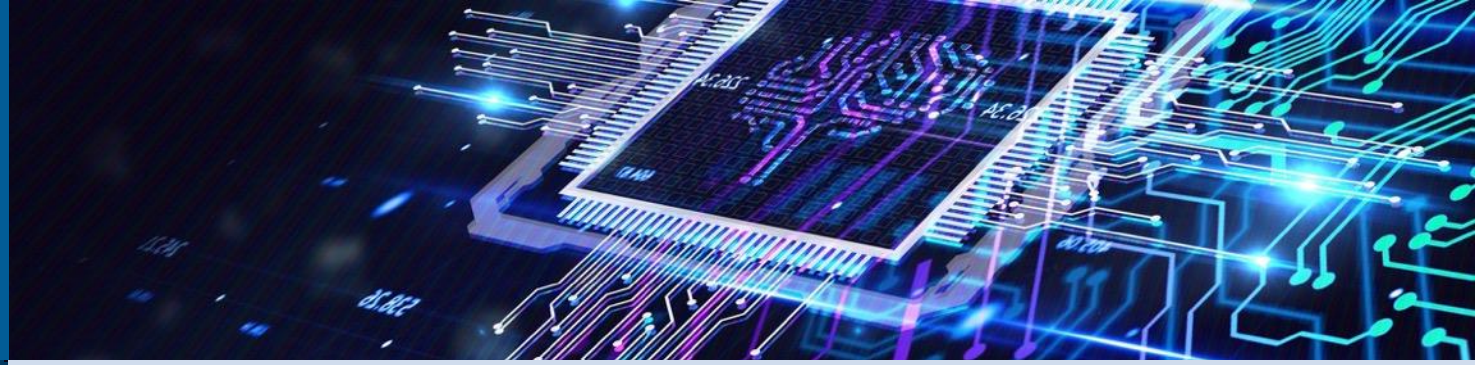
CSC

ICT Solutions for Brilliant Minds



Programming for LUMI GPUs

Dr. Jussi Heikonen, CSC – IT Center for Science Ltd.

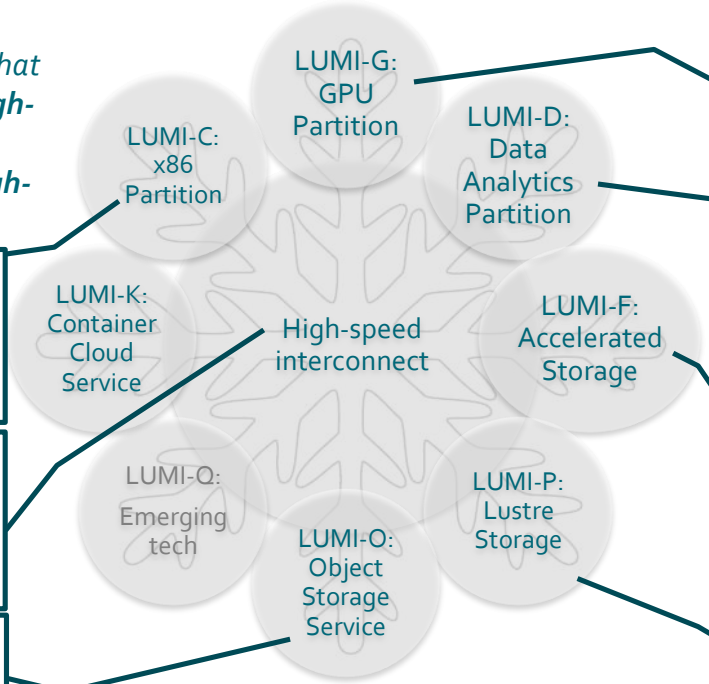
LUMI, the Queen of the North

LUMI is a Tier-0 ***GPU-accelerated supercomputer*** *that enables the convergence of* ***high-performance computing***, ***artificial intelligence***, *and* ***high-performance data analytics.***

Supplementary CPU partition: ~200,000 AMD EPYC Milan CPU cores

Possibility for combining different resources within a single run. HPE Slingshot technology.

30 PB encrypted object storage (Ceph) for storing, sharing and staging data



Tier-0 GPU partition: over 550 Pflop/s powered by AMD Instinct MI250X GPUs

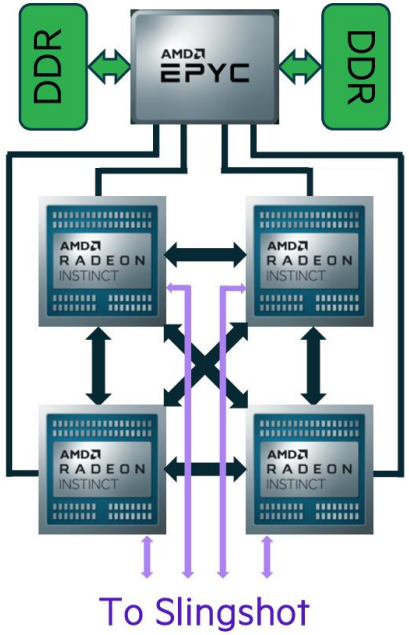
Interactive partition with 32 TB of memory and graphics GPUs for data analytics and visualization

7 PB Flash-based storage layer with extreme I/O bandwidth of 2 TB/s and IOPS capability. Cray ClusterStor E1000.

80 PB parallel file system

LUMI-G nodes

- The GPU partition will consist of 2560 nodes, each node with one 64 core AMD Trento CPU and four AMD MI250X GPUs.
- Each MI250X GPU consists of **two** compute dies, each with 110 compute units each, and each compute unit has 64 stream processors for a total of **14080 stream processors**.
- Each GPU node features four 200 Gbit/s network interconnect cards, i.e. has 800 Gbit/s injection bandwidth.
- The MI250X GPU comes with a total of 128 GB of HBM2e memory offering over 3.2 TB/s of memory bandwidth.
- A single MI250X card is capable of delivering **42.2 TFLOP/s** of performance in the **HPL** benchmarks.
- The committed Linpack performance of LUMI-G is 375 Pflop/s.



2560 nodes with 4 x MI250X + 1 x AMD Trento processor, 512 GB host memory and 512 GB device memory
4 x 200 Gbit/s NIC

LUMI timeline

#3 Top500
#3 Green500
#3 HPCG

System procurement:
November 2019 – August 2020

Data center preparation ready Q1/2020

2020

2021

2022

Pilot use
Extended beta testing

Q3/2022 Final configuration
GA 09/22

1st phase
Q4/2021
- LUMI-C, P, F, D
- Early Access Platform
GA 01/01/22

Q2/2022
- 12 cabinets of LUMI-G
- LUMI-C NIC upgrade

Rest of LUMI-G cabinets

GPU basics

- CPU = host, GPU = device
- CPU offloads computing intensive parts to GPUs
- The power of a GPU comes from a very high number of cores/threads/stream processors
  - MI250X has 14080 stream processors
- To utilize a GPU efficiently the programmer has to expose enough SIMD (Single Instruction Multiple Data) parallelism
- Memory access key to performance
  - CPU-GPU transfers are "slow"
  - Contiguous access & data reuse are essential
- Use GPU enabled libraries

GPU programming models

- CUDA
  - NVIDIA
  - C++
  - Fortran: CUDAFortran
  - Low level, high performance
- HIP
  - AMD version of CUDA, almost 1-1 mapping
  - Runs on Nvidia hw too (requires ROCm)
  - C++
  - Fortran: HIPFort (kernels in C++)

- OpenACC
  - NVIDIA (AMD)
  - C++, Fortran
  - Pragma/directive based
- OpenMP
  - NVIDIA & AMD
  - C++, Fortran
  - Pragma/directive based

- OneAPI/DPC++
  - Intel (open standard)
- (hip)SYCL, Kokkos, …
  - Generic

HIP (Heterogeneous-Compute Interface for Portability)

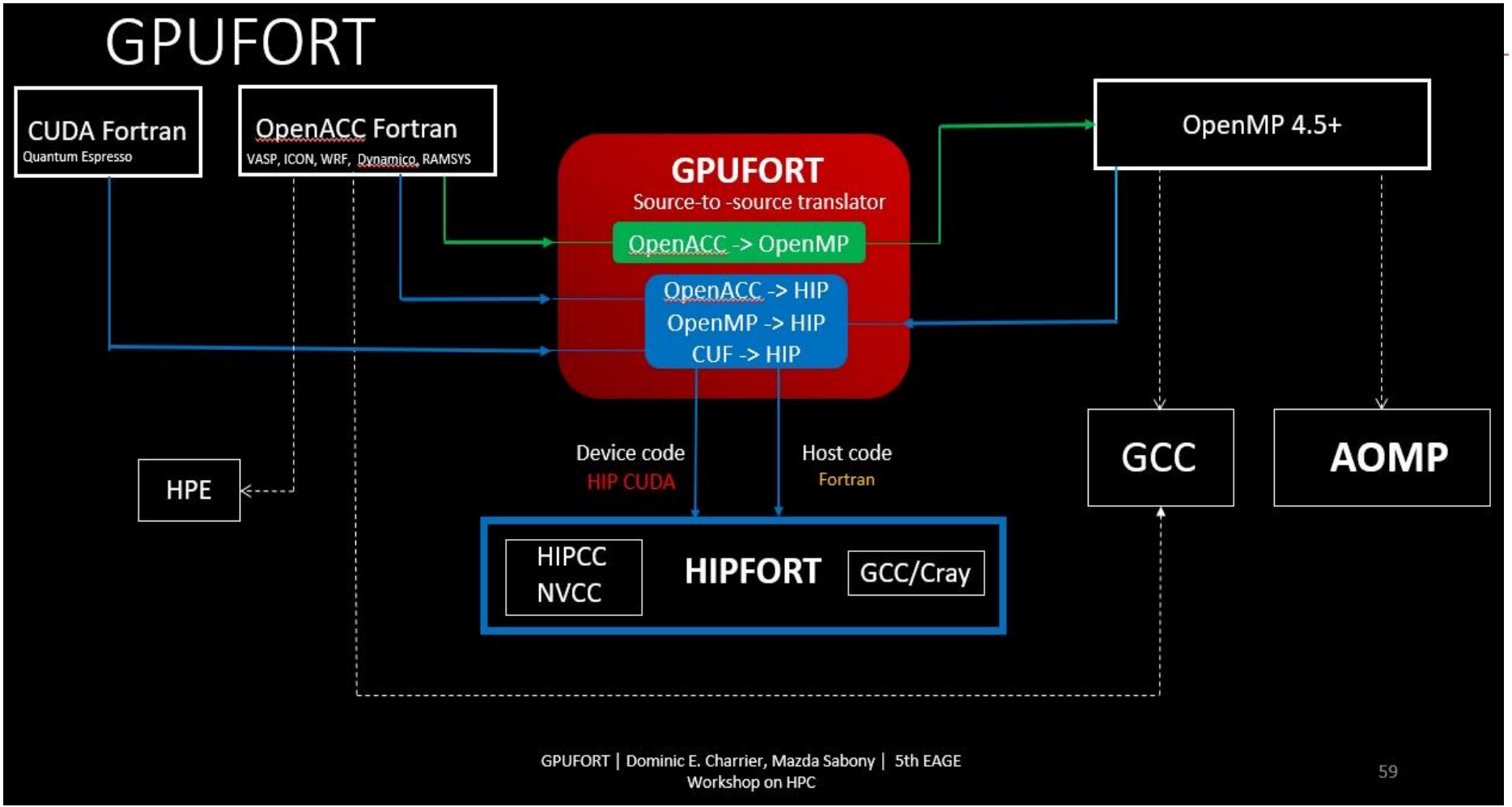
- HIP is basically AMD's version of CUDA: there is almost 1-1 correspondence between CUDA and HIP: If you know CUDA, you know HIP
  - Not all CUDA features are supported however
- For C++ only
- Hipify: source-to-source translator (a perl script) that converts CUDA to HIP (in place)
- Runs on Nvidia too but requires installing the HIP ROCm stack
- Hipfort: Fortran bindings for APIs (memory allocation and copies etc.)
  - Kernels need to be written in C++ and called through interfaces

LUMI/AMD GPU programming: starting from scratch

- Use C++
  - Largest choice of GPU programming paradigms, tools, libraries
  - New stuff typically becomes first available for C++
- For best performance
  - HIP or hipified CUDA
  - Takes some effort
  - Portability good (but requires ROCm on Nvidia hw)
- For easier programming
  - OpenMP (OpenACC will NOT be available for C++!)
  - Performance: may be lower than with HIP
  - Portability good
- Frameworks promise good performance & portability
  - hipSYCL (under construction),
  - Kokkos

LUMI/AMD GPU programming: porting existing GPU codes

- C++
  - CUDA: hipify
  - OpenMP: as is
  - OpenACC: Convert to OpenMP or HIP (GPUfort?)
  - (Kokkos: as is)
- Fortran
  - CUDAfortran
    - Convert memory allocation and copy etc. Calls to hipfort
    - Convert kernels to C++ and HIP (or CUDA & and hipify)
    - C++ kernels are callable from Fortran
  - OpenMP: as is
  - OpenACC: Version 2.7 supported currently
  - Gpufort? Next slide …


GPUFORT
CUDA Fortran
Quantum Espresso
OpenACC Fortran
VASP, ICON, WRF, Dynamico, RAMSYS
GPUFORT
Source-to -source translator
OpenACC -> OpenMP
OpenACC -> HIP
OpenMP -> HIP
CUF -> HIP
OpenMP 4.5+
HPE
Device code
HIP CUDA
Host code
Fortran
GCC
AOMP
HIPCC
NVCC
HIPFORT
GCC/Cray
GPUFORT | Dominic E. Charrier, Mazda Sabony | 5th EAGE Workshop on HPC
59

LUMI/AMD GPU programming: porting existing CPU codes to GPUs

- C++: See Starting from scratch
- Fortran
  - For performance: HIP & hipfort
    - C++ kernels/device code
    - Portability requires ROCm on Nvidia hw
  - For easier programming: OpenMP
    - Portability good
  - OpenACC not recommended due to limited support
  - Kokkos & hipSYCL?

Portability considerations

- C++ easier than Fortran
- OpenMP is ”easy” but there may be some performance hit
- CUDA & hipify
- GPUfort may make things easier
- Keep device code isolated to minimize porting effort

We are hiring!

- **https://www.csc.fi/en/careers**


facebook.com/CSCfi


twitter.com/CSCfi


linkedin.com/company/csc---it-center-for-science


github.com/CSCfi