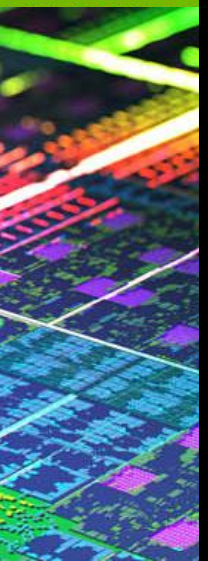
Introduction to Deploying, Managing, and Using GPU Clusters

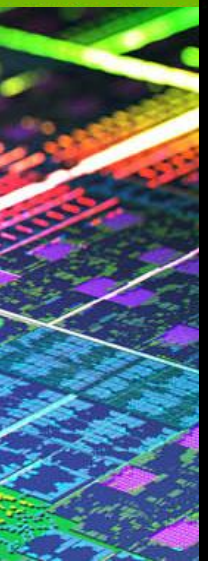
Dale Southard <dsouthard@nvidia.com>

About the Speaker and You

[Dale] is a senior solution architect with NVIDIA (I fix things). I primarily cover large-scale HPC in Gov/Edu/Research and cloud computing. In the past I was a HW architect in the LLNL systems group designing the vis/post-processing solutions.

[You] are here because you are interested in designing, deploying, managing, and using clusters with GPUs for High Performance Computing.





Hardware Selection

Start with the Correct GPU

- Passively Cooled
- Higher Performance
- Chassis/BMC Integration
- Out-of-Band Monitoring

Tesla M-series is Designed for Servers

(C-series GPUs will work, but those target workstation environments, not servers)

Choosing the Correct Server

Historically sites wanted “x16 slots”

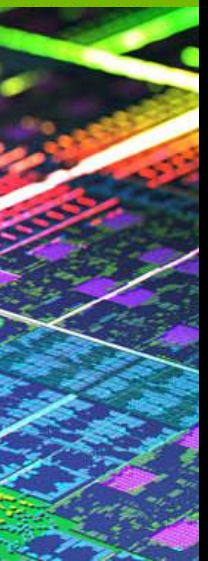
…then we wanted “x16 electrical slots”

…then we wanted “full bandwidth”

…then we wanted “full simultaneous bandwidth”

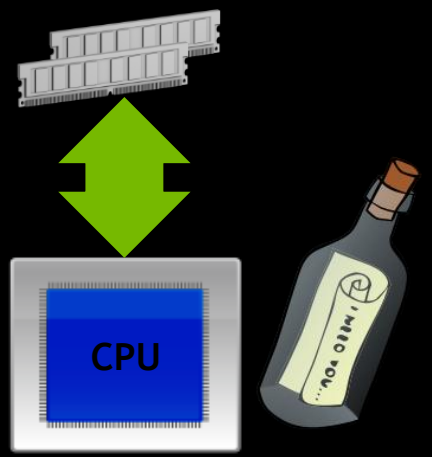
…then we wanted “dual IOH bridges”

Now we will want “peer-to-peer capable”



Doing it Wrong

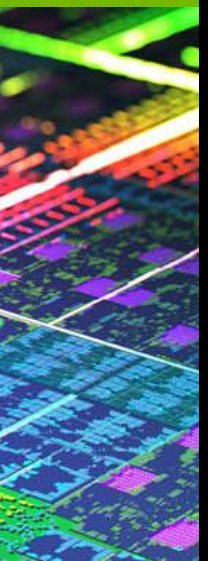
QPI @3.2	12.8 GB/s
HT 3.0 (16b)	10.4 GB/s
PCIe gen2	8 GB/s
PCIe gen3	16 GB/s



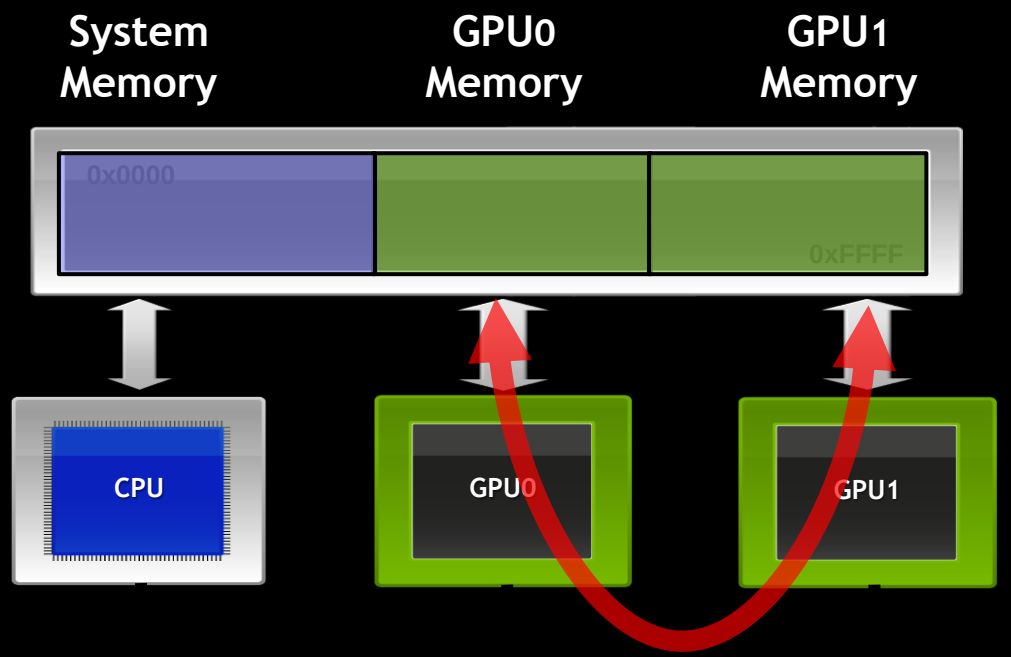
GPUDirect

A long-term effort to eliminate the CPU bottleneck

- Version 1: GPU and NIC shared pinned memory
- Version 2: Peer-to-peer memory access between GPUs
- RDMA: API for access to GPU memory from other devices

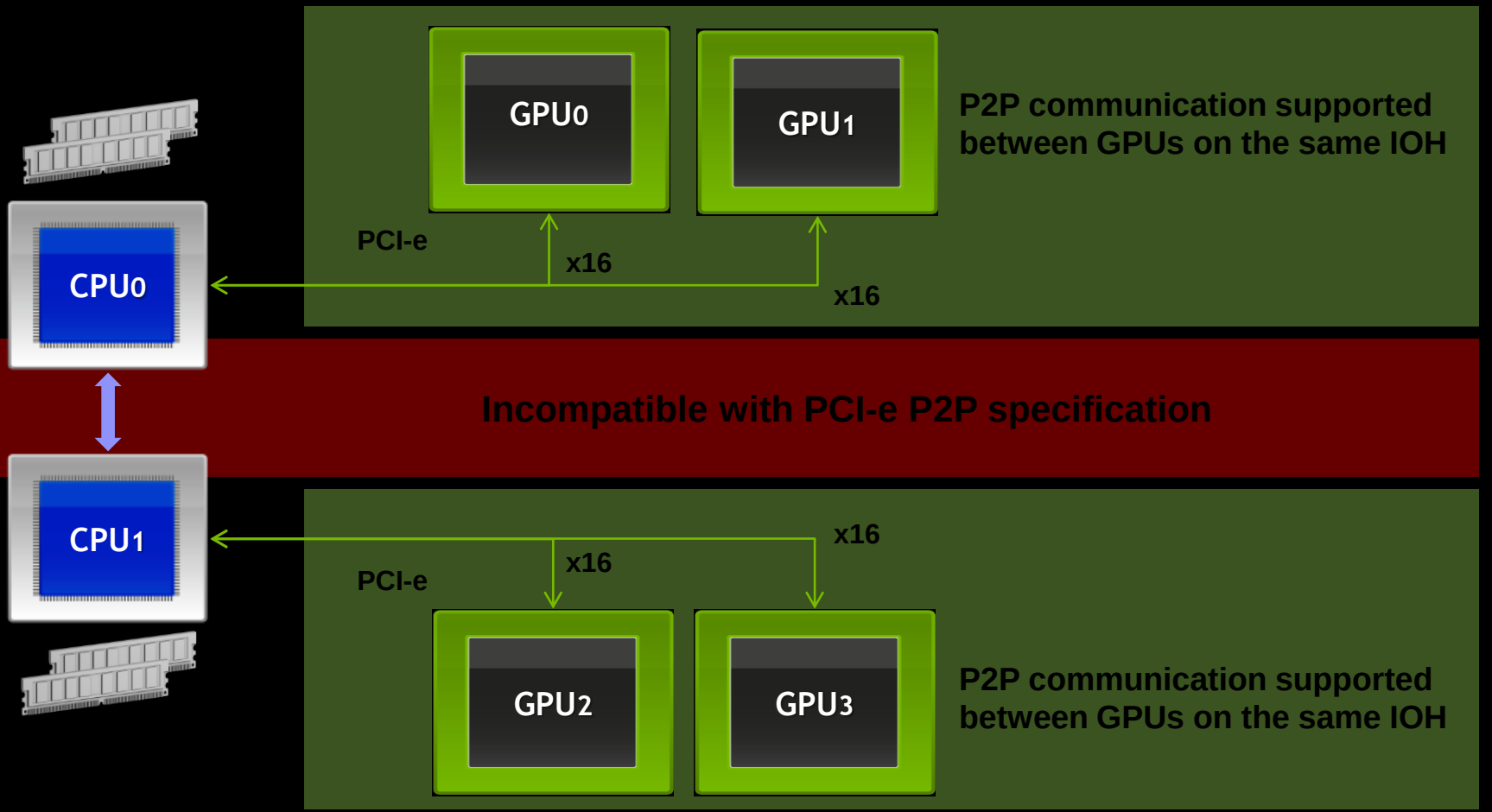


UVA (and RDMA) Drives Server Design

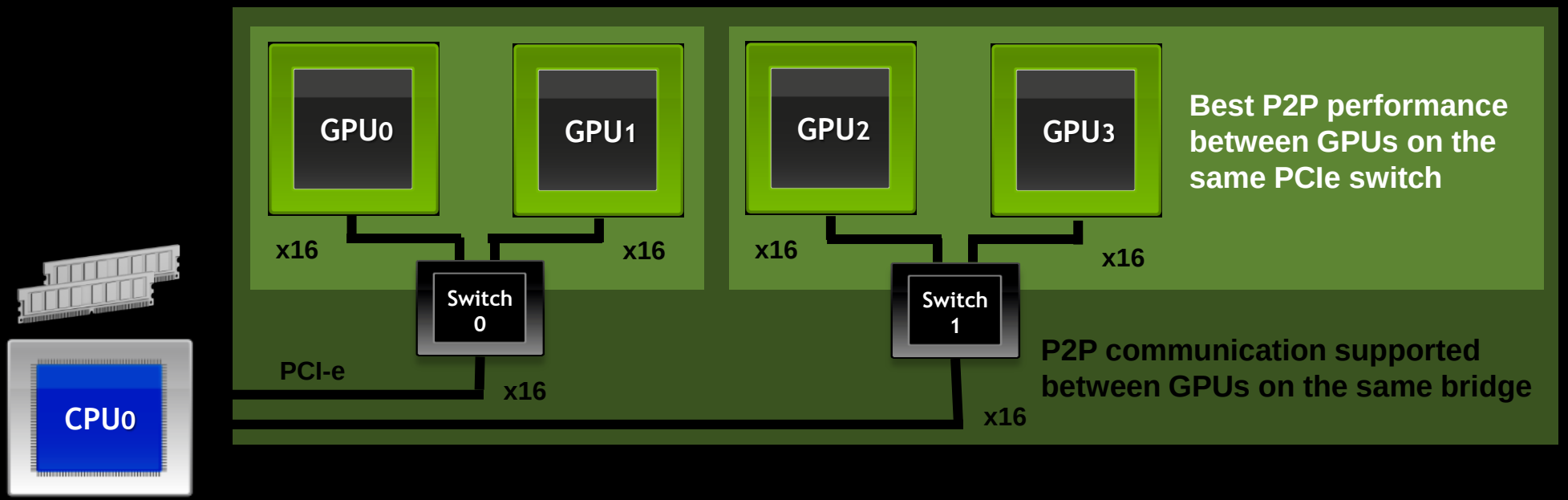


Direct Transfer & Direct Access between GPUs

Topology Matters for P2P Communication

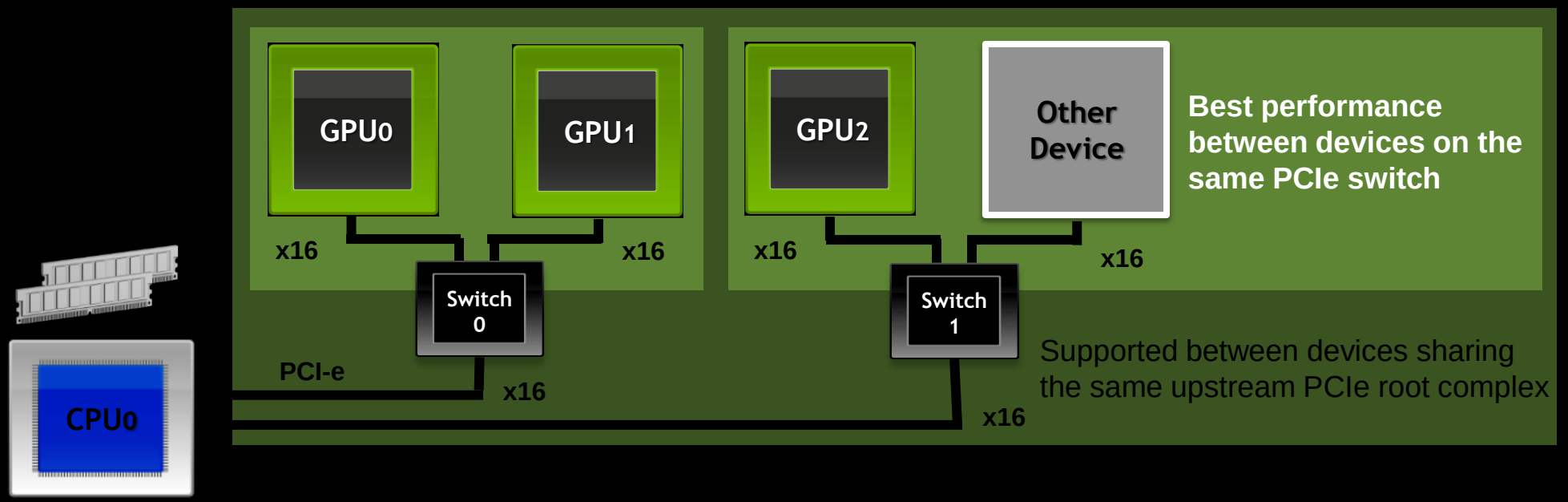


Topology Matters, PCIe Switches



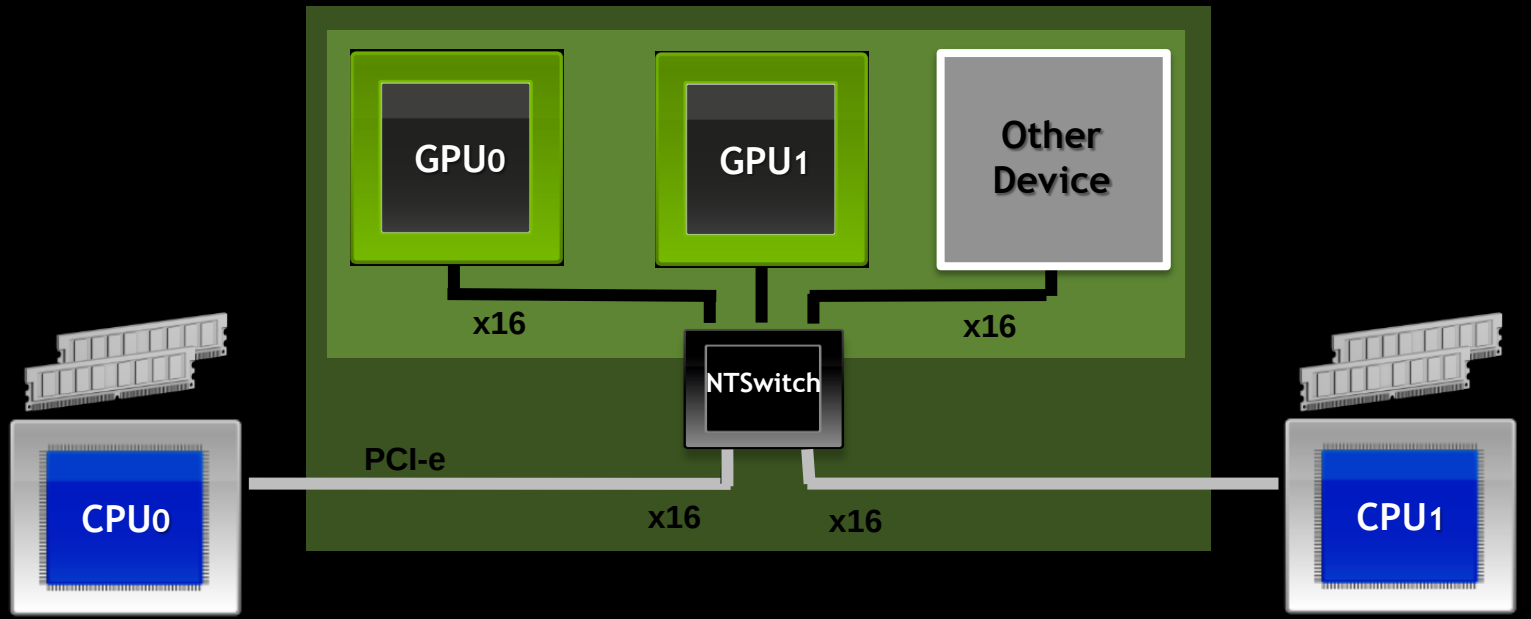
PCI-e Switches: Fully Supported

Topology Matters, GPUDirect RDMA

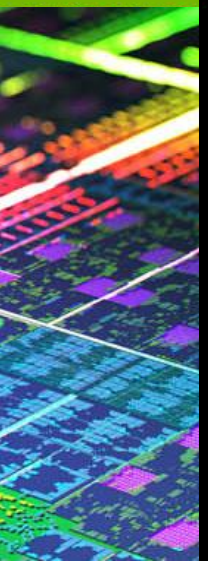


PCI-e Switches: Fully Supported

Non-transparent Switches



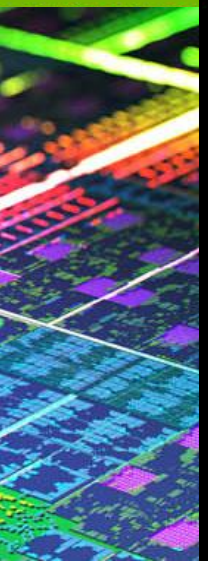
Future Flexibility



Driver Installation and System Configuration

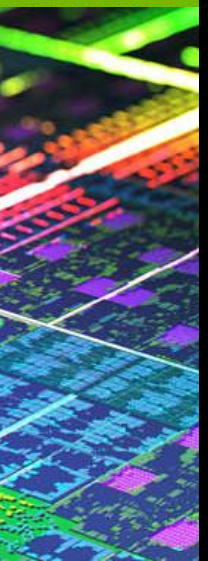
SBIOS Configuration – Cooling Matters

- Tesla M (and X) modules have passive heatsinks
  - Depend on chassis fans for airflow
  - Communicate thermals to BMC
- Some systems require specific fan settings
- Some systems require baffles or blanks or other airflow directors
- During burn-in/acceptance
  - Verify that fans are correctly set
  - Verify that your workload performance is consistent on all nodes



Driver Installation

- NVIDIA provides drivers with a runnable shell script installer
- May want to install libs on login nodes without GPUs
  - sh driver.run --no-kernel-module --no-x-check --no-opengl-files
- Many cluster managers repackage the driver
  - Install into a tree
  - Extract and unpack the tarfile
- If you are also using OpenGL
  - UseDisplayDevice none and a virtual screen
  - Be wary of dual ownership of files with Mesa



Dealing with Runlevel 3

Most clusters operate at runlevel 3 (no xdm), so best practice is to configure the GPUs from an init.d script:

- `modprobe nvidia`
- `mknod` devices
- Assert that ECC is set correctly
- Optionally set compute-exclusive mode
- Optionally set persistence

NVIDIA provides both command-line (`nvidia-smi`) & API (NVML)

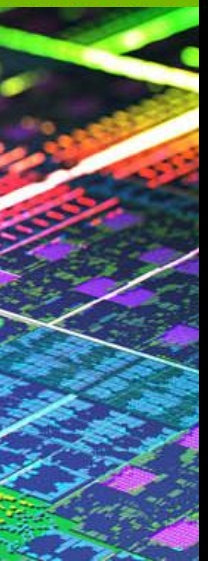
Provisioning Distro'isms

Don't let Nouveau driver load/modeset

- `echo -e 'blacklist nouveau\noptions nouveau modeset=0' >\`
  `/etc/modprobe.d/disable-nouveau.conf`
- May require `dracut` or `update-initramfs`

Ubuntu

- Upstart will respawn display manager,
  use `` `stop gdm` `` or `` `stop lightdm` ``
- maybe from ssh connection in 11.04+

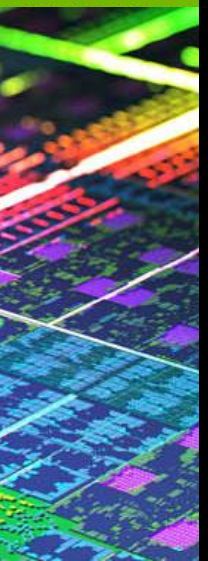


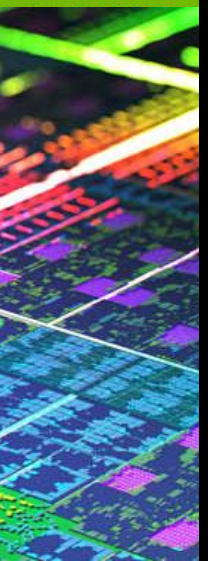
Compute Mode

The Compute Mode setting controls simultaneous use

- `DEFAULT` allow multiple simultaneous processes
- `EXCLUSIVE_THREAD` allows only one context
- `EXCLUSIVE_PROCESS` one process, but multiple threads
- `PROHIBITED`

Can be set by command -line (`nvidia-smi`) & API (NVML)





Persistence Mode

Controls driver unloading

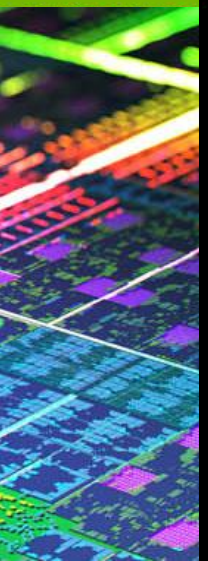
- Persistence mode set
  - Driver does not unload when GPU is idle
  - Slightly lower idle power
  - Faster job startup
- Persistence mode not set
  - If ECC is on, memory is cleared between jobs

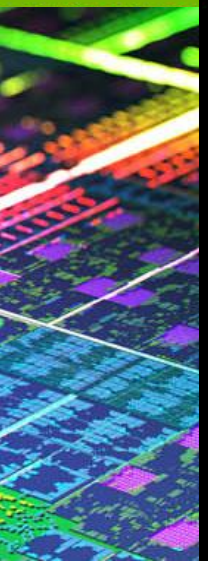
Can be set by command-line (`nvidia-smi`) & API (NVML)

GPU Operation Mode (GOM)

- Disabling graphics features to optimize for compute
  - Support on Kepler GK110 based K20/K20X (not on C-Class)
- Modes:
  - All On – All features are on (including graphics)
  - Compute – Graphics feature off, only compute tasks can run
  - Low Double Precision – Graphics are on, but reduced FP64 features

Persistent, but currently requires reboot to change modes





Limiting Clocks and Power

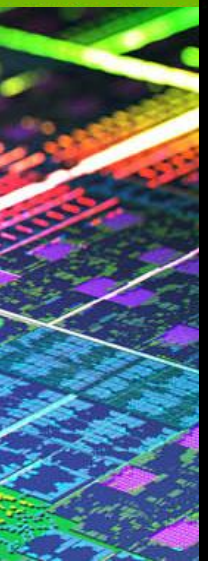
- Kepler supports “locking” clocks or power to lower default
  - Clock control exposed in nvidia-smi
  - Setting application clocks sets the default performance state
  - Setting power limit sets the point capping maximum
- Not all combinations supported -- nvidia-smi –q –d SUPPORTED_CLOCKS
- Power management and thermal brake are still in effect

Not Persistent – clocks reset when driver unloads

Power and the Green500

- Kepler has ~40% advantage over Intel Phi in Perf/W
  - But Beacon (Phi) edged out Titan (Kepler) 2.5 GF/W vs 2.1 GF/W
- Tuning for Green500
  - Optimize the CPU:Accelrator ratio (Beacon was 2:4, Titan 1:1)
  - If you don't need it, turn it off (disable USB, hotplug cores, etc)
  - Smaller/Flatter is better
  - Colder is better
  - Measure accurately

There's no magic to the Green500 – just lots of details



Job Scheduling

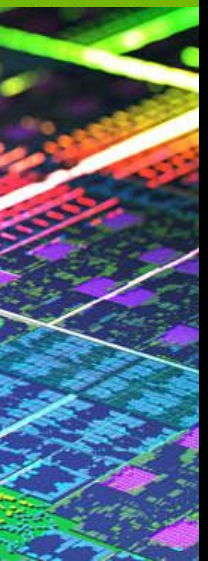
For time-sharing a node use $CUDA_VISIBLE_DEVICES:

```
$ ./deviceQuery -noprompt | egrep "^Device"
 Device 0: "Tesla C2050"
 Device 1: "Tesla C1060"
 Device 2: "Quadro FX 3800"

$ export CUDA_VISIBLE_DEVICES="0,2"

$ ./deviceQuery -noprompt | egrep "^Device"
 Device 0: "Tesla C2050"
 Device 1: "Quadro FX 3800"
```

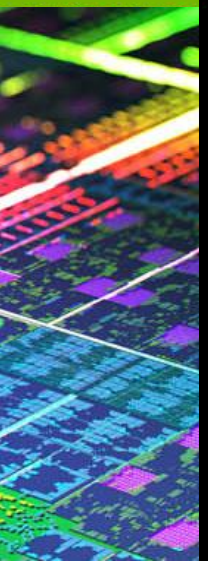
Several batch systems and resource mangers support GPUs as independent consumables using $CUDA_VISIBLE_DEVICES



Resource Limits

- UVA depends on allocating virtual address space
- Virtual address space != physical ram consumption
- Use cgroups, not ulimit

Several batch systems and resource mangers support cgroups directly or via plugins.

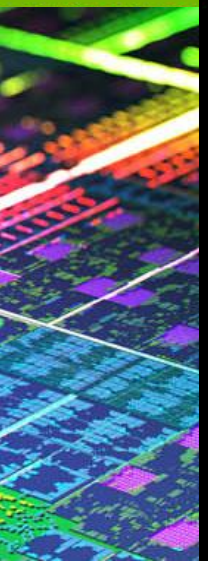


GPU Management

GPU Management: Acceptance and Burn-in

- After cluster is installed and configured, verify functionality!
  - The best acceptance test is the intended workload
  - As an alternative, use healthmon, BandwidthTest, or other samples
- Common issues and their symptoms:
  - Configuration errors (inconsistent performance between nodes)
  - Cooling problems (inconsistent performance between runs)
  - Seating/Chipset performance (slow GPUs, slow transfers)

Acceptance testing is easier with good monitoring framework

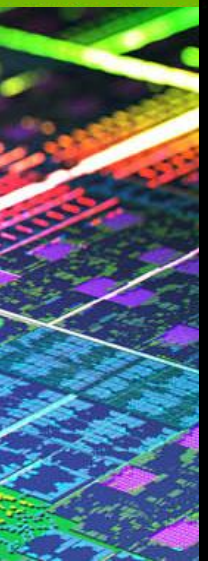


GPU Management: Finding Problems

- Monitoring though nvidia-smi or NVML
- Watching for XID errors in syslog
- PCIe parity via EDAC

```
modprobe   edac_mc
echo 1 > /sys/devices/system/edac/pci/check_pci_parity
```

- User feedback and testing

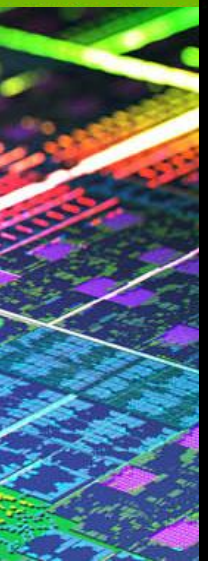


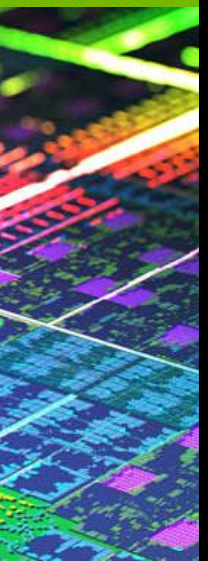
GPU Management: Monitoring

- Environmental and utilization metrics are available
- Tesla M-series may provide OoB access to telemetry via BMC
- NVIDIA provides command line (nvidia-smi) and Library (NVML)
  - NVML can be used from C or Python or Perl
  - NMVL has been integrated into Ganglia gmond

http://developer.nvidia.com/nvidia-management-library-nvml

https://developer.nvidia.com/tesla-deployment-kit





GPU Management: XID Errors

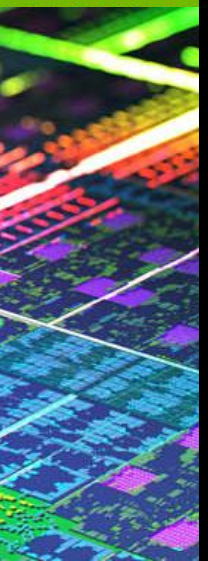
- XID errors are reported to syslog via NVRM
- May indicate a HW error or programming error
- Common non-HW causes of XIDs:
  - Out-of-bounds memory access causing XID 13
  - Illegal access causing XID 31
  - Termination of program causing XID 45

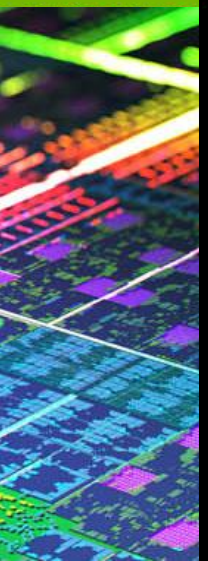
CUDA-gdb or memcheck can often provide addition info

Handling Bad Devices

- Three different enumeration systems:
  - PCIe
  - CUDA runtime
  - nvidia-smi
- Do not assume that the three enumerations are consistent!
- PCIe device ID, serial number, and UUID are consitent

Always have operators check serial number of pulled HW



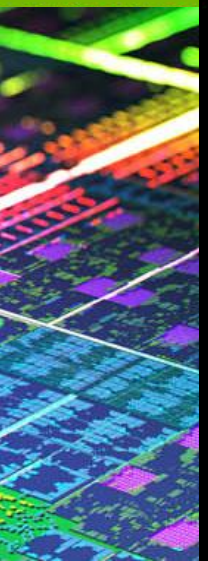


GPU Management: Error Containment

Tesla 20-series:

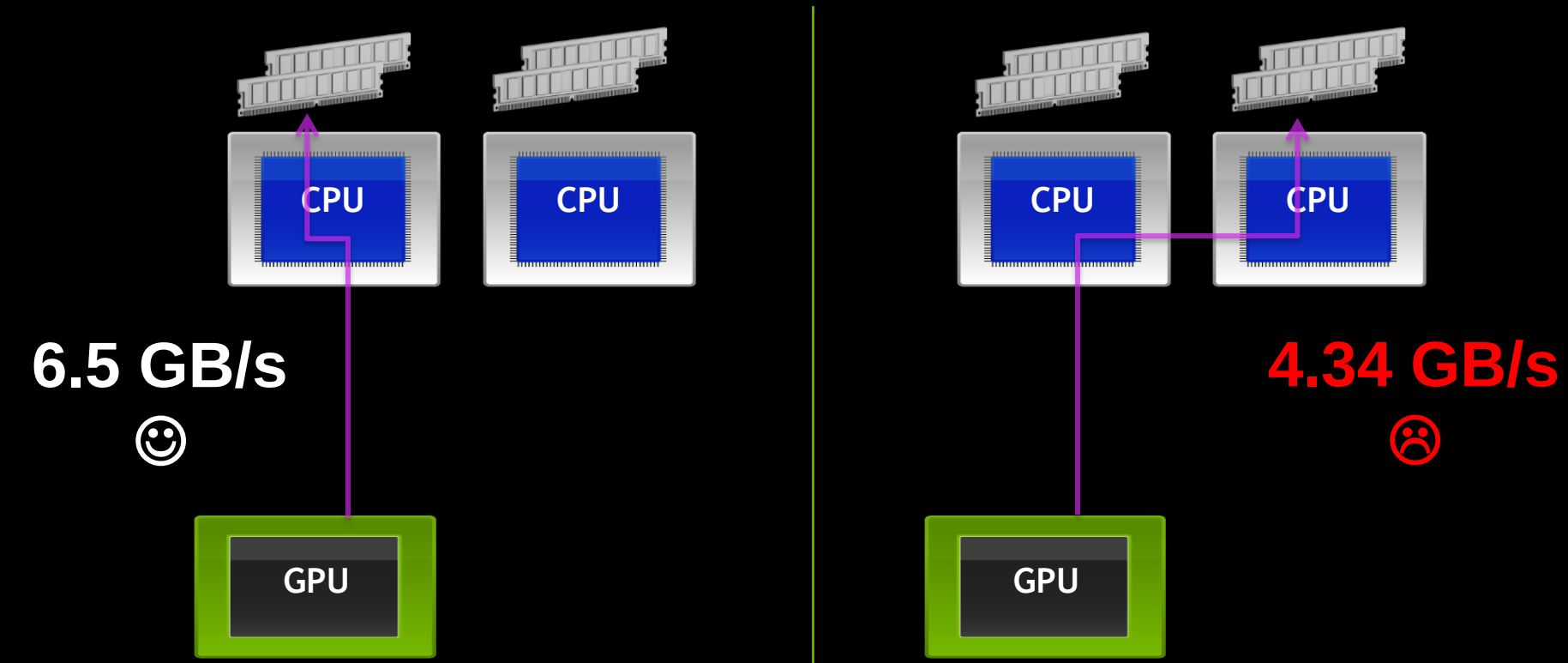
- ECC Protection on DRAM and on-chip memories
- ECC can be turned on/off via `nvidia-smi` or NVML (reboot)
- Volatile and Persistent ECC counters
- Counts available via `nvidia-smi`, NVML, and sometimes SMBus
- Correctable errors are logged but not scrubbed
- Uncorrectable errors cause user and XID system error
- Uncorrectable errors cause GPU to reject work until reboot
- Uncorrectable errors do not cause MCE or reboot

Full ECC containment without requiring system halt



GPU Utilization

GPU Utilization: Topology Matters (again)

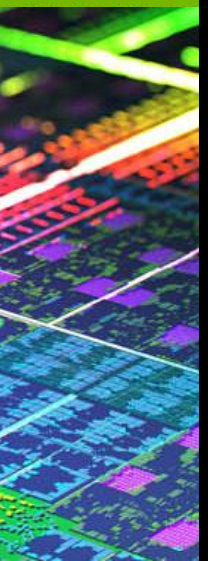


MPI rank (process) mapping impacts performance

GPU Utilization: Detecting Local Rank

Easy on OpenMPI and MVAPICH, modular arithmetic everywhere else

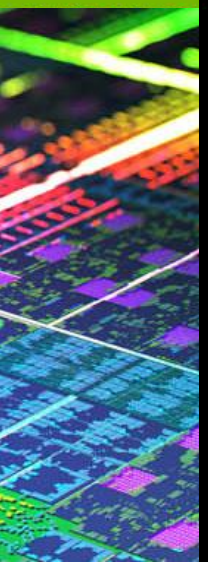
```
if [[ -n ${OMPI_COMM_WORLD_LOCAL_RANK} ]]
  then
  lrank=${OMPI_COMM_WORLD_LOCAL_RANK}
elif [[ -n ${MV2_COMM_WORLD_LOCAL_RANK} ]]
  then
  lrank= ${MV2_COMM_WORLD_LOCAL_RANK}
elif [[ -n ${PMI_ID} && -n ${MPISPAWN_LOCAL_NPROCS} ]]
  then
  let lrank=${PMI_ID}%${PERHOST}
else
  echo could not determine local rank
fi
```

GPU Utilization: NUMA Placement

Once you have the local rank, use it to bind the process to NUMA

```
case ${lrank} in
[0-1])
  numactl --cpunodebind=0 ${@}
  ;;
[2-3])
  numactl --cpunodebind=2 ${@}
  ;;
[4-5])
  numactl --cpunodebind=3 ${@}
  ;;
esac
```

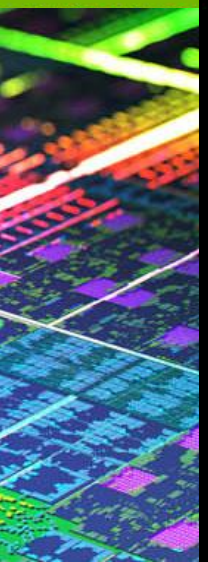


GPU Utilization: Wrapping for NUMA

Glue the previous two fragments into a wrapper script (`numawrap`)

- Use as `mpirun numawrap rest -of -command`
- The `${@}` will pick up the command at runtime
- Only need to figure out the topology once
- May require MV2_USE_AFFINITY=0 ; MV2_ENABLE_AFFINITY=0

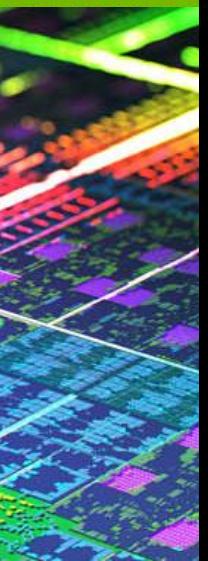
Eventually hwloc may allow MPI to automate the placement

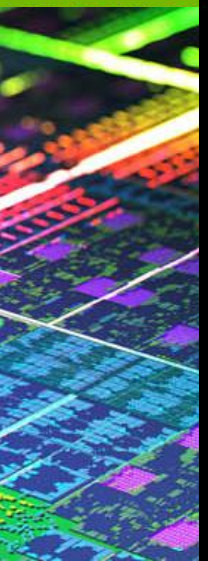


GPU Management: CUDA Proxy

- Proxy facilitates GPU sharing among processes or MPI ranks
  - CUDA kernels run under a single context
  - Eliminates timeouts
  - Increases concurrency via improved use of Hyper-Q
- CUDA Proxy is a daemon and control program
  - Controlled with nvidia-cuda-proxy-control

Currently, use of proxy will confuse most debuggers





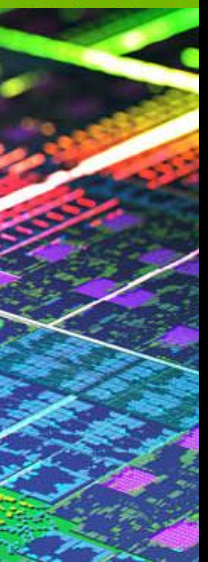
Summary

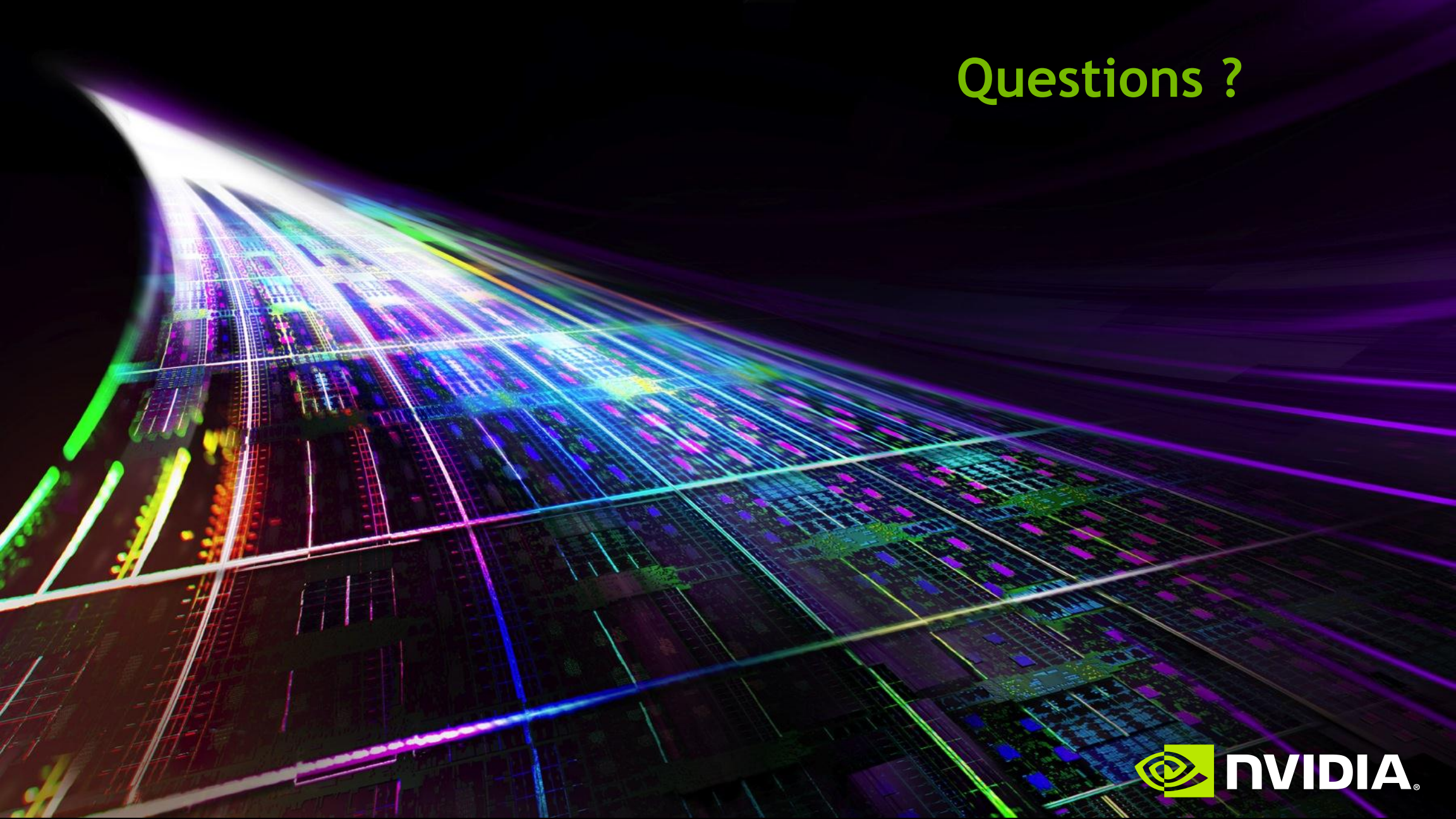
Things to Take Home

- Pick the correct HW
- Topology Matters
- Lots of hooks for management
- Topology Matters

Want more?

S3516 - Building Your Own GPU Research Cluster Using Open Source Software Stack




Questions ?
NVIDIA