

# SYSTEM OVERVIEW: OLCF SUMMIT

**MOCHI BOOTCAMP**  
ECP Annual Meeting

# SUMMIT

- 200 PF [HPL I presume]
- 4608 Nodes
- 2 POWER9 CPU per node (9,216 total)
- 6 NVIDIA Volta per node (27,648 total)
- Mellanox EDR Infiniband
- 1600 GB NVRAM per node
- 250 PB GPFS, 2.5 TB/sec advertised bandwidth



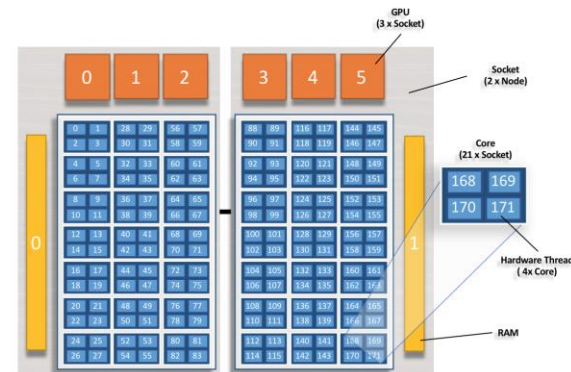
# JOB SUBMISSION

- Three kinds of nodes on summit:
  - Login: everyone starts here
  - Launch: service nodes. Shared among all users (play nice)
  - Compute: where all the fun happens. Users request an allocation of compute nodes from the LSF scheduler (via **bsub**) and run commands on those nodes with **jsrun**
- Check job with **bjobs**
- Interactive example:

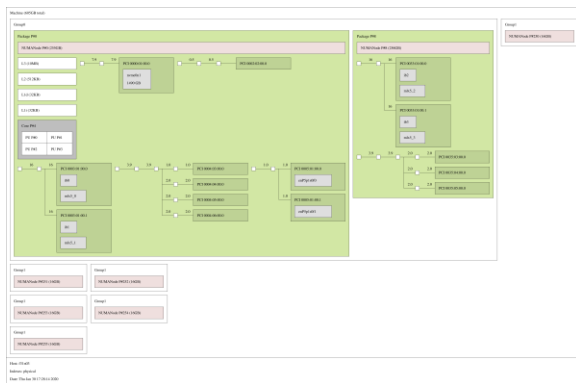
```
[robl@login1]% bsub -Is -W 0:15 -nnodes 2 -P CSC332 $SHELL
Job <843986> is submitted to default queue <batch>.
<<Waiting for dispatch ...>>
<<Starting on batch4>>
[\u@batch4.summit \W]\$ jsrun -r 1 hostname
b29n05
b29n04
```

# RESOURCE SETS

- Lots of ways to carve up nodes allocated with 'bsub'
- "Resource Sets" describe how many cores/threads per node to use
  - e.g. "One GPU per task" vs
  - "I will manage GPU and CPU resources myself"
- Our Mochi services will run with one resource set per node
  - `bsub -nnodes=2...`
  - `jsrun -n 1 -a 1 -c ALL_CPUS -g ALL_GPUS`
  - Your own code might demand other layout.



# RESOURCE SET DEMONSTRATION



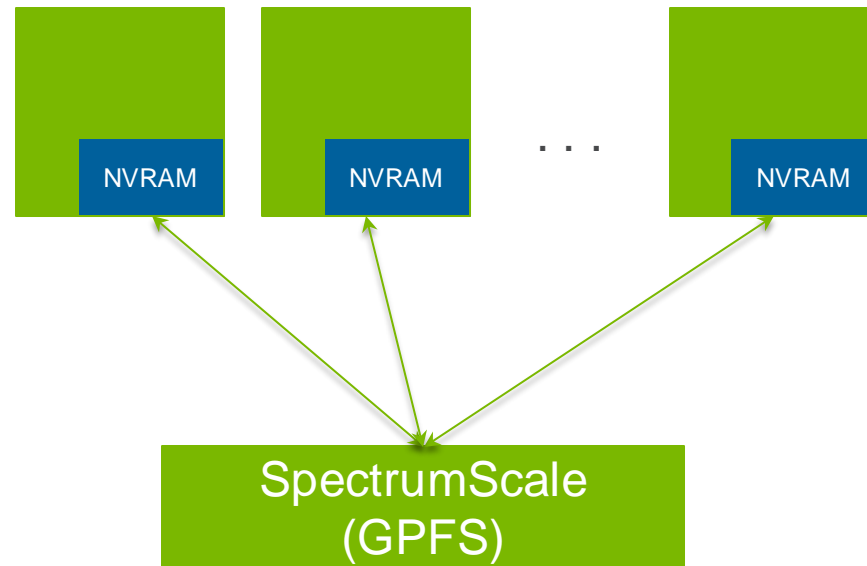
```
jsrun -n1 -r1 lstopo
```



```
jsrun -n 1 -a 1 -c ALL_CPUS -g ALL_GPUS
```

# SUMMIT STORAGE

- 1.6 TB NVRAM
  - /mnt/bb/\$USER
  - Request with `-alloc_flags NVME`
  - Only supports file-per-process or file-per-node
  - ‘Spectral’ library can help with stage-in and stage-out
- 250PB SpectrumScale (GPFS)
  - /gpfs/alpine/\$PROJECT/scratch/\$USER
  - Usual GPFS behavior
    - File per process will take “forever”
    - Shared file writes require 16 MiB block alignment



# SPACK ON SUMMIT

- Just highlighting Summit-specific information: hopefully you already did the homework!
- Review <https://xgitlab.cels.anl.gov/sds/mochi-boot-camp/blob/master/ecp-am-2020/sessions/hands-on/README.md> : anything stump you?
- Libfabric: use verbs variant, not ucx to drive infiniband
- Almost everything you need is available as a module.
- .. but there's a bug with cmake, so spack has to build that one itself

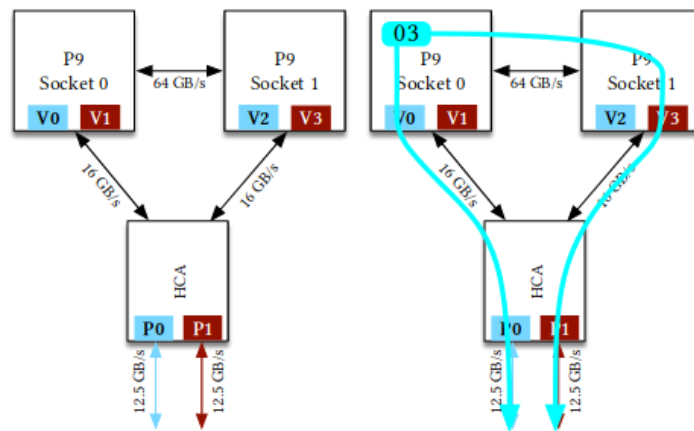
```
libfabric:  
  variants: fabrics=verbs,rxm,mrml
```

```
automake:  
  modules:  
    automake@1.16.1: automake/1.16.1  
  buildable: False
```

# SUMMIT NETWORKING

## The “Multi-rail” bonus exercise

- 2 CPU per node, 1 network card (HCA)
- Each node sees 4 Infiniband ports
- Careful selection will result in one CPU driving both HCA ports
- Otherwise, will only see about half of advertised performance
- See if you can find the right value/config for FI\_OFI\_MRAIL\_ADDR



**Figure 2: The left image shows the mapping of virtual to physical ports for both sockets. The right image shows socket 0 striping data over virtual ports V0 and V3.**

Image from Zimmer, Atchley, et. al “An evaluation of the CORAL interconnects”, SC2019



# RESOURCES

- [https://docs.olcf.ornl.gov/systems/summit\\_user\\_guide.html](https://docs.olcf.ornl.gov/systems/summit_user_guide.html)
  - One-stop survey of just about anything you need
- <https://dl.acm.org/doi/10.1145/3295500.3356166>
  - In-depth study of Summit (and Sierra) networking
- [https://ofiwg.github.io/libfabric/master/man/fi\\_mrail.7.html](https://ofiwg.github.io/libfabric/master/man/fi_mrail.7.html)
  - Libfabric documentation for the “multi-rail” protocol
- [https://www.olcf.ornl.gov/wp-content/uploads/2018/12/spectrum\\_scale\\_summit\\_workshop.pdf](https://www.olcf.ornl.gov/wp-content/uploads/2018/12/spectrum_scale_summit_workshop.pdf)
  - Lots of Spectrum-Scale (GPFS) tuning parameters