OAK RIDGE National Laboratory

Spock Tips & Information

Tom Papatheodore

Oak Ridge Leadership Computing Facility

Spock Training – May 20, 2021

Where to Find Helpful Environment Variables

`man` pages for `srun`, `salloc`, `sbatch`

- Input and output environment variables can be found at bottom of each page
- E.g., `SLURM_JOB_ID, SLURM_LOCALID, SLURM_JOB_NODELIST`

`man intro_mpi`

- MPI environment variables
- E.g,. `MPICH_GPU_SUPPORT_ENABLED, MPICH_GPU_IPC_THRESHOLD`

`module show <package_name>`

- Shows how loading a package will change your environment

```
$ module show boost
-------------------------------------------------------------------------------
   /sw/spock/spack-envs/base/modules/spack/linux-sles15-x86_64/Core/boost/1.73.0.lua:
-------------------------------------------------------------------------------
whatis("Name : boost")
whatis("Version : 1.73.0")
whatis("Target : x86_64")
whatis("Short description : Boost provides free peer-reviewed portable C++ source libraries, emphasizing libraries that work well with the C++ Standard Library.")
help([[Boost provides free peer-reviewed portable C++ source libraries,
emphasizing libraries that work well with the C++ Standard Library.
Boost libraries are intended to be widely useful, and usable across a
broad spectrum of applications. The Boost license encourages both
commercial and non-commercial use.]])
prepend_path("CMAKE_PREFIX_PATH","/sw/spock/spack-envs/base/opt/linux-sles15-x86_64/gcc-7.5.0/boost-1.73.0-nnaww46oatokeudktwfaxu6dvl7zsyhw/")
setenv("BOOST_ROOT","/sw/spock/spack-envs/base/opt/linux-sles15-x86_64/gcc-7.5.0/boost-1.73.0-nnaww46oatokeudktwfaxu6dvl7zsyhw")
setenv("OLCF_BOOST_ROOT","/sw/spock/spack-envs/base/opt/linux-sles15-x86_64/gcc-7.5.0/boost-1.73.0-nnaww46oatokeudktwfaxu6dvl7zsyhw")
```

Where to Find Helpful Environment Variables

```
$ man intro_
intro_asm_intrin      intro_cblas           intro_directives      intro_intrin          intro_lapacke         intro_openmp
intro_pmi             intro_scalapack
intro_blacs           intro_coarray         intro_dsmml           intro_intrinsics      intro_libm            intro_papi
intro_pragmas         intro_specialibs
intro_blas1           intro_conversion      intro_ffio            intro_io              intro_libsci          intro_PAPI
intro_pxf             intro_timing
intro_blas2           intro_craypat         intro_hugepages       intro_irt             intro_mpi             intro_perftools
intro_quad            intro_upc
intro_blas3           intro_craype-api      intro_ieee            intro_lapack          intro_openacc         intro_pgas
intro_quad_precision
```

Helpful Tools

Users might find the following tools helpful…

`rocm-smi`

- AMD ROCm System Management Interface
- Find useful information about AMD GPUs (e.g., running processes, memory usage, temperature, driver version)
- `rocm-smi --help`

`rocminfo`

- ROCm Application for Reporting System Info
- Find useful system information (CPUs and GPUs)
- `rocminfo --help`

https://code.ornl.gov/olcf/hello_mpi_omp

- MPI + OpenMP Hello World-type program for process and thread placement

https://code.ornl.gov/olcf/hello_jobstep

- MPI + OpenMP + HIP Hello World-type program for process, thread, and GPU mapping

Debugging Issues on a Compute Node

You can `ssh` to compute nodes that are allocated to your job.

`squeue` (or other ways) will show the list of compute nodes for a job.

`ssh <spock_node_id>`

From there you can start troubleshooting. E.g., using `rocm-smi` to look at memory usage, processes on the GPUs, or `gstack <pid>` for CPU processes.

Terminal 1

```
$ salloc -A stf016 -t 30 -N 1
salloc: Pending job allocation 261090
salloc: job 261090 queued and waiting for resources
salloc: job 261090 has been allocated resources
salloc: Granted job allocation 261090
salloc: Waiting for resource configuration
salloc: Nodes spock13 are ready for job
```

Terminal 2

```
$ ssh spock13

$ hostname
spock13

$ rocm-smi --showpids
======================== ROCm System Management Interface ========================
================================== KFD Processes ==================================
KFD process information:
PID          PROCESS NAME GPU(s)       VRAM USED    SDMA USED    CU OCCUPANCY
70605        p2p          2            2155888640   0            0
70604        p2p          2            2155888640   0            0
============================== End of ROCm SMI Log ===============================

$ rocm-smi --showmemuse
======================== ROCm System Management Interface ========================
================================== Current Memory Use ============================
GPU[0]                   : GPU memory use (%): 5
GPU[1]                   : GPU memory use (%): 0
GPU[2]                   : GPU memory use (%): 0
GPU[3]                   : GPU memory use (%): 5
============================== End of ROCm SMI Log ===============================

$ top # Use to find PID

$ gstack <PID>
```

Custom Slurm Commands

You can customize the output of e.g,. `squeue, sinfo, sacct` to your preference.

```
$ squeue -o "%7A | %10P | %10u | %10a | %20j | %11l | %9M | %6D | %3t | %8Q | %30R”

JOBID   | PARTITION  | USER       | ACCOUNT    | NAME                | TIME_LIMIT  | TIME      | NODES  | ST  | PRIORITY | NODELIST(REASON)
261090  | batch      | tpapathe   | stf016     | interactive         | 1:00:00     | 27:42     | 1      | R   | 2300     | spock13
```

```
$ sacct --format=User,JobID,Jobname,partition,state,time,start,elapsed,nnodes,ncpus,nodelist,Priority -j 261090

     User        JobID    JobName  Partition      State  Timelimit               Start    Elapsed   NNodes      NCPUS        NodeList   Priority
--------- ------------ ---------- ---------- ---------- ---------- ------------------- ---------- -------- ---------- --------------- ----------
 tpapathe 261090       interacti+      batch    RUNNING   01:00:00 2021-05-19T17:25:27   00:35:02        1        128          spock13         20
          261090.inte+ interacti+                RUNNING            2021-05-19T17:25:27   00:35:02        1         64          spock13
          261090.exte+     extern                RUNNING            2021-05-19T17:25:27   00:35:02        1        128          spock13
          261090.0            p2p              COMPLETED            2021-05-19T17:31:54   00:00:53        1         64          spock13
          261090.1            p2p              COMPLETED            2021-05-19T17:32:56   00:00:52        1         64          spock13
          261090.2            p2p              COMPLETED            2021-05-19T17:33:56   00:00:52        1         64          spock13
```

```
$ sinfo --format="%10P | %6a | %6D | %10T | %9s | %30N" -p caar

PARTITION  | AVAIL  | NODES  | STATE      | JOB_SIZE  | NODELIST
caar       | up     | 5      | allocated  | 1-16      | spock[01-02,13,25-26]
caar       | up     | 31     | idle       | 1-16      | spock[03-12,14-24,27-36]
```

Capture Job Information

```
$ cat submit.sh
#!/bin/bash
#SBATCH -A STF016
#SBATCH -J test
#SBATCH -o %x-%j.out
#SBATCH -t 00:05:00
#SBATCH -p batch
#SBATCH -N 1

scontrol show job ${SLURM_JOBID}

srun -n1 ./p2p --correct

sacct -j ${SLURM_JOBID} -o jobid%20,Start%20,elapsed%20
```

```
$ cat test-261226.out
JobId=261226 JobName=test
   UserId=tpapathe(5987) GroupId=tpapathe(8654) MCS_label=N/A
   Priority=594 Nice=0 Account=stf016 QOS=normal
   JobState=RUNNING Reason=None Dependency=(null)
   Requeue=1 Restarts=0 BatchFlag=1 Reboot=0 ExitCode=0:0
   RunTime=00:00:00 TimeLimit=00:05:00 TimeMin=N/A
   SubmitTime=2021-05-19T20:45:43 EligibleTime=2021-05-19T20:45:43
   AccrueTime=2021-05-19T20:45:43
   StartTime=2021-05-19T20:58:17 EndTime=2021-05-19T21:03:17 Deadline=N/A
   SuspendTime=None SecsPreSuspend=0 LastSchedEval=2021-05-19T20:58:17
   Partition=batch AllocNode:Sid=login1:24709
   ReqNodeList=(null) ExcNodeList=(null)
   NodeList=spock25
   BatchHost=spock25
   NumNodes=1 NumCPUs=128 NumTasks=1 CPUs/Task=1 ReqB:S:C:T=0:0:*:1
   TRES=cpu=128,node=1,billing=128
   Socks/Node=* NtasksPerN:B:S:C=0:0:*:* CoreSpec=*
   MinCPUsNode=1 MinMemoryNode=0 MinTmpDiskNode=0
   Features=(null) DelayBoot=00:00:00
   OverSubscribe=NO Contiguous=0 Licenses=(null) Network=(null)
   Command=/autofs/nccs-svm1_home1/tpapathe/systems/spock/scripts/slurm_scripts/submit.sh
   WorkDir=/autofs/nccs-svm1_home1/tpapathe/systems/spock/scripts/slurm_scripts
   StdErr=/autofs/nccs-svm1_home1/tpapathe/systems/spock/scripts/slurm_scripts/test-261226.out
   StdIn=/dev/null
   StdOut=/autofs/nccs-svm1_home1/tpapathe/systems/spock/scripts/slurm_scripts/test-261226.out
   Power=
   NtasksPerTRES:0

...<APPLICATION OUTPUT>...

 JobID               Start             Elapsed
------------------- ------------------- -------------------
              261226  2021-05-19T20:58:17             00:00:42
        261226.batch  2021-05-19T20:58:17             00:00:42
       261226.extern  2021-05-19T20:58:17             00:00:42
            261226.0  2021-05-19T20:58:17             00:00:42
```

Add other information as needed.

- e.g., `module -t list`

Show Information About Completed Jobs

Specify a start (`-S`) and end (`-E`) time to search for completed jobs.

- The default time window depends on other options (see `man sacct`)

```
$ sacct --user=tpapathe -S 2021-05-18T23:00 -E 2021-05-18T23:45
       JobID    JobName  Partition    Account  AllocCPUS      State ExitCode
------------ ---------- ---------- ---------- ---------- ---------- --------
259409         scontrol      batch     stf016        128  COMPLETED      0:0
259409.exte+     extern                 stf016        128  COMPLETED      0:0
259409.0       scontrol                 stf016         64  COMPLETED      0:0
259426             test      batch     stf016        128  COMPLETED      0:0
259426.batch      batch                 stf016         64  COMPLETED      0:0
259426.exte+     extern                 stf016        128  COMPLETED      0:0
259426.0       hostname                 stf016         64  COMPLETED      0:0
```

Cray and HIP Compiler Wrappers

Remove need to link certain libraries & headers, but sometimes it's helpful to see what's actually being invoked.

```
$ CC -craype-verbose test.cpp
clang++ -march=znver2 -dynamic -D__CRAY_X86_ROME -D__CRAYXT_COMPUTE_LINUX_TARGET --gcc-toolchain=/opt/gcc/8.1.0/snos -isystem /opt/cray/pe/cce/11.0.4/cce-
clang/x86_64/lib/clang/11.0.0/include -isystem /opt/cray/pe/cce/11.0.4/cce/x86_64/include/craylibs -Wl,-rpath=/opt/cray/pe/cce/11.0.4/cce/x86_64/lib -Wl,-
rpath=/opt/cray/pe/gcc-libs tests.cpp -I/opt/cray/pe/libsci/21.04.1.1/CRAY/9.0/x86_64/include -I/opt/cray/pe/mpich/8.1.4/ofi/cray/9.1/include -
I/opt/cray/pe/pmi/6.0.10/include -I/opt/cray/pe/dsmml/0.1.4/dsmml//include -I/opt/cray/xpmem/2.2.40-2.1_2.7__g3cf3325.shasta/include -
L/opt/cray/pe/libsci/21.04.1.1/CRAY/9.0/x86_64/lib -L/opt/cray/pe/mpich/8.1.4/ofi/cray/9.1/lib -L/opt/cray/pe/pmi/6.0.10/lib -
L/opt/cray/pe/dsmml/0.1.4/dsmml//lib -L/opt/cray/pe/cce/11.0.4/cce/x86_64/lib/pkgconfig/../ -L/opt/cray/xpmem/2.2.40-2.1_2.7__g3cf3325.shasta/lib64 -lpmi -Wl,--
as-needed,-lsci_cray_mpi,--no-as-needed -Wl,--as-needed,-lsci_cray,--no-as-needed -ldl -Wl,--as-needed,-lmpi_cray,--no-as-needed -lxpmem -Wl,--as-needed,-
ldsmml,--no-as-needed -Wl,--as-needed,-lpgas-shmem,--no-as-needed -lquadmath -lmodules -lfi -lcraymath -lf -lu -lcsup -Wl,--as-needed,-lpthread,-latomic,--no-as-
needed -Wl,--as-needed,-lm,--no-as-needed
clang-11: error: no such file or directory: 'test.cpp'
```

```
$ export HIPCC_VERBOSE=3
$ hipcc test.cpp
HIP_PATH=/sw/spock/spack-envs/views/rocm-4.1.0/hip
HIP_PLATFORM=amd
HIP_COMPILER=clang
HIP_RUNTIME=rocclr
ROCM_PATH=/sw/spock/spack-envs/views/rocm-4.1.0
HIP_ROCCLR_HOME=/sw/spock/spack-envs/views/rocm-4.1.0/hip
HIP_CLANG_PATH=/sw/spock/spack-envs/views/rocm-4.1.0/llvm/bin
HIP_CLANG_INCLUDE_PATH=/autofs/nccs-svm1_sw/spock/spack-envs/views/rocm-4.1.0/llvm/lib/clang/12.0.0/include
HIP_INCLUDE_PATH=/sw/spock/spack-envs/views/rocm-4.1.0/hip/include
HIP_LIB_PATH=/sw/spock/spack-envs/views/rocm-4.1.0/hip/lib
DEVICE_LIB_PATH=/sw/spock/spack-envs/views/rocm-4.1.0/amdgcn/bitcode
hipcc-cmd: /sw/spock/spack-envs/views/rocm-4.1.0/llvm/bin/clang++ -D__HIP_ROCclr__ -std=c++11 -isystem /autofs/nccs-svm1_sw/spock/spack-envs/views/rocm-
4.1.0/llvm/lib/clang/12.0.0/include/.. -isystem /sw/spock/spack-envs/views/rocm-4.1.0/include -D__HIP_ROCclr__ -isystem /sw/spock/spack-envs/views/rocm-
4.1.0/hip/include --offload-arch=gfx908 -O3 -mllvm -amdgpu-early-inline-all=true -mllvm -amdgpu-function-calls=false -fhip-new-launch-api --rocm-
path=/sw/spock/spack-envs/views/rocm-4.1.0  -L/sw/spock/spack-envs/views/rocm-4.1.0/hip/lib -O3 -lgcc_s -lgcc -lpthread -lm  -x hip test.cpp -Wl,--enable-new-
dtags -Wl,--rpath=/sw/spock/spack-envs/views/rocm-4.1.0/hip/lib:/sw/spock/spack-envs/views/rocm-4.1.0/lib -lamdhip64  -L/sw/spock/spack-envs/views/rocm-
4.1.0/llvm/bin/../lib/clang/12.0.0/lib/linux -lclang_rt.builtins-x86_64
clang-12: error: no such file or directory: 'test.cpp'
```

HIPCC_VERBOSE=<value>, where value can be 1-7

Linking in HSA Library

```
ld.lld: error: /opt/cray/pe/mpich/8.1.4/gtl/lib/libmpi_gtl_hsa.so: undefined reference to hsa_amd_memory_pool_allocate [--no-allow-shlib-undefined]
ld.lld: error: /opt/cray/pe/mpich/8.1.4/gtl/lib/libmpi_gtl_hsa.so: undefined reference to hsa_amd_agent_iterate_memory_pools [--no-allow-shlib-undefined]
ld.lld: error: /opt/cray/pe/mpich/8.1.4/gtl/lib/libmpi_gtl_hsa.so: undefined reference to hsa_iterate_agents [--no-allow-shlib-undefined]
```

Users might need to manually link in `libhsa-runtime64.so` in some codes.

- E.g., When setting `PE_MPICH_GTL_DIR_amd_gfx908` and `PE_MPICH_GTL_LIBS_amd_gfx908` to use GPU-Aware MPI.
- If needed, you can link in this library using: `$ROCM_PATH/lib -lhsa-runtime64`

You can learn more about HSA here:

- https://rocmdocs.amd.com/en/latest/Installation_Guide/ROCR-Runtime.html
- https://rocmdocs.amd.com/en/latest/ROCm_Glossary/ROCm-Glossary.html

Keeping Up with the Clangs

There are multiple version of Clang compilers on Spock. If you're uncertain of which version is being used in your environment, you can issue `which clang++` to find out.

```
$ module -t list
cce/11.0.4
craype/2.7.6
craype-x86-rome
libfabric/1.11.0.3.74
craype-network-ofi
cray-dsmml/0.1.4
perftools-base/21.02.0
xpmem/2.2.40-2.1_2.7__g3cf3325.shasta
cray-mpich/8.1.4
cray-libsci/21.04.1.1
cray-pmi/6.0.10
cray-pmi-lib/6.0.10
DefApps/default
PrgEnv-cray/8.0.0
rocm/4.1.0

$ which clang++
/sw/spock/spack-envs/views/rocm-4.1.0/llvm/bin/clang++

$ module unload rocm

$ which clang++
/opt/cray/pe/cce/11.0.4/cce-clang/x86_64/bin/clang++
```

It is highly recommended to use the Cray compiler wrappers (`cc` , `CC` , and `ftn`) whenever possible. See the next section for more details.

Vendor	Programming Environment	Compiler Module	Language	Compiler Wrapper	Compiler
Cray	`PrgEnv-cray`	`cce`	C	`cc`	`craycc`
			C++	`CC`	`craycxx` or `crayCC`
			Fortran	`ftn`	`crayftn`
AMD	Not yet available	`rocm`	C	Not yet available	`$ROCM_PATH/llvm/bin/clang`
			C++	Not yet available	`$ROCM_PATH/llvm/bin/clang++`
			Fortran	Not yet available	`$ROCM_PATH/llvm/bin/flang`

NOTE: It is highly recommended to use the Cray compiler wrappers whenever possible, but if for some reason the wrappers cannot be used, the `craycc`, `crayCC`, and `crayftn` compilers should be used instead of directly using Cray's Clang compilers.

- For ROCm Clang compilers, give full path
- For Cray Clang compilers, use compiler wrappers or `craycc`, `crayCC`, `crayftn`

Generate Detailed Node Image

From within an interactive job (or via batch script)…

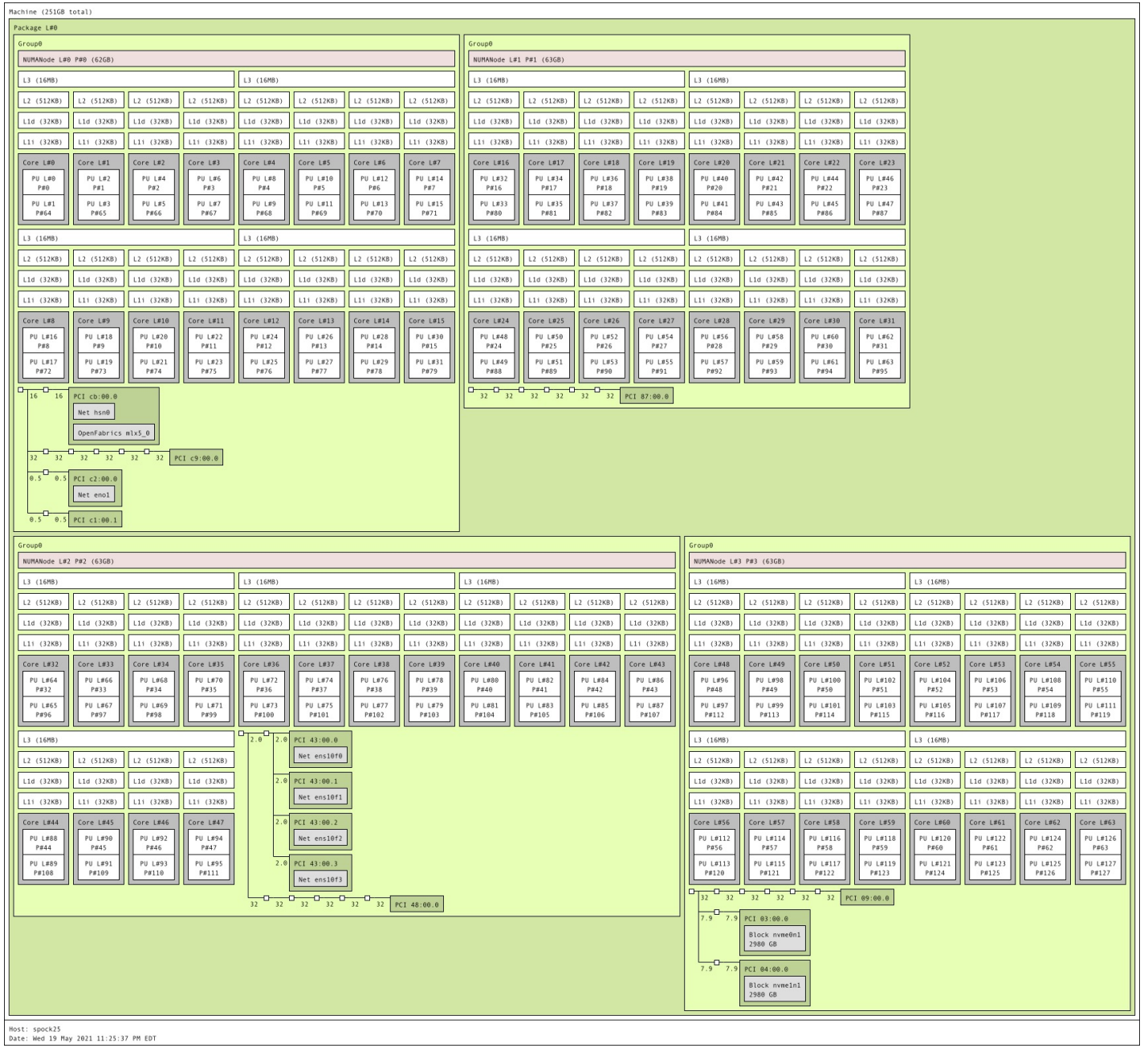
```
$ salloc -A stf016 -t 30 -N 1
salloc: Pending job allocation 258383
salloc: job 258383 queued and waiting for resources
salloc: job 258383 has been allocated resources
salloc: Granted job allocation 258383
salloc: Waiting for resource configuration
salloc: Nodes spock27 are ready for job

$ module load hwloc

$ srun -n1 lstopo spock_compute_node.svg
```

`scp` to your laptop/desktop and open in browser or convert to png using e.g., inkscape

I don't expect you to be able to read this in the slide :)



OAK RIDGE National Laboratory

Helpful Resources

Spock Quick-Start Guide: https://docs.olcf.ornl.gov/systems/spock_quick_start_guide.html

- Make sure to look at the Known Issues section if you encounter problems.

Spock Mailing List: All Spock users are included on a mailing list where OLCF will communicate about downtimes, software upgrades, etc.

ROCm Documentation: https://rocmdocs.amd.com/en/latest/index.html

ROCm GitHub Pages:

- **https://github.com/ROCmSoftwarePlatform**
- **https://github.com/ROCm-Developer-Tools**

Getting Help: If you have problems or need help running on Spock, please submit a support ticket to help@olcf.ornl.gov.

Questions?

OAK RIDGE National Laboratory

Backup Slides

Find CPUs available to each MPI rank

Prepend task number to lines of stdout/err.

This is what is being launched on the compute node

From within an interactive job…

```
$ salloc -A stf016 -t 30 -N 1
salloc: Pending job allocation 258383
salloc: job 258383 queued and waiting for resources
salloc: job 258383 has been allocated resources
salloc: Granted job allocation 258383
salloc: Waiting for resource configuration
salloc: Nodes spock27 are ready for job
```

I want my 8 MPI ranks to be placed 2-per-NUMA, where ranks 0-1 are on NUMA 0, ranks 2-3 are on NUMA 1, etc.

How do I know which CPU cores are available to each MPI rank?

Cyclic across NUMAs

```
$ srun -l -n8 -c2 /bin/bash -c 'grep Cpus_allowed_list /proc/self/status' | sort
0: Cpus_allowed_list:     0-1
1: Cpus_allowed_list:     16-17
2: Cpus_allowed_list:     32-33
3: Cpus_allowed_list:     48-49
4: Cpus_allowed_list:     2-3
5: Cpus_allowed_list:     18-19
6: Cpus_allowed_list:     34-35
7: Cpus_allowed_list:     50-51
```

Block across NUMAs

```
$ srun -l -n8 -c2 -m *:block /bin/bash -c 'grep Cpus_allowed_list /proc/self/status' | sort
0: Cpus_allowed_list:     0-1
1: Cpus_allowed_list:     2-3
2: Cpus_allowed_list:     4-5
3: Cpus_allowed_list:     6-7
4: Cpus_allowed_list:     8-9
5: Cpus_allowed_list:     10-11
6: Cpus_allowed_list:     12-13
7: Cpus_allowed_list:     14-15
```

Block across NUMAs (adjust CPUs/task)

```
$ srun -l -n8 -c8 -m *:block /bin/bash -c 'grep Cpus_allowed_list /proc/self/status' | sort
0: Cpus_allowed_list:     0-7
1: Cpus_allowed_list:     8-15
2: Cpus_allowed_list:     16-23
3: Cpus_allowed_list:     24-31
4: Cpus_allowed_list:     32-39
5: Cpus_allowed_list:     40-47
6: Cpus_allowed_list:     48-55
7: Cpus_allowed_list:     56-63
```

Slurm partitions

Spock's compute nodes are separated into 2 Slurm partitions (queues): 1 for CAAR projects and 1 for ECP.

CAAR Partition

Number of Nodes	Max Walltime
1 – 4	3 hours
5 – 16	1 hour

- 24 total compute nodes
- Each user can have
  - → 1 running and 1 eligible job at a time (per project)
  - → No limit on the number of jobs submitted.

ECP Partition

Number of Nodes	Max Walltime
1 – 4	3 hours

- 12 total compute nodes
- Each user can have
  - → 1 running and 1 eligible job at a time (per project)
  - → Up to 5 jobs submitted

If CAAR or ECP teams require a temporary exception to this policy, please email help@olcf.ornl.gov with your request and it will be given to the OLCF Resource Utilization Council (RUC) for review

GPU-Aware MPI

When using GPU-Aware MPI with the Cray compiler wrappers, there are currently some extra steps needed…

The following modules must be loaded:

```
module load PrgEnv-cray
module load craype-accel-amd-gfx908
module load rocm
```

The following environment variables must be set *before compiling* so the executable picks up GTL

```
export PE_MPICH_GTL_DIR_amd_gfx908="-L/opt/cray/pe/mpich/8.1.4/gtl/lib"
export PE_MPICH_GTL_LIBS_amd_gfx908="-lmpi_gtl_hsa"
```

The following environment variables must be set before running

```
export MPIR_CVAR_GPU_EAGER_DEVICE_MEM=0
export MPICH_GPU_SUPPORT_ENABLED=1
export MPICH_SMP_SINGLE_COPY_MODE=CMA
```

If you set this in a script, it will need to be removed once a known issue is resolved.

In addition, the following header files and libraries must be included:

```
-I${ROCM_PATH}/include
-L${ROCM_PATH}/lib -lamdhip64 -lhsa-runtime64
```