OAK RIDGE National Laboratory

Spock Slurm Training

May 20th, 2021

Presenters

Matthew Davis - HPC Engineer

Hong Liu - HPC Engineer

ORNL is managed by UT-Battelle, LLC for the US Department of Energy

U.S. DEPARTMENT OF ENERGY

Outline

What is slurm

- Workload manager

How to run jobs on Spock

- Interactive Jobs
- Single Command job (non-interactive)
- Batch jobs

Process and Thread mapping

Slurm

What is slurm

- Slurm is an open source, fault-tolerant, and highly scalable cluster management and job scheduling system for large and small Linux clusters

- Role of slurm -

  Resource management and job scheduling

Slurm

How to run jobs on Spock ?

3 ways of submitting and launching jobs on Spock

- Interactive Jobs
- Single Command job (non-interactive)
- Batch jobs

Slurm

The Slurm commands associated with these methods:

sbatch	Used to submit a batch script to allocate a Slurm job allocation
salloc	Used to allocate an interactive Slurm job allocation
srun	Used to run a job (job step) on the resources allocated with sbatch or salloc

Interactive Jobs

```
$ salloc -A <project_id> -J <job_name> -t 00:05:00 -p <partition> -N 2
salloc: Granted job allocation 4258
salloc: Waiting for resource configuration
salloc: Nodes spock[10-11] are ready for job
```

```
$ srun -n 2 --ntasks-per-node=1 ./a.out
<output printed to terminal>
```

Interactive Jobs

```
[200~liuh@login1:~/Training> salloc -A STF007uanofn -J test -t 00:05:00 -p caar -N 2
salloc: Pending job allocation 249623
salloc: job 249623 queued and waiting for resources
salloc: job 249623 has been allocated resources
salloc: Granted job allocation 249623
salloc: Waiting for resource configuration
salloc: Nodes spock[31-32] are ready for job
liuh@spock31:~/Training> hostname
spock31
liuh@spock31:~/Training> srun -n 2 --ntasks-per-node=1 ./a.out
Hello, World!
Hello, World!
~
~
```

Single Command Job

Single Command Jobs (non-interactive)

```
$ srun -A <project_id> -t 00:05:00 -p <partition> -N 2 -n 4
--ntasks-per-node=2 ./a.out
<output printed to terminal>
```

```
liuh@login1:~/Training> srun -A STF007uanofn -t 00:05:00 -p caar -N 2 -n 4 --ntasks-per-node=2 ./a.out
srun: job 249650 queued and waiting for resources
srun: job 249650 has been allocated resources
Hello, World!
Hello, World!
Hello, World!
Hello, World!
```

Batch Script

```
#!/bin/bash
#SBATCH -A <project_id>
#SBATCH -J <job_name>
#SBATCH -o %x-%j.out
#SBATCH -t 00:05:00
#SBATCH -p <partition>
#SBATCH -N 2
srun -n4 --ntasks-per-node=2 ./a.out
```

```
liuh@login1:~/Training> sbatch test.sh
Submitted batch job 249660
liuh@login1:~/Training> ls *249660*
test-249660.out
liuh@login1:~/Training> more test-249660.out
Hello, World!
Hello, World!
Hello, World!
Hello, World!
```

Partitions

Spock's compute nodes are separated into 2 Slurm partitions:

- CAAR

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

- ECP

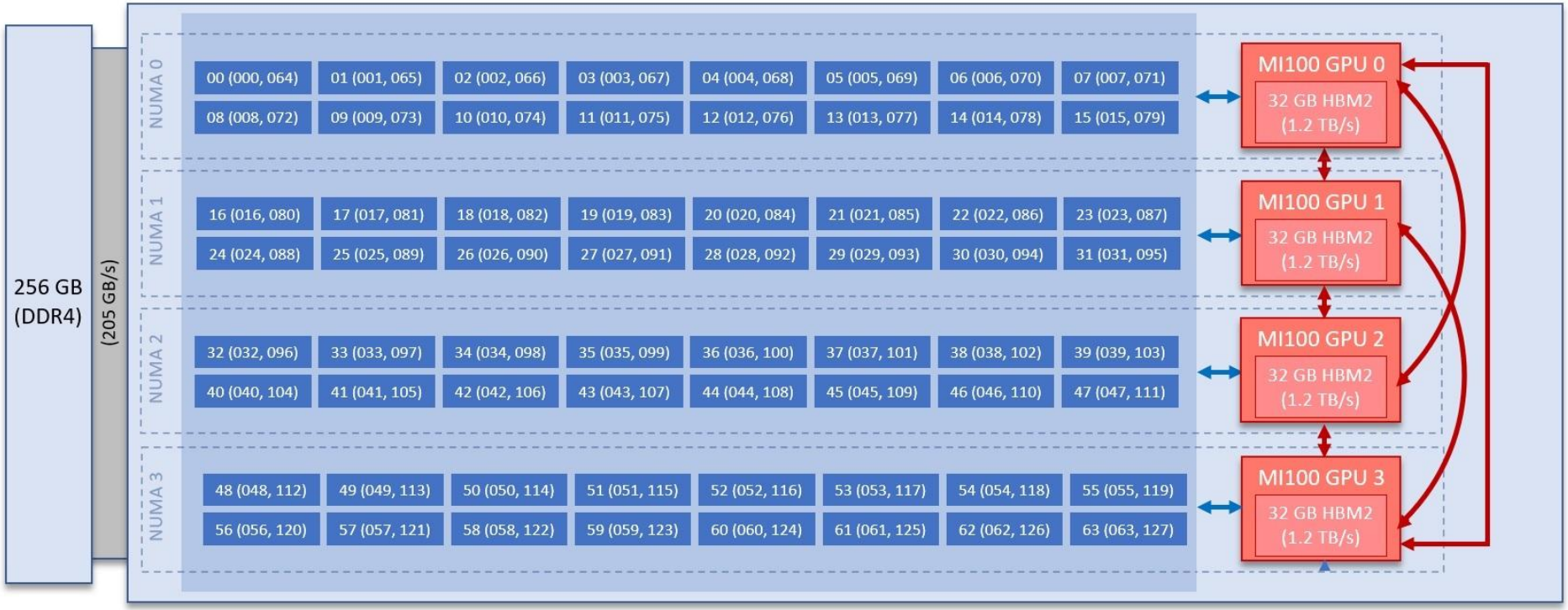
Number of Nodes	Max Walltime
1 - 4	3 hours

Slurm

Common Slurm Commands

`sinfo`	**Used to view partition and node information.** `sinfo -p caar`
`squeue`	**Used to view job and job step information for jobs in the scheduling queue.** `squeue -l -u <user_id>`
`sacct`	**Used to view accounting data for jobs and job steps in the job accounting log** `sacct -u <username> -S 2021-01-04T13:00:00 -o "jobid%5,jobname%25,user%15,nodelist%20" -X`
`scancel`	`scancel <jobid>`
`scontrol`	`scontrol hold <jobid>`

Process and Thread Mapping



Process and Thread Mapping

CPU mapping

-c, --cpus-per-task=<ncpus>	Request that ncpus be allocated per process (default is 1). ncpus refers to hardware threads.
--threads-per-core=<threads>	In task layout, use the specified maximum number of threads per core (default is 1; there are 2 hardware threads per physical CPU core).
--cpu-bind=threads	Bind tasks to CPUs. threads - Automatically generate masks binding tasks to threads. (Although this option is not explicitly used in the following examples, it is the default CPU binding.)

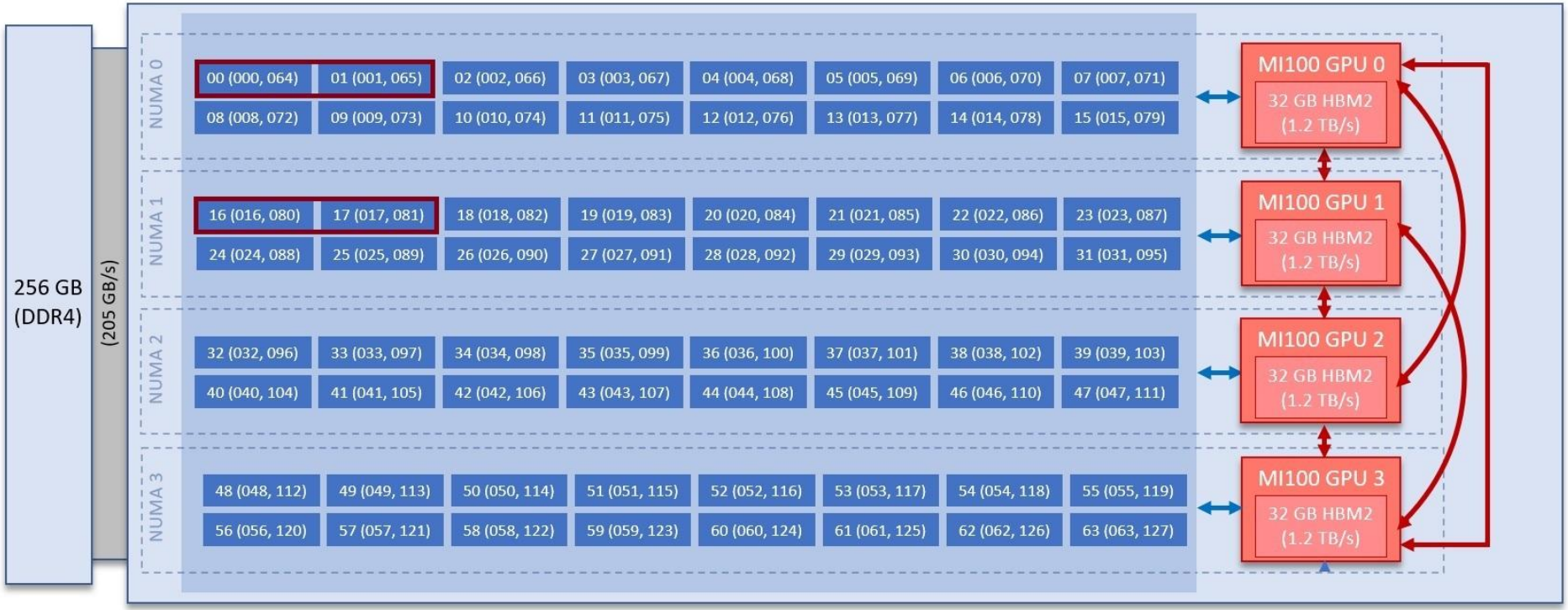
CPU Mapping

```
salloc -A <project_id> -t 30 -p <parition> -N 1

$ export OMP_NUM_THREADS=2
$ srun -n2 -c2 ./hello_mpi_omp | sort

MPI 000 - OMP 000 - HWT 000 - Node spock13
MPI 000 - OMP 001 - HWT 001 - Node spock13
MPI 001 - OMP 000 - HWT 016 - Node spock13
MPI 001 - OMP 001 - HWT 017 - Node spock13
```

CPU Mapping


256 GB (DDR4)
(205 GB/s)
NUMA 0
00 (000, 064)
01 (001, 065)
02 (002, 066)
03 (003, 067)
04 (004, 068)
05 (005, 069)
06 (006, 070)
07 (007, 071)
08 (008, 072)
09 (009, 073)
10 (010, 074)
11 (011, 075)
12 (012, 076)
13 (013, 077)
14 (014, 078)
15 (015, 079)
MI100 GPU 0
32 GB HBM2 (1.2 TB/s)
NUMA 1
16 (016, 080)
17 (017, 081)
18 (018, 082)
19 (019, 083)
20 (020, 084)
21 (021, 085)
22 (022, 086)
23 (023, 087)
24 (024, 088)
25 (025, 089)
26 (026, 090)
27 (027, 091)
28 (028, 092)
29 (029, 093)
30 (030, 094)
31 (031, 095)
MI100 GPU 1
32 GB HBM2 (1.2 TB/s)
NUMA 2
32 (032, 096)
33 (033, 097)
34 (034, 098)
35 (035, 099)
36 (036, 100)
37 (037, 101)
38 (038, 102)
39 (039, 103)
40 (040, 104)
41 (041, 105)
42 (042, 106)
43 (043, 107)
44 (044, 108)
45 (045, 109)
46 (046, 110)
47 (047, 111)
MI100 GPU 2
32 GB HBM2 (1.2 TB/s)
NUMA 3
48 (048, 112)
49 (049, 113)
50 (050, 114)
51 (051, 115)
52 (052, 116)
53 (053, 117)
54 (054, 118)
55 (055, 119)
56 (056, 120)
57 (057, 121)
58 (058, 122)
59 (059, 123)
60 (060, 124)
61 (061, 125)
62 (062, 126)
63 (063, 127)
MI100 GPU 3
32 GB HBM2 (1.2 TB/s)

Process and Thread Mapping

GPU mapping

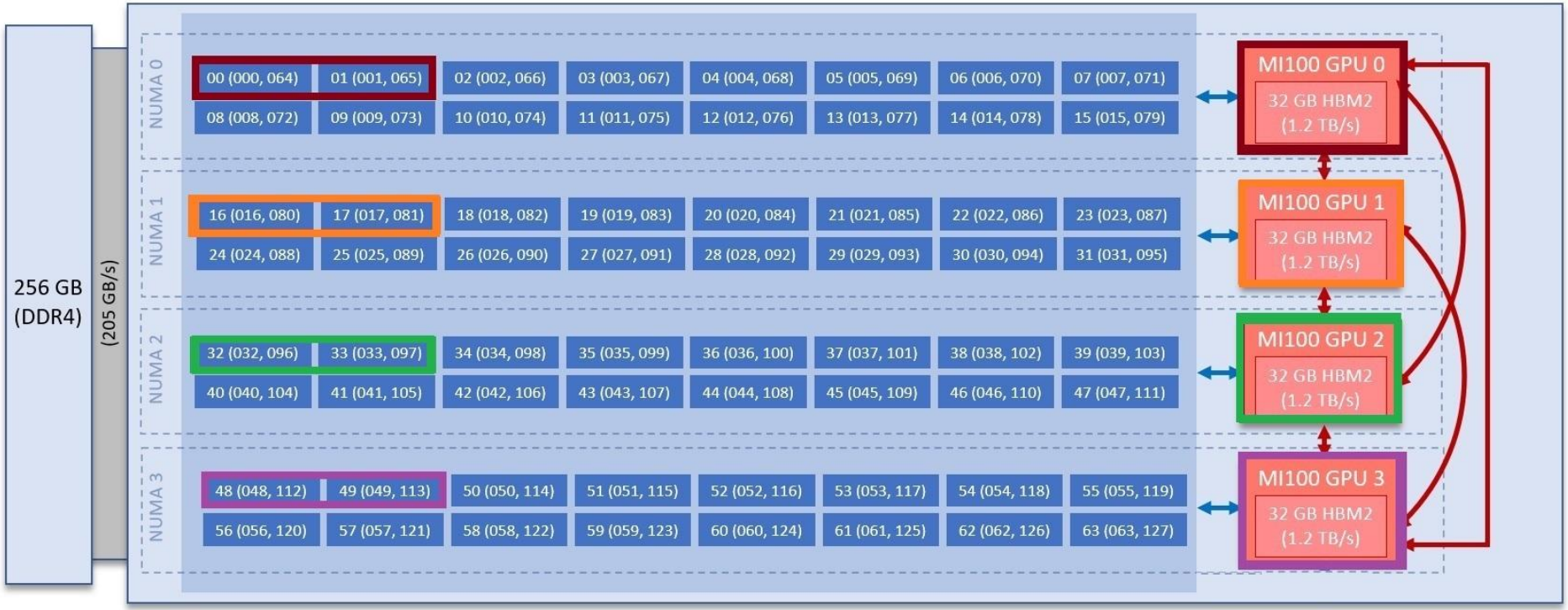
--ntasks-per-gpu=<ntasks>	Request that there are ntasks tasks invoked for every GPU.
--gpu-bind=map_gpu:<list>	Bind tasks to specific GPUs by setting GPU masks on tasks (or ranks) as specified where <list> is <gpu_id_for_task_0>,<gpu_id_for_task_1>,.... If the number of tasks (or ranks) exceeds the number of elements in this list, elements in the list will be reused as needed starting from the beginning of the list. To simplify support for large task counts, the lists may follow a map with an asterisk and repetition count. (For example map_gpu:0*4,1*4)

Map 1 task per GPU

```
$ export OMP_NUM_THREADS=2
$ srun -n4 -c2 --ntasks-per-gpu=1 ./hello_jobstep | sort
```

```
MPI 000 - OMP 000 - HWT 000 - Node spock13 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 000 - OMP 001 - HWT 001 - Node spock13 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 001 - OMP 000 - HWT 016 - Node spock13 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 001 - OMP 001 - HWT 017 - Node spock13 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 002 - OMP 000 - HWT 032 - Node spock13 - RT_GPU_ID 0 - GPU_ID 2 - Bus_ID 48
MPI 002 - OMP 001 - HWT 033 - Node spock13 - RT_GPU_ID 0 - GPU_ID 2 - Bus_ID 48
MPI 003 - OMP 000 - HWT 048 - Node spock13 - RT_GPU_ID 0 - GPU_ID 3 - Bus_ID 09
MPI 003 - OMP 001 - HWT 049 - Node spock13 - RT_GPU_ID 0 - GPU_ID 3 - Bus_ID 09
```

Map 1 task per GPU

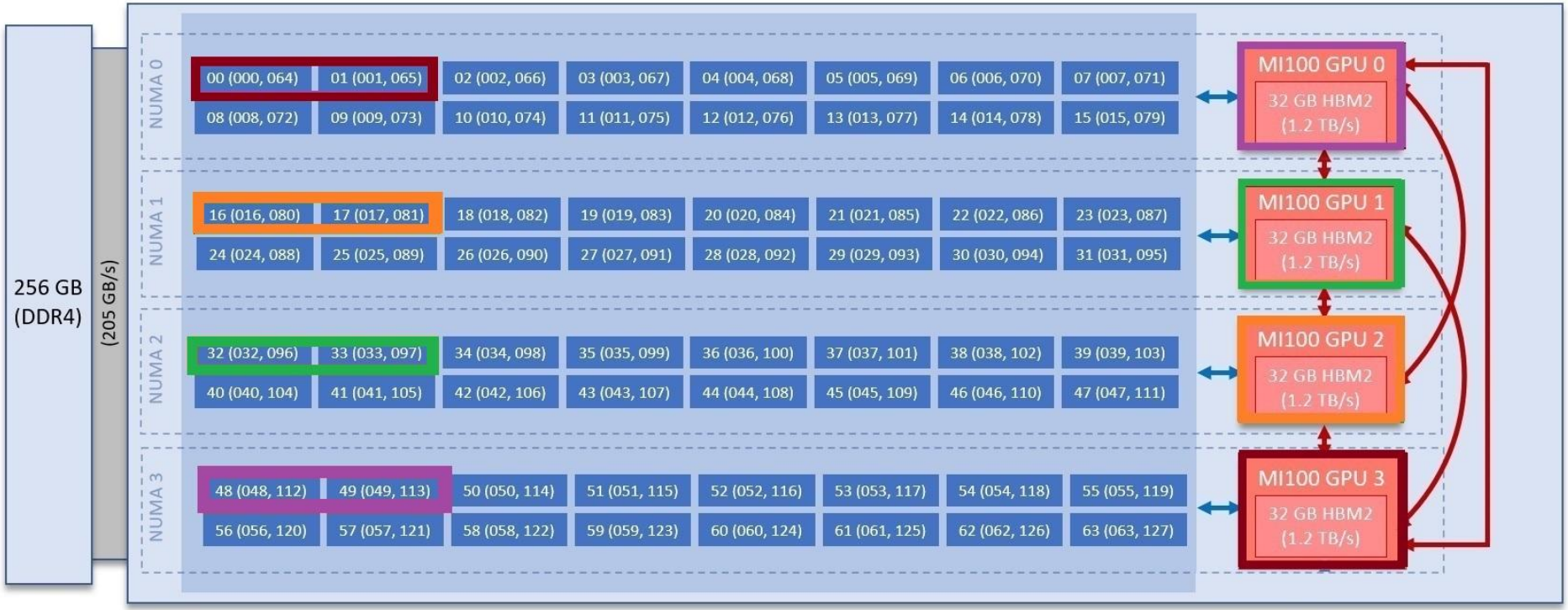


Map 1 task to a specific GPU

```
$ export OMP_NUM_THREADS=2
$ srun -n4 -c2 --ntasks-per-gpu=1 --gpu-bind=map_gpu:3,2,1,0 ./hello_jobstep | sort

MPI 000 - OMP 000 - HWT 000 - Node spock13 - RT_GPU_ID 0 - GPU_ID 3 - Bus_ID 09
MPI 000 - OMP 001 - HWT 001 - Node spock13 - RT_GPU_ID 0 - GPU_ID 3 - Bus_ID 09
MPI 001 - OMP 000 - HWT 016 - Node spock13 - RT_GPU_ID 0 - GPU_ID 2 - Bus_ID 48
MPI 001 - OMP 001 - HWT 017 - Node spock13 - RT_GPU_ID 0 - GPU_ID 2 - Bus_ID 48
MPI 002 - OMP 000 - HWT 032 - Node spock13 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 002 - OMP 001 - HWT 033 - Node spock13 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 003 - OMP 000 - HWT 048 - Node spock13 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 003 - OMP 001 - HWT 049 - Node spock13 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
```

Map 1 task to a specific GPU

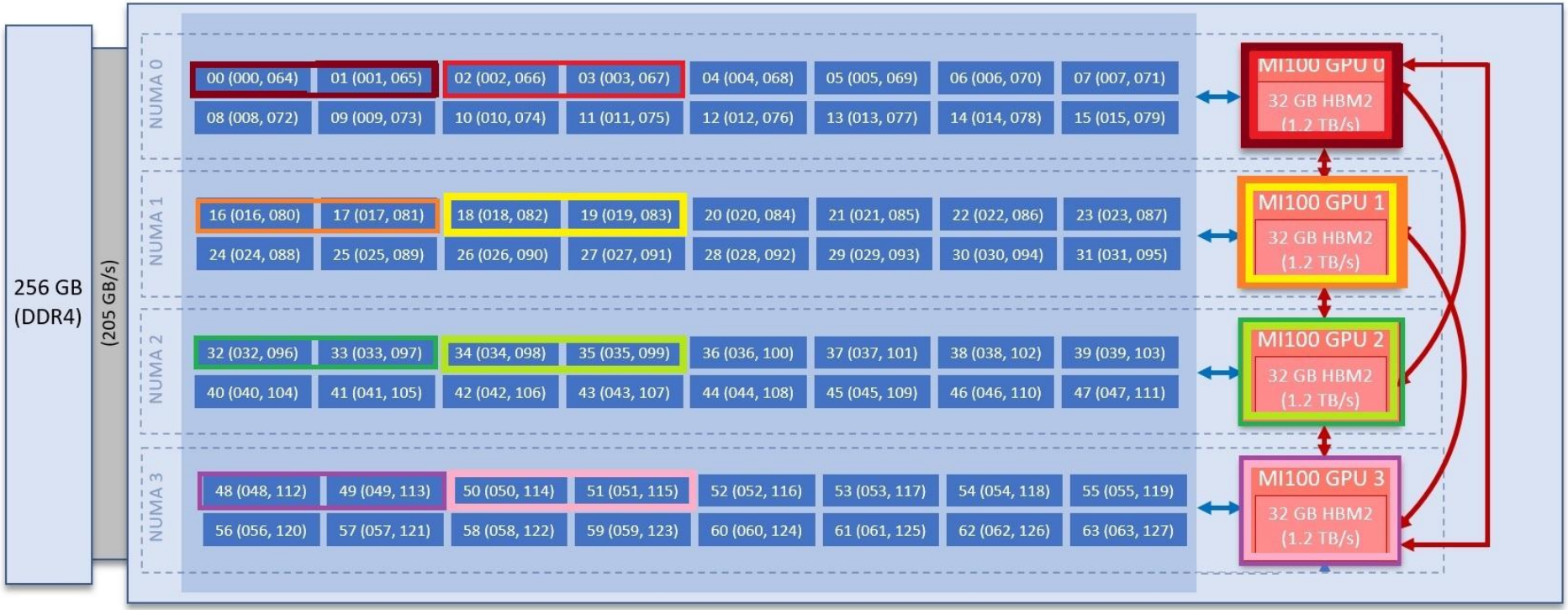


Map 2 tasks per GPU

```
$ export OMP_NUM_THREADS=2
$ srun -n8 -c2 --ntasks-per-gpu=2  ./hello_jobstep | sort
```

```
MPI 000 - OMP 000 - HWT 000 - Node spock03 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 000 - OMP 001 - HWT 001 - Node spock03 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 001 - OMP 000 - HWT 016 - Node spock03 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 001 - OMP 001 - HWT 017 - Node spock03 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 002 - OMP 000 - HWT 032 - Node spock03 - RT_GPU_ID 0 - GPU_ID 2 - Bus_ID 48
MPI 002 - OMP 001 - HWT 033 - Node spock03 - RT_GPU_ID 0 - GPU_ID 2 - Bus_ID 48
MPI 003 - OMP 000 - HWT 048 - Node spock03 - RT_GPU_ID 0 - GPU_ID 3 - Bus_ID 09
MPI 003 - OMP 001 - HWT 049 - Node spock03 - RT_GPU_ID 0 - GPU_ID 3 - Bus_ID 09
MPI 004 - OMP 000 - HWT 002 - Node spock03 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 004 - OMP 001 - HWT 003 - Node spock03 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 005 - OMP 000 - HWT 018 - Node spock03 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 005 - OMP 001 - HWT 019 - Node spock03 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 006 - OMP 000 - HWT 034 - Node spock03 - RT_GPU_ID 0 - GPU_ID 2 - Bus_ID 48
MPI 006 - OMP 001 - HWT 035 - Node spock03 - RT_GPU_ID 0 - GPU_ID 2 - Bus_ID 48
MPI 007 - OMP 000 - HWT 050 - Node spock03 - RT_GPU_ID 0 - GPU_ID 3 - Bus_ID 09
MPI 007 - OMP 001 - HWT 051 - Node spock03 - RT_GPU_ID 0 - GPU_ID 3 - Bus_ID 09
```

Map 2 task per GPU

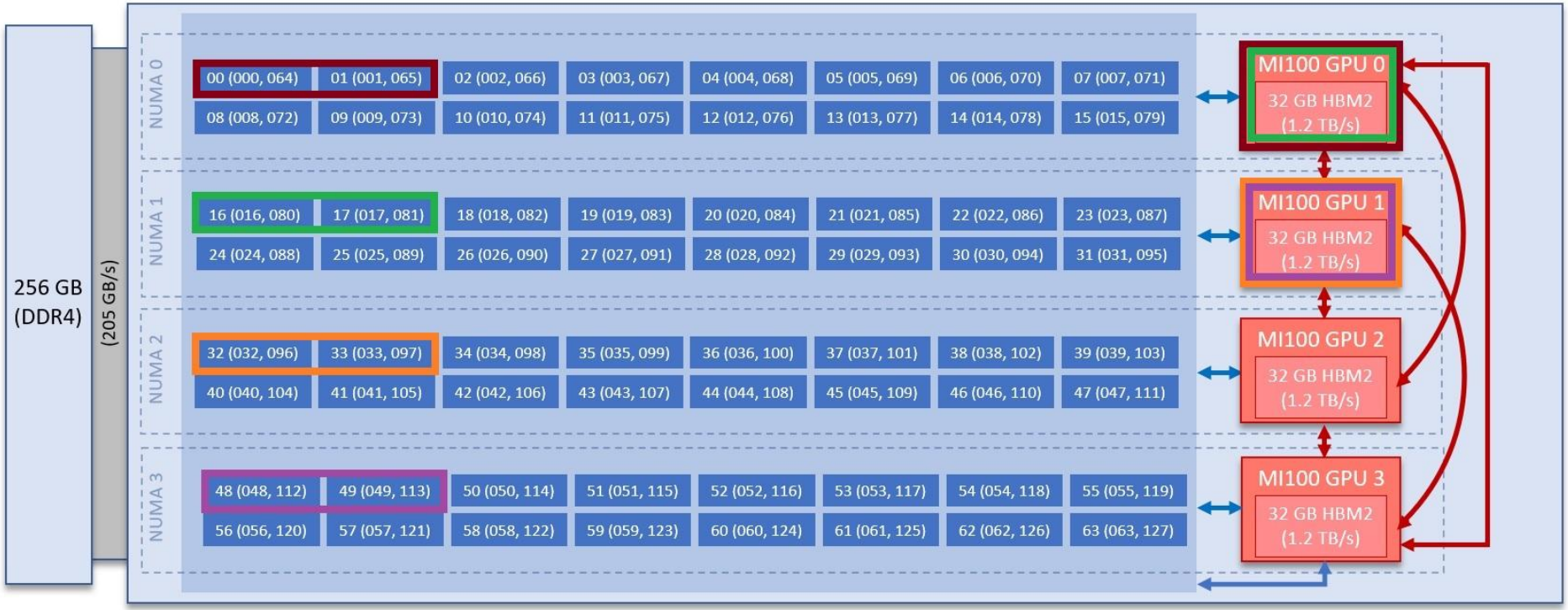


Mapping multiple MPI ranks a GPU

```
$ export OMP_NUM_THREADS=2
$ srun -n4 -c2 --ntasks-per-gpu=2 --gpu-bind=map_gpu:0*2,1*2 ./hello_jobstep | sort

MPI 000 - OMP 000 - HWT 000 - Node spock13 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 000 - OMP 001 - HWT 001 - Node spock13 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 001 - OMP 000 - HWT 016 - Node spock13 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 001 - OMP 001 - HWT 017 - Node spock13 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 002 - OMP 000 - HWT 032 - Node spock13 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 002 - OMP 001 - HWT 033 - Node spock13 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 003 - OMP 000 - HWT 048 - Node spock13 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 003 - OMP 001 - HWT 049 - Node spock13 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
```

Mapping multiple MPI ranks a GPU



Mapping multiple MPI ranks a GPU

```
export OMP_NUM_THREADS=2
$ srun -n8 -c2 --ntasks-per-gpu=4 --gpu-bind=map_gpu:0*2,1*2 ./hello_jobstep | sort
MPI 000 - OMP 000 - HWT 000 - Node spock13 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 000 - OMP 001 - HWT 001 - Node spock13 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 001 - OMP 000 - HWT 016 - Node spock13 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 001 - OMP 001 - HWT 017 - Node spock13 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 002 - OMP 000 - HWT 032 - Node spock13 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 002 - OMP 001 - HWT 033 - Node spock13 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 003 - OMP 000 - HWT 048 - Node spock13 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 003 - OMP 001 - HWT 049 - Node spock13 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 004 - OMP 000 - HWT 002 - Node spock13 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 004 - OMP 001 - HWT 003 - Node spock13 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 005 - OMP 000 - HWT 018 - Node spock13 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 005 - OMP 001 - HWT 019 - Node spock13 - RT_GPU_ID 0 - GPU_ID 0 - Bus_ID c9
MPI 006 - OMP 000 - HWT 034 - Node spock13 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 006 - OMP 001 - HWT 035 - Node spock13 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 007 - OMP 000 - HWT 050 - Node spock13 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
MPI 007 - OMP 001 - HWT 051 - Node spock13 - RT_GPU_ID 0 - GPU_ID 1 - Bus_ID 87
```

Mapping multiple MPI ranks a GPU