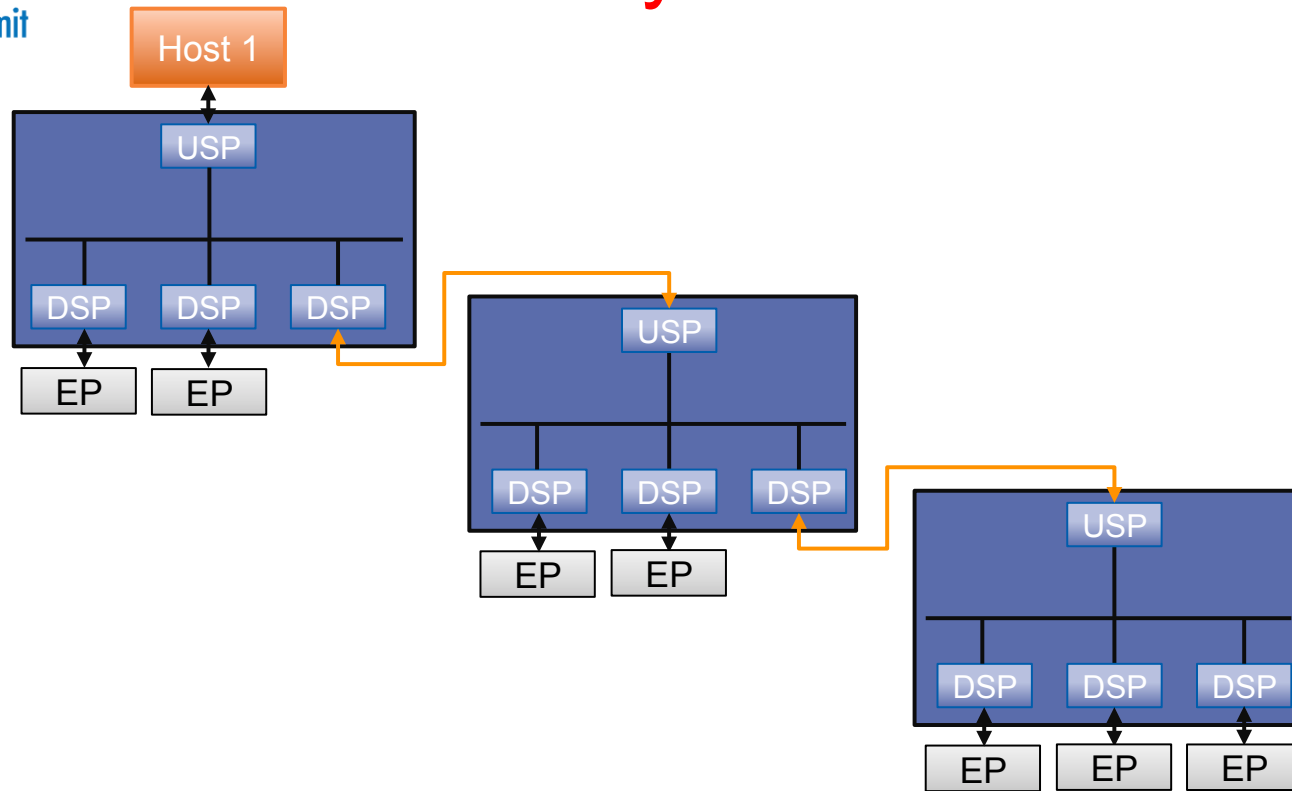
- Increase in use of GPUs and NVM in DC
- System designers need:
  - Efficient resource deployment
  - High-BW, low-latency interconnect
  - Flexible, composable architectures
- There are restrictions in standard PCIe that present challenges for system design

# PCIe Hierarchy Restriction

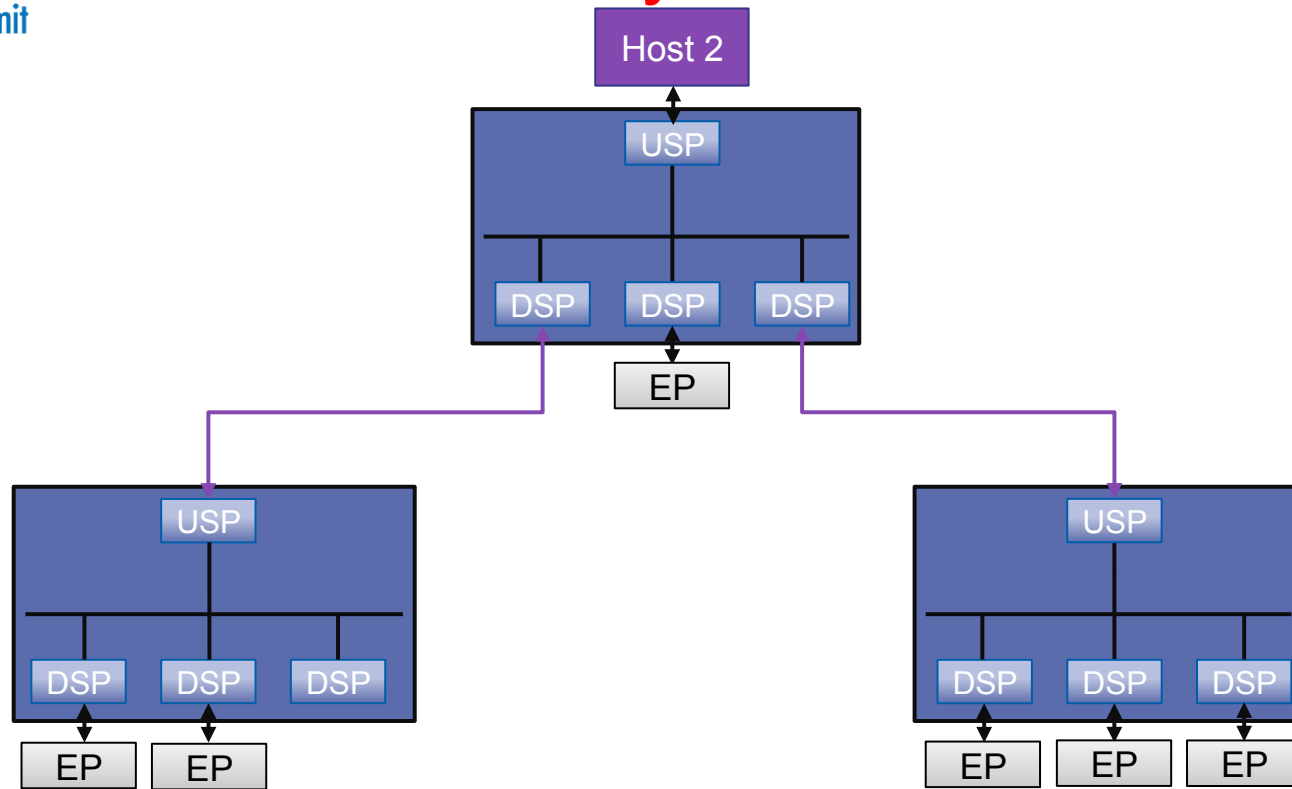
- PCIe hierarchy is restrictive, making scale out challenging



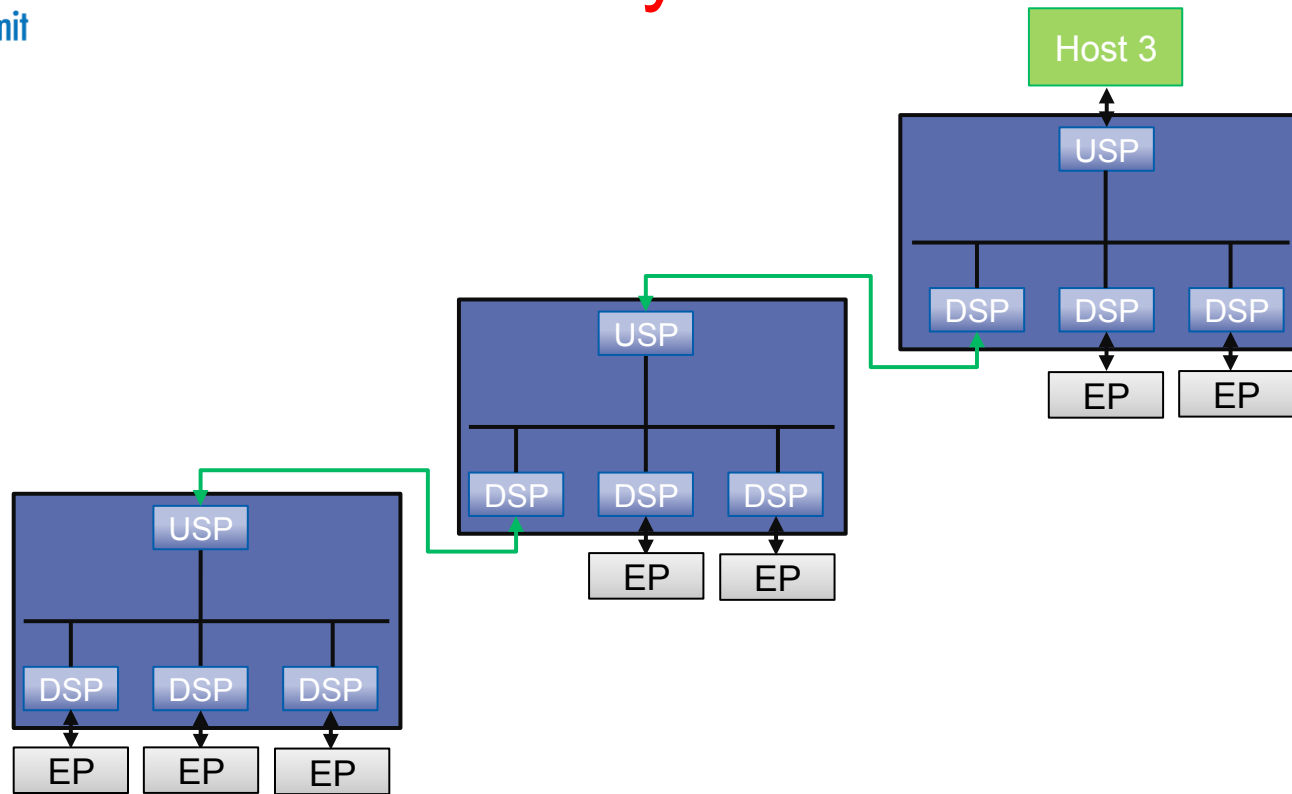
# PCIe Hierarchy Restriction: Host 1



# PCIe Hierarchy Restriction: Host 2



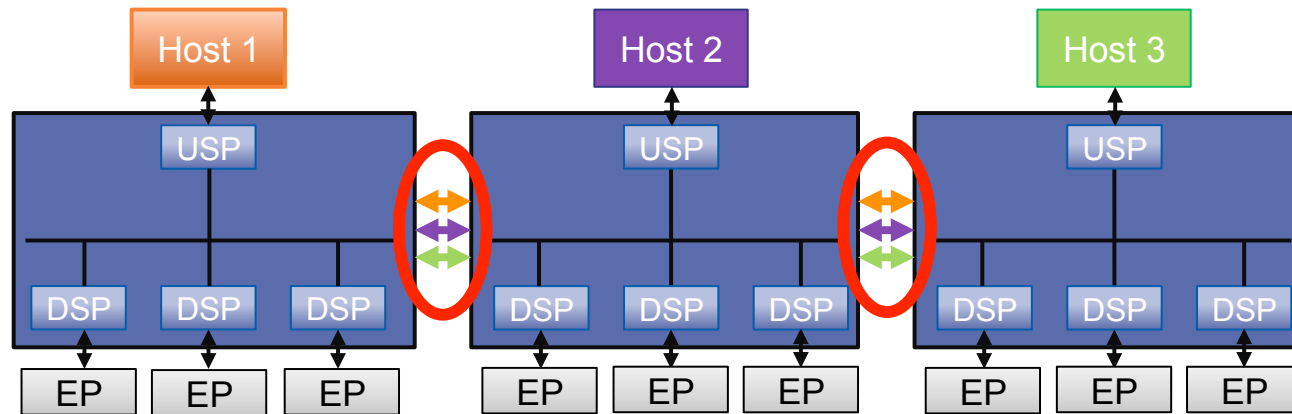
# PCIe Hierarchy Restriction: Host 3





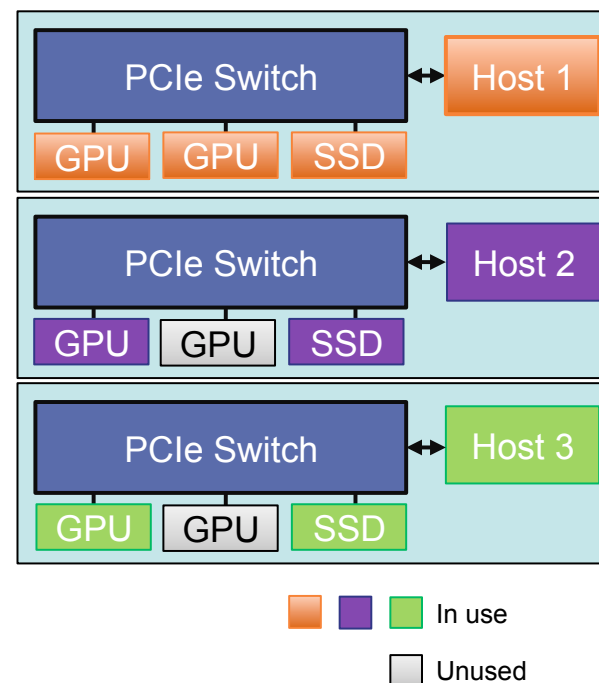
## PCIe Hierarchy Restriction (continued)

- The multiple links required for transparent scale out complicates design and decreases efficiency



# PCIe Single Domain Restriction

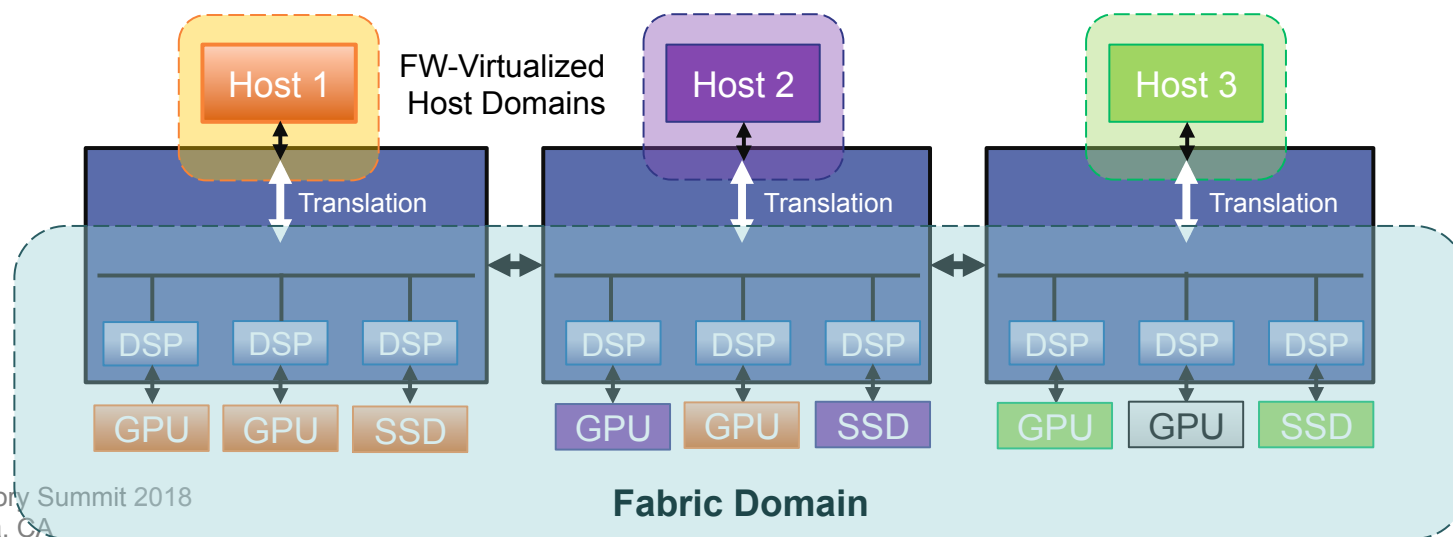
- PCIe is single domain
  - Unused EPs are stranded
  - Complicated, non-standard NT drivers required for sharing





# PCIe Fabrics for Scaling

- Fabric routing is proprietary, non-hierarchical
- Fabric links are shared among hosts

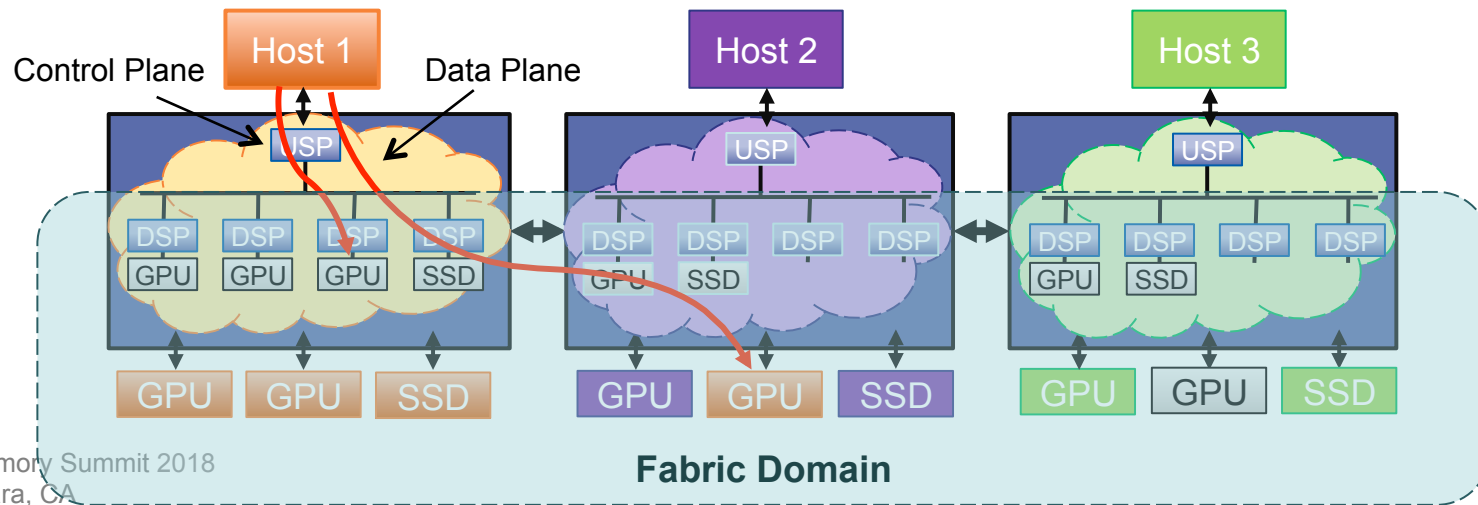






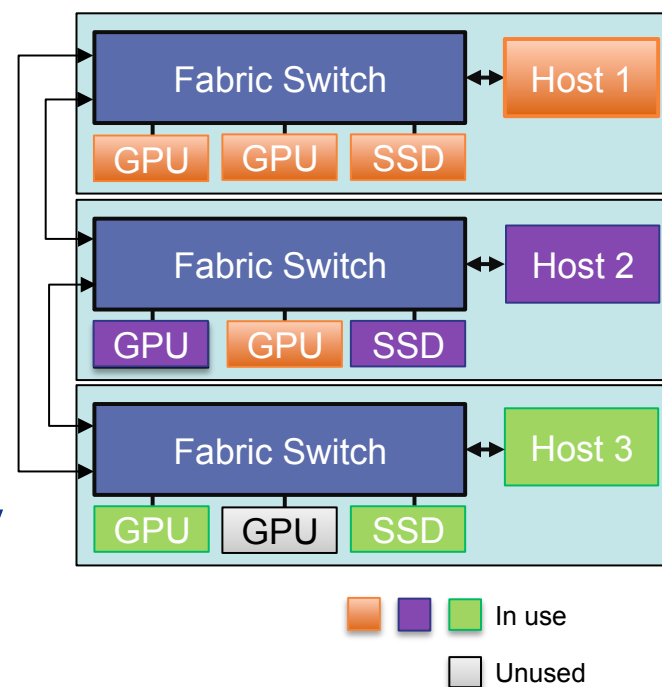
# PCIe Fabrics for Scaling (continued)

- Embedded CPU handles the control plane, but data is routed directly by switch HW



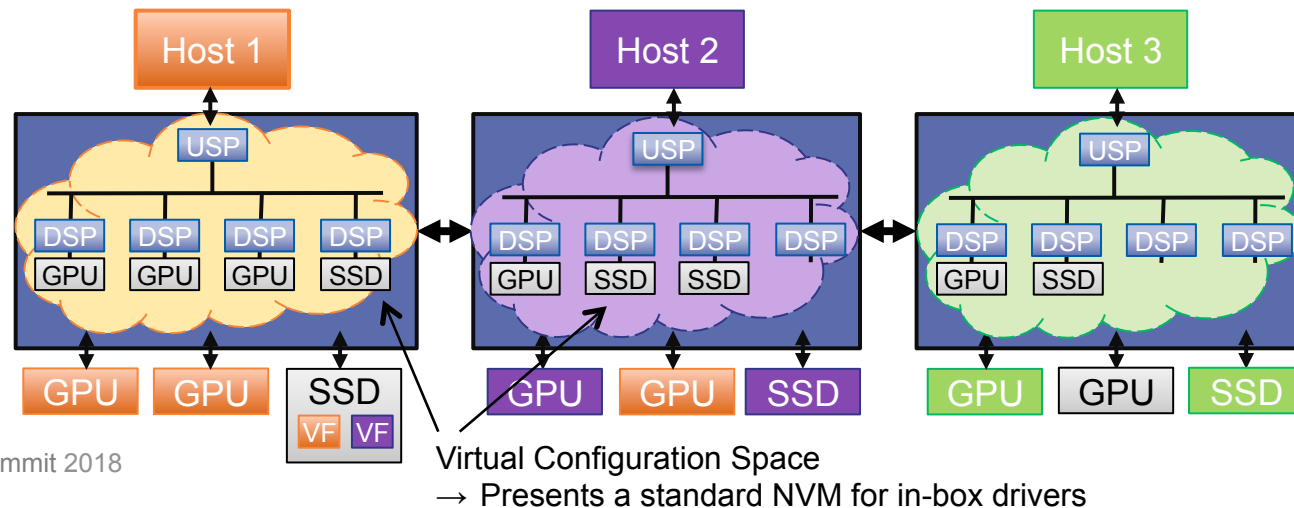
## Device Sharing on a PCIe Fabric

- Unused devices can be dynamically assigned (no longer stranded)
- Low-latency, high-BW P2P within the rack
- Standard drivers to simplify system development



# Multi-host Sharing of SSDs

- Fabric resources assigned by function
- SR-IOV: EP appears as multiple functions

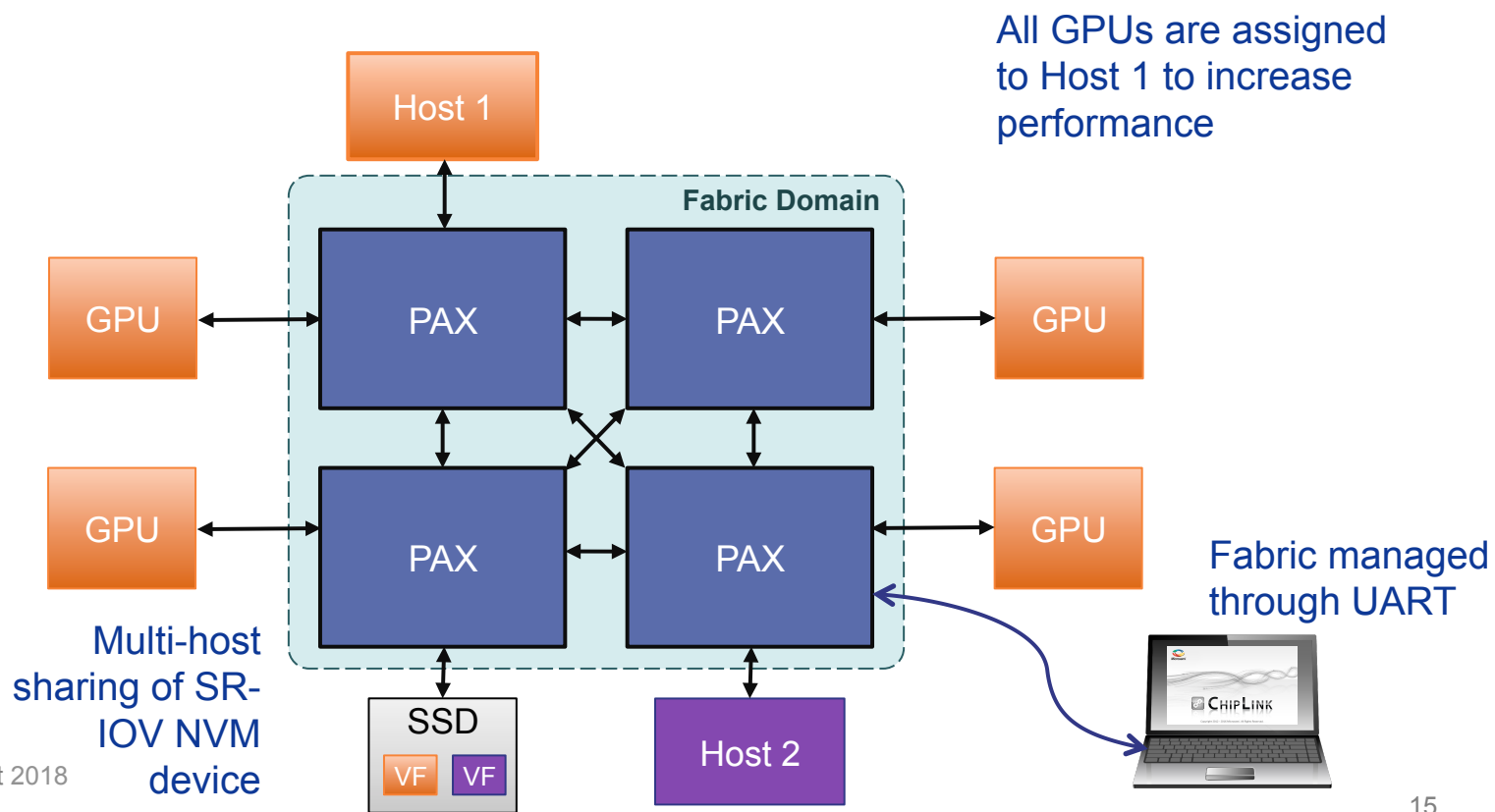




## Demo: Multi-host Sharing of NVMe and GPUs

- Dynamic partitioning of GPUs and multi-host sharing of SR-IOV SSDs in real time
  - Standard host drivers in Windows Server 2016 and Ubuntu Server 16.04 LTS
- GPU P2P transfers across the fabric
  - CUDA P2P BW test and TensorFlow cifar10 image classification multi-GPU training algorithm

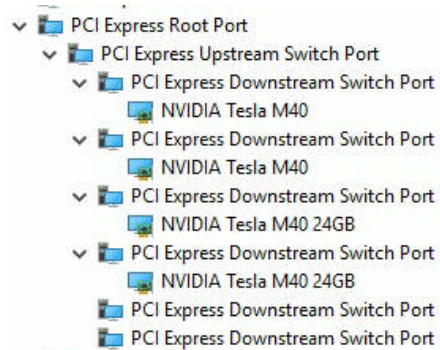
## Demo: Multi-host Sharing of NVMe and GPUs (continued)



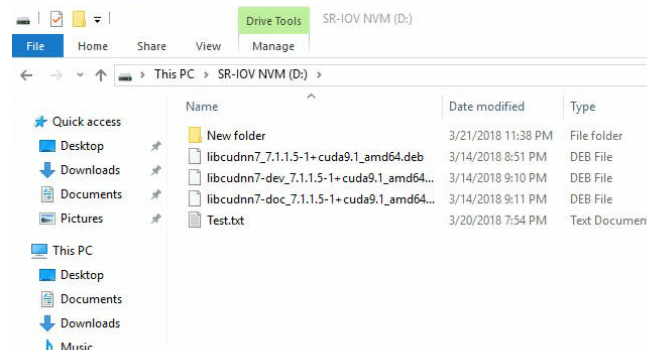


# Demo: Multi-host Sharing of NVMe and GPUs (continued)

## Domain Virtualized as a Spec-compliant PCIe Switch



## NVM VF Appears as a Standard NVM Device







# Demo: Multi-host Sharing of NVMe and GPUs (continued)

## CUDA P2P Bandwidth

```
P2P Connectivity Matrix
```

D\D	0	1	2	3
0	1	1	1	1
1	1	1	1	1
2	1	1	1	1
3	1	1	1	1

```
Unidirectional P2P=Enabled Bandwidth Matrix (GB/s)
```

D\D	0	1	2	3
0	210.61	12.96	12.54	12.53
1	12.52	211.35	13.08	13.06
2	12.52	12.52	212.61	13.05
3	13.06	13.06	12.54	211.36

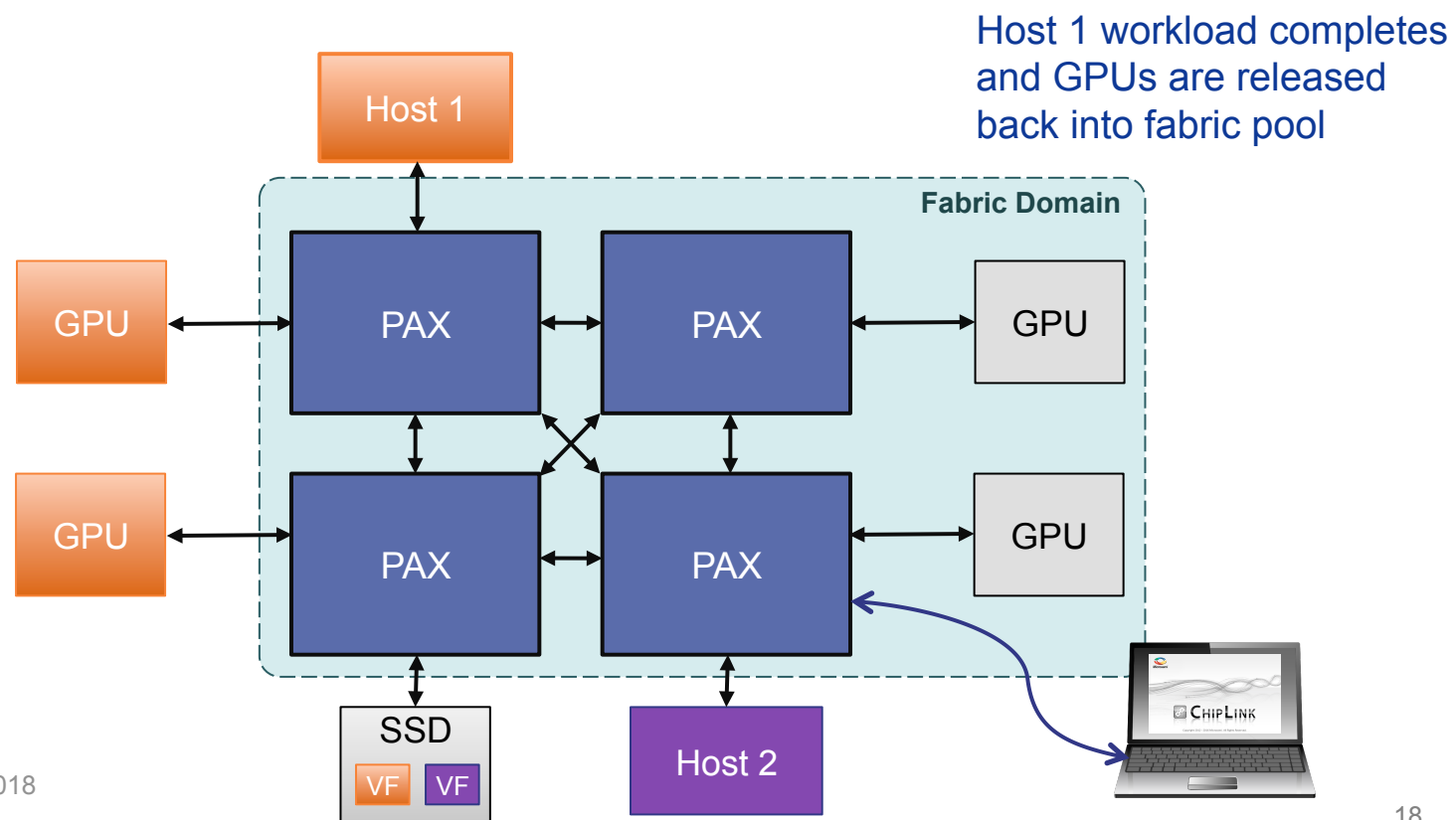
```
Bidirectional P2P=Enabled Bandwidth Matrix (GB/s)
```

D\D	0	1	2	3
0	213.51	24.81	24.77	24.72
1	24.73	213.55	24.73	25.74
2	24.53	24.57	214.73	24.86
3	24.83	25.72	24.73	214.58

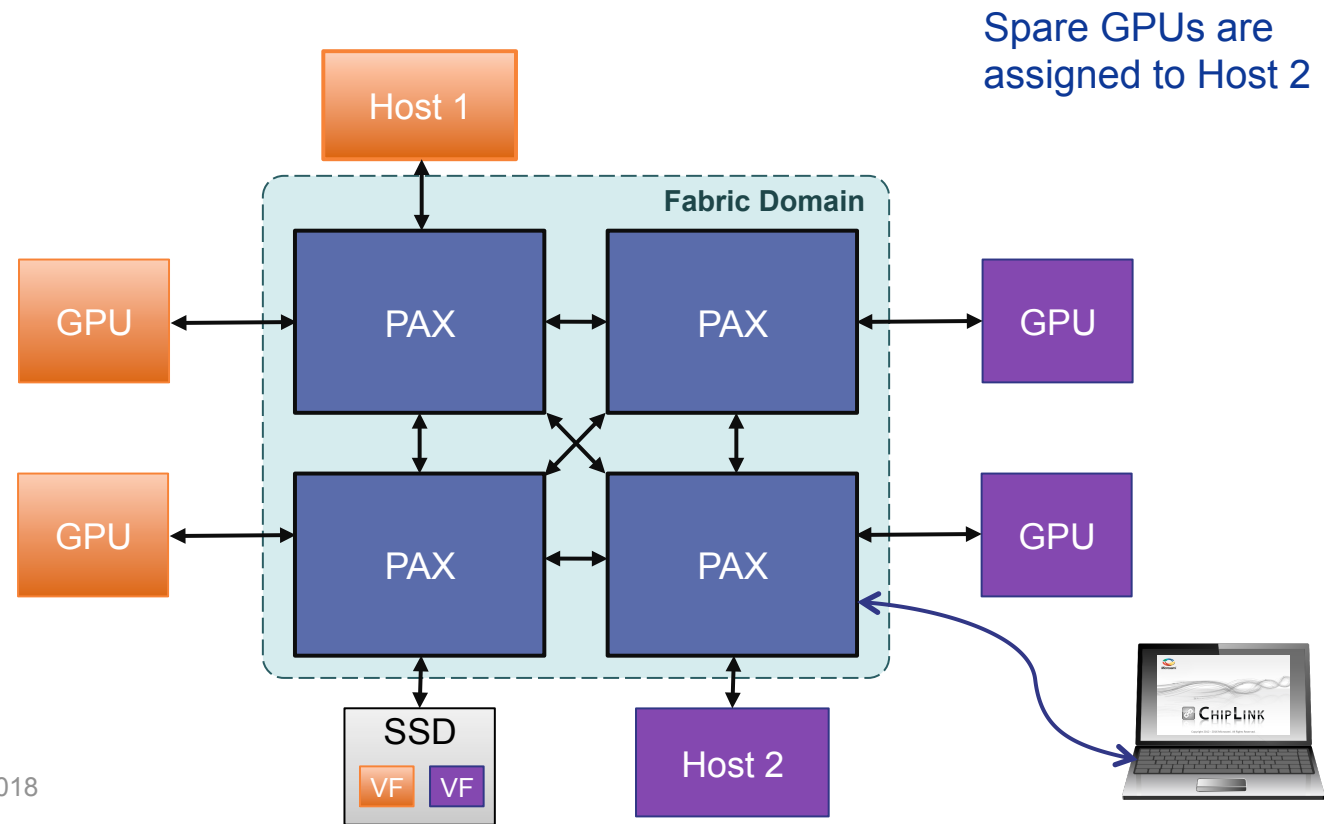
## Running Tensorflow Model

```
Administrator: Command Prompt - python cifar10_multi_gpu_train.py
2018-04-06 18:37:52.698874: I C:\tf_jenkins\home\workspace\rel-win\M\windows-gpu\PY\36\tensorflow\core\common_runtime\gp
u\gpu_device.cc:1030] Found device 3 with properties:
name: Tesla M40 24GB major: 5 minor: 2 memoryClockRate(GHz): 1.112
pciBusID: 0000:07:00.0
totalMemory: 22.43GiB freeMemory: 22.18GiB
2018-04-06 18:37:52.701446: I C:\tf_jenkins\home\workspace\rel-win\M\windows-gpu\PY\36\tensorflow\core\common_runtime\gp
u\gpu_device.cc:1045] Device peer to peer matrix
2018-04-06 18:37:52.702693: I C:\tf_jenkins\home\workspace\rel-win\M\windows-gpu\PY\36\tensorflow\core\common_runtime\gp
u\gpu_device.cc:1051] DMA: 0 1 2 3
2018-04-06 18:37:52.703145: I C:\tf_jenkins\home\workspace\rel-win\M\windows-gpu\PY\36\tensorflow\core\common_runtime\gp
u\gpu_device.cc:1061] 0: Y Y Y Y
2018-04-06 18:37:52.704059: I C:\tf_jenkins\home\workspace\rel-win\M\windows-gpu\PY\36\tensorflow\core\common_runtime\gp
u\gpu_device.cc:1061] 1: Y Y Y Y
2018-04-06 18:37:52.704960: I C:\tf_jenkins\home\workspace\rel-win\M\windows-gpu\PY\36\tensorflow\core\common_runtime\gp
u\gpu_device.cc:1061] 2: Y Y Y Y
2018-04-06 18:37:52.705856: I C:\tf_jenkins\home\workspace\rel-win\M\windows-gpu\PY\36\tensorflow\core\common_runtime\gp
u\gpu_device.cc:1061] 3: Y Y Y Y
2018-04-06 18:37:52.706817: I C:\tf_jenkins\home\workspace\rel-win\M\windows-gpu\PY\36\tensorflow\core\common_runtime\gp
u\gpu_device.cc:1120] Creating TensorFlow device (/device:GPU:0) -> (device: 0, name: Tesla M40, pci bus id: 0000:04:00.
0, compute capability: 5.2)
2018-04-06 18:37:52.707744: I C:\tf_jenkins\home\workspace\rel-win\M\windows-gpu\PY\36\tensorflow\core\common_runtime\gp
u\gpu_device.cc:1120] Creating TensorFlow device (/device:GPU:1) -> (device: 1, name: Tesla M40, pci bus id: 0000:05:00.
0, compute capability: 5.2)
2018-04-06 18:37:52.708655: I C:\tf_jenkins\home\workspace\rel-win\M\windows-gpu\PY\36\tensorflow\core\common_runtime\gp
u\gpu_device.cc:1120] Creating TensorFlow device (/device:GPU:2) -> (device: 2, name: Tesla M40 24GB, pci bus id: 0000:0
6:00.0, compute capability: 5.2)
2018-04-06 18:37:52.709626: I C:\tf_jenkins\home\workspace\rel-win\M\windows-gpu\PY\36\tensorflow\core\common_runtime\gp
u\gpu_device.cc:1120] Creating TensorFlow device (/device:GPU:3) -> (device: 3, name: Tesla M40 24GB, pci bus id: 0000:0
7:00.0, compute capability: 5.2)
```

## Demo: Multi-host Sharing of NVMe and GPUs (continued)



## Demo: Multi-host Sharing of NVMe and GPUs (continued)





# Demo: Multi-host Sharing of NVMe and GPUs (continued)

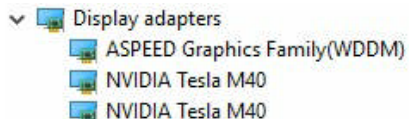
Domain Virtualized as a Spec-compliant PCIe Switch

```
00.0-[03-07]--+00.0-[04]----00.0 NVIDIA Corporation GM200GL [Tesla M40]
+01.0-[05]----00.0 NVIDIA Corporation GM200GL [Tesla M40]
+02.0-[06]----00.0 Samsung Electronics Co Ltd Device a822
\03.0-[07]--
```

NVM VF Appears as a Standard NVM Device

```
ubuntu@bbyapps-ubuntu1604-se:~$ ls /dev
autofs          hidraw1         loop3           nvme0n1p2      sg0            tty2            tty4            tty6            ttyS20         userio
bbyapps-ubuntu1604-se-vg hidraw2         loop4           port           sg1            tty20          tty40          tty60          ttyS21         vcs
block           hidraw3         loop5           ppp            sg2            tty21          tty41          tty61          ttyS22         vcs1
brq             hpet           loop6           psaux          sim            tty22          tty42          tty62          ttyS23         vcs2
brfs-control    iwlwifi        loop7           pax            snapshot       tty23          tty43          tty63          ttyS24         vcs3
bus             iwlwifi        loop-control    pcp0           sasl           tty24          tty44          tty64          ttyS25         vcs4
char            i2c-0          mknod          ptp1           stderr         tty25          tty45          tty8           ttyS26         vcs5
console         i2c-1          ncelog         rtc            stdin          tty26          tty46          tty9           ttyS27         vcs6
core            i2c-2          ncm            random         stdout         tty27          tty47          ttyprintk      ttyS28         vcsa
cuse            i2c-3          network_bandwidth rkill         tty           tty28          tty48          tty30          ttyS29         vcsa1
cpu             i2c-4          nvme            rtc            tty0           tty29          tty49          tty10          ttyS3         vcsa2
cuse            i2c-5          net            rtc0          tty1           tty3           tty5           ttyS10         ttyS30         vcsa3
disk            i2c-6          network_latency sda            tty10          tty30          tty50          ttyS11         ttyS31         vcsa4
dm-0            initctl       network_throughput sda1          tty11          tty31          tty51          ttyS12         ttyS34         vcsa5
dm-1            input         null           sda2          tty12          tty32          tty52          ttyS13         ttyS5         vcsa6
dri             kmsg          nvidia0        sda5          tty13          tty33          tty53          ttyS14         ttyS6         vfi
ecryptfs        kom           nvidia1        sdb            tty14          tty34          tty54          ttyS15         ttyS7         vga_arbiter
fb0             lighthouse    nvidia1        sdb1          tty15          tty35          tty55          ttyS16         ttyS8         vhc1
fa              log           nvidia-vm      sdb2          tty16          tty36          tty56          ttyS17         ttyS9         vhost-net
full            loop0         nvme0          sdb3          tty17          tty37          tty57          ttyS18         uhid         zero
fuse            loop1         nvme0n1        sdc            tty18          tty38          tty58          ttyS19         uinput
hidraw0         loop2         nvme0n1p1      sdc1          tty19          tty39          tty59          ttyS2         urandom
ubuntu@bbyapps-ubuntu1604-se:~$ sudo mount /dev/nvme0n1p2 /mnt
ubuntu@bbyapps-ubuntu1604-se:~$ ls /mnt
libcuda7_7.1.15-1+cuda9.1_amd64.deb  libcuda7-doc_7.1.15-1+cuda9.1_amd64.deb  SRECYCLE.BIN  Test.txt
libcuda7-dev_7.1.15-1+cuda9.1_amd64.deb  New folder  System Volume Information
```

Windows Host Still Running During Dynamic Reassignment





# Demo – Multi-host Sharing of NVMe and GPUs

## CUDA P2P Bandwidth

```
P2P Connectivity Matrix
D\D  0  1
0    1  1
1    1  1
```

```
Unidirectional P2P=Enabled Bandwidth Matrix (GB/s)
D\D  0  1
0 214.21 12.27
1 13.06 212.99
```

```
Bidirectional P2P=Enabled Bandwidth Matrix (GB/s)
D\D  0  1
0 214.53 24.33
1 24.83 215.55
```

- PCIe spec-compliant host domain
- Simple management
- Standard drivers
- Dynamic reassignment appears as spec-compliant surprise-plug



## Summary

Microsemi's Switchtec PAX Switches enable new architectures for next-gen solutions

Benefits of PCIe fabrics with PAX:

- Scalable, low-latency, cost-effective
- Simple Management (PCIe, UART, TWI, Ethernet)
- Multi-host sharing of SR-IOV NVMe devices
- Standard host drivers



a  MICROCHIP company

Live Demo at  
Booth #213