

## **Wikiprint Book**

**Title: General information**

**Subject: DEEP - Public/User\_Guide/SDV\_KNLs**

**Version: 12**

**Date: 22.07.2024 19:25:46**

## Table of Contents

<b>General information</b>	<b>3</b>
<b>Node allocation</b>	<b>3</b>
Available knl partitions	3
<b>Compiling</b>	<b>3</b>
Multi-node Jobs	3
<b>5 things to consider when using KNL</b>	<b>3</b>

## General information

We have 3 KNLs in the SDV right now.

All KNL nodes have their own local NVMe device installed.

## Node allocation

Nodes can be allocated through the Slurm based batch system that is also used for the DEEP-EST system and the SDV Xeon Cluster. You can start an interactive session on our KNLs like this:

```
srunc --partition=kn1 -N 2 -n 8 --pty /bin/bash -i
```

When using a batch script, you have to adapt the `--partition` option within your script: `--partition=kn1`

## Available knl partitions

- kn1: The DEEP-ER knl nodes (all of them, regardless of cpu and configuration)
- kn1256: the 256-core knls (kn15)
- kn1272: the 272-core knls (kn14, kn16)
- snc4: the knls configured in SNC-4 mode (kn15)

## Compiling

Use the `-xMIC-AVX512` flag instead of `-mmic`.

Check actual vectorisation with `-qopt-report=5 -qopt-report-phase=vec` → info given in `*.oprpt` files

## Multi-node Jobs

The KNL nodes are only connected via Gigabit Ethernet, hence there is no need to load the Extoll module to run jobs on multiple nodes.

## 5 things to consider when using KNL

- Make sure to use the fast MCDRAM:
  - When MCDRAM is in cache mode:
    - No changes are needed.
  - When MCDRAM is in flat mode:
    - If the total memory footprint of the application is smaller than the size of MCDRAM: `numactl ?m 1 ./my_application.out` (Allocations that don't fit into MCDRAM make the application fail.)
    - If the total memory footprint of the application is larger than the size of MCDRAM: `numactl ?p 1 ./my_application.out` (Allocations that don't fit into MCDRAM spill over to DDR)
    - To make a manual choice of what should be allocated in the MCDRAM: Use the `memkind` library.
- Verify that the pinning is as you wish:
  - Start job on KNL node(s).
  - Log in on KNL.
  - Invoke `htop`.
  - Check the load distribution.
  - Remark: Each core can execute 1, 2 or 4 threads. On KNL ? unlike on KNC ? already one thread per core can lead to optimal performance.
- Use VTune/Advisor to analyse the performance:
  - Start job on KNL node(s).
  - Log in on KNL.
  - `'module load VTune / Advisor'`.
  - Run `amplxe-gui / advixe-gui`.
  - Follow instructions.
  - Remark: If you run into errors of the sort `?sepd not available?` please contact the administrator. Both tools rely on a kernel module to access hardware counter.

- i. Provide hints to the compiler:
  - Check `*optrpt` for info on vectorisation.
  - If you find `?unaligned...?` → add alignment in your code by adding `"#pragma vector aligned"` before the loop.
  - If a loop does not vectorise although it clearly should, you can add `"#pragma simd"` before the loop.
  - Re-check `*.optrpt`.
  - Re-check in VTune / Advisor
- i. Verify the performance via benchmarks:
  - Set up JUBE for your code.
  - Benchmark the various versions with proper timing.
  - Be aware: VTune / Advisor sometimes give estimates that are a little off. It's imperative to check the actual performance.