Wikiprint Book

Title: System usage

Subject: DEEP - Public/User_Guide/DEEP-EST_DAM

Version: 24

Date: 19.05.2024 23:25:27

Table of Contents

System usage

The DEEP-EST Data Analytics Module (DAM) can be used through the SLURM based batch system that is also used for (most of) the Software Development Vehicles (SDV). You can request a DAM node (`dp-dam[01-16]`) with an interactive session like this:

```
srun -A deepsea -N 1 --tasks-per-node 4 -p dp-dam --time=1:0:0 --pty --interactive /bin/bash
kreutz1@dp-dam01 ~]$ srun -n 8 hostname
dp-dam01
dp-dam01
dp-dam01
dp-dam01
```

When using a batch script, you have to adapt the partition option within your script: `--partition=dp-dam` (or short form: `-p dp-dam`)

Persistent Memory

Each of the DAM nodes is equipped with ?Intel's Optane DC Persistent Memory Modules (DCPMM). All DAM nodes (`dp-dam[01-16]`) expose 3 TB of persistent memory.

The DCPMMs can be driven in different modes. For further information of the operation modes and how to use them, please refer to the following ?information

Currently all nodes are running in "App Direct Mode".

Using Cuda

The first 12 DAM nodes are equipped with GPUs

- `dp-dam[01-08]`: 1 x Nvidia V100
- `dp-dam[09-12]`: 2 x Nvidia V100

Please use the `gres` option with `srun` if you would like to use GPUs on DAM nodes, e.g. in an interactive session:

```
srun -A deepsea -p dp-dam --gres=gpu:1 -t 1:0:0 --interactive --pty /bin/bash  # to start an interactive session on an DAM
srun -A deepsea -p dp-dam --gres=gpu:2 -t 1:0:0 --interactive --pty /bin/bash  # to start an interactive session on an DAM
```

To compile and run Cuda applications on the Nvidia V100 cards included in the DAM nodes, it is necessary to load the CUDA module. It's advised to use the 2022 Stage to avoid Nvidia driver mismatch issues.

```
module --force purge
ml use $OTHERSTAGES
ml Stages/2022
ml CUDA
[kreutz1@deepv ~]$ ml

Currently Loaded Modules:
 1) Stages/2022 (S)   2) nvidia-driver/.default (H,g,u)   3) CUDA/11.5 (g,u)

 Where:
  S:  Module is Sticky, requires --force to unload or purge
  g:  built for GPU
  u:  Built by user
```

Using FPGAs

Nodes `dp-dam[13-16] are equipped with 2 x Stratix 10 FPGAs each (?Intel PAC d5005).

It is recommended to do the first steps in an interactive session on a DAM node. Since there is (currently) no FPGA resource defined in SLURM for these nodes, please use the `--hostlist=` option with `srun` to open a session on a DAM node equipped with FPGAs, for example:

```
srun -A deepsea -p dp-dam --nodelist=dp-dam13 -t 1:0:0 --interactive --pty /bin/bash
```

For getting started using OpenCL with the FPGAs you can find some hints as well as the slides and exercises from the Intel FPGA workshop held at JSC in:

```
/usr/local/software/legacy/fpga/
```

More details to follow.

Filesystems and local storage

The home filesystem on the DEEP-EST Cluster Module is provided via GPFS/NFS and hence the same as on (most of) the remaining compute nodes. The local storage system of the DAM running BeeGFS is available at

```
/work
```

The file servers are reachable through the 40 GbE interface of the DAM nodes.

This is NOT the same storage being used on the DEEP-ER SDV system. Both, the DEEP-EST prototype system and the DEEP-ER SDV have their own local storage.

It's possible to access the local storage of the DEEP-ER SDV (`/sdv-work`), but you have to keep in mind that the file servers of that storage can just be accessed through 1 GbE ! Hence, it should not be used for performance relevant applications since it is much slower than the DEEP-EST local storages mounted to `/work`.

There is node local storage available for the DEEP-EST DAM node (2 x 1.5 TB NVMe SSD), it is mounted to `/nvme/scratch` and `/nvme/scratch2`. Additionally, there is a small (about 380 GB) scratch folder available in `/scratch`. Remember that the three **scratch folders** are not persistent and **will be cleaned after your job has finished** !

Multi-node Jobs

Multi-node MPI jobs can be launched on the DAM nodes with ParaStation MPI by loading the `Intel` (or `GCC`) and `ParaStationMPI` modules.

Extoll: As of 12.12.2019, the first half of the DAM nodes (`dp-dam[01-08]`) has only GbE connectivity, while the second half has also the faster Extoll interconnect active (nodes `dp-dam[09-16]`). To run multi-node MPI jobs on the DAM nodes, it is strongly recommended to use the `dp-dam-ext` partition, which includes only the nodes providing EXTOLL connectivity. If necessary, users can also run MPI jobs on the other DAM nodes (using the `dp-dam` partition) by setting the `PSP_TCP=1` environment variable in their scripts. This will cause any MPI communication to go through the slower 40 Gb Ethernet fabric.

A release-candidate version of ParaStationMPI with CUDA awareness and GPU direct support for Extoll is currently being tested. Once released it will become available on the DAM nodes with the modules environment. Further information on CUDA awareness can be found in the ParaStationMPI section. As a temporary workaround, the current version of ParaStationMPI automatically performs device-to-host, host-to-host and host-to-device copies transparently to the user, so it can be used to run applications requiring a CUDA-aware MPI implementation (with limited data transfer performance).

For using Cluster nodes in heterogeneous jobs together with CM and/or ESB nodes, please see info about heterogeneous jobs.