OPENFABRICS ALLIANCE

Los Alamos National Lab - Perspectives on OFI libfabric

Howard Pritchard

ANNUAL INTERNATIONAL OPENFABRICS SOFTWARE DEVELOPERS' WORKSHOP

#OFADevWorkshop

LA-UR-15-22243

What We'll Cover

- Background – Trinity/TN8 systems
  - Knights Landing Processor
  - Cray XC Interconnect
- Open MPI – yet more about this
- Libfabric at LANL – the why and the how
- Some observations

The TN8 System: Cori

- Cori will support the broad Office of Science research community and begin to transition the workload to more energy efficient architectures
- Cray XC system with over 9300 Intel Knights Landing compute nodes –mid 2016
  - Self-hosted, (not an accelerator) many core processor with over 60 cores per node, 4 HTs per core
  - On-package 16 GB high-bandwidth memory
  - 96 GB DDR memory per node
- Data Intensive Science Support
  - 10 Haswell processor cabinets to support data intensive applications – Summer 2015
  - NVRAM Burst Buffer to accelerate data intensive applications
  - 28 PB of disk, >700 GB/sec I/O bandwidth
- Robust Application Readiness Plan
  - Outreach and training for user community
  - Application deep dives with Intel and Cray
  - 8 post-docs integrated with key application teams

Image source: Wikipedia

System named after Gerty Cori, Biochemist and first American woman to receive the Nobel prize in science.

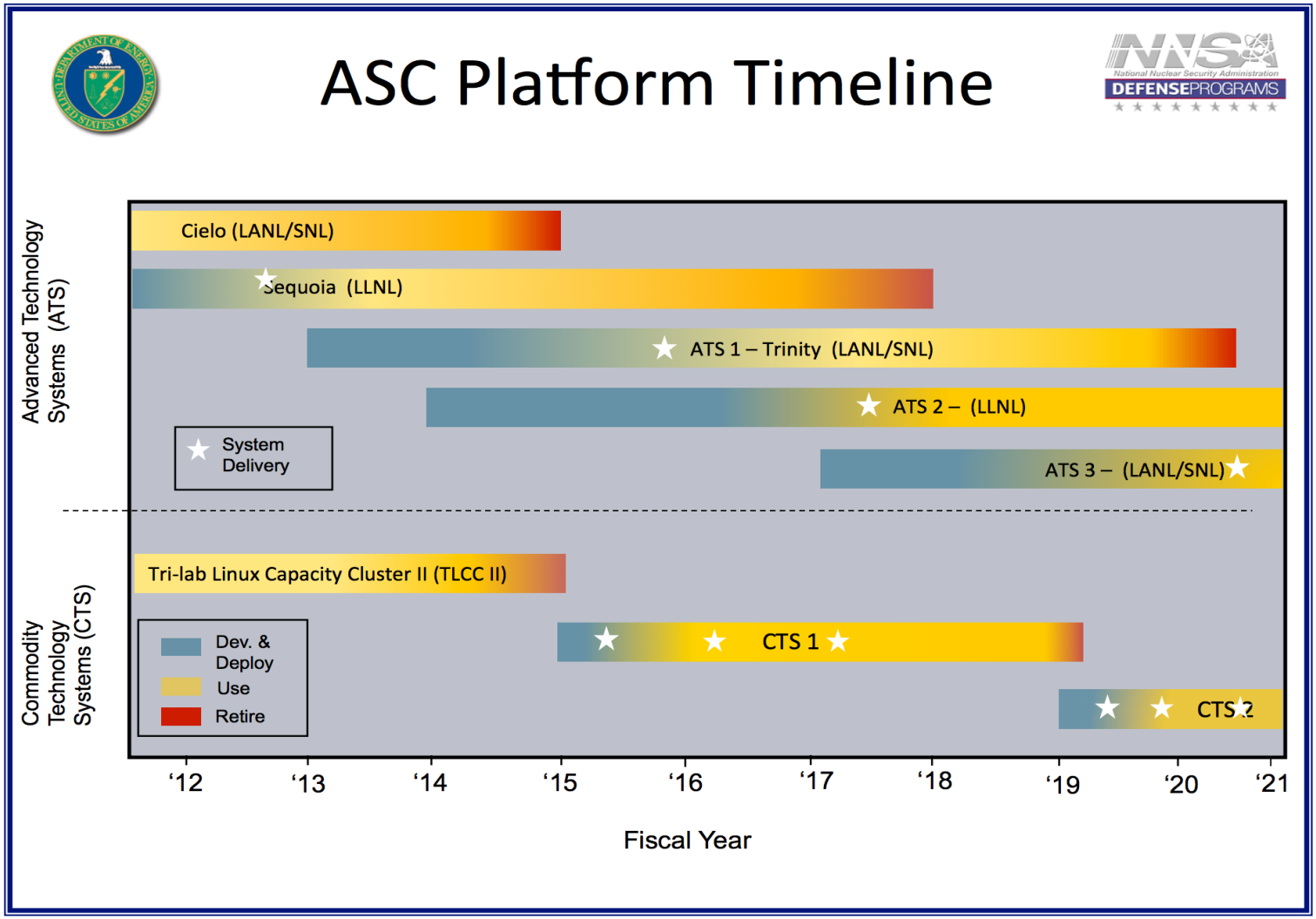
KNL and MPI Applications

- NERSC experience with KNL precursor KNC shows that for good performance on MIC architecture, essential to use at least two threads/core
- Multiple levels of thread parallelism likely necessary
- Forget about running 1 MPI rank/core if performance and/or scalability is important
- Significant implications for MPI – x86_64 business-as-usual won't work

Trinity/Cori Interconnect

- Cray XC Aries – dragonfly network (3 stages)
- Custom network using a custom, low-level network API
- Will be reaching end of life by the time Trinity/Cori installed
- Likely no follow-on network that uses the Aries low-level network API - GNI

Beyond Trinity

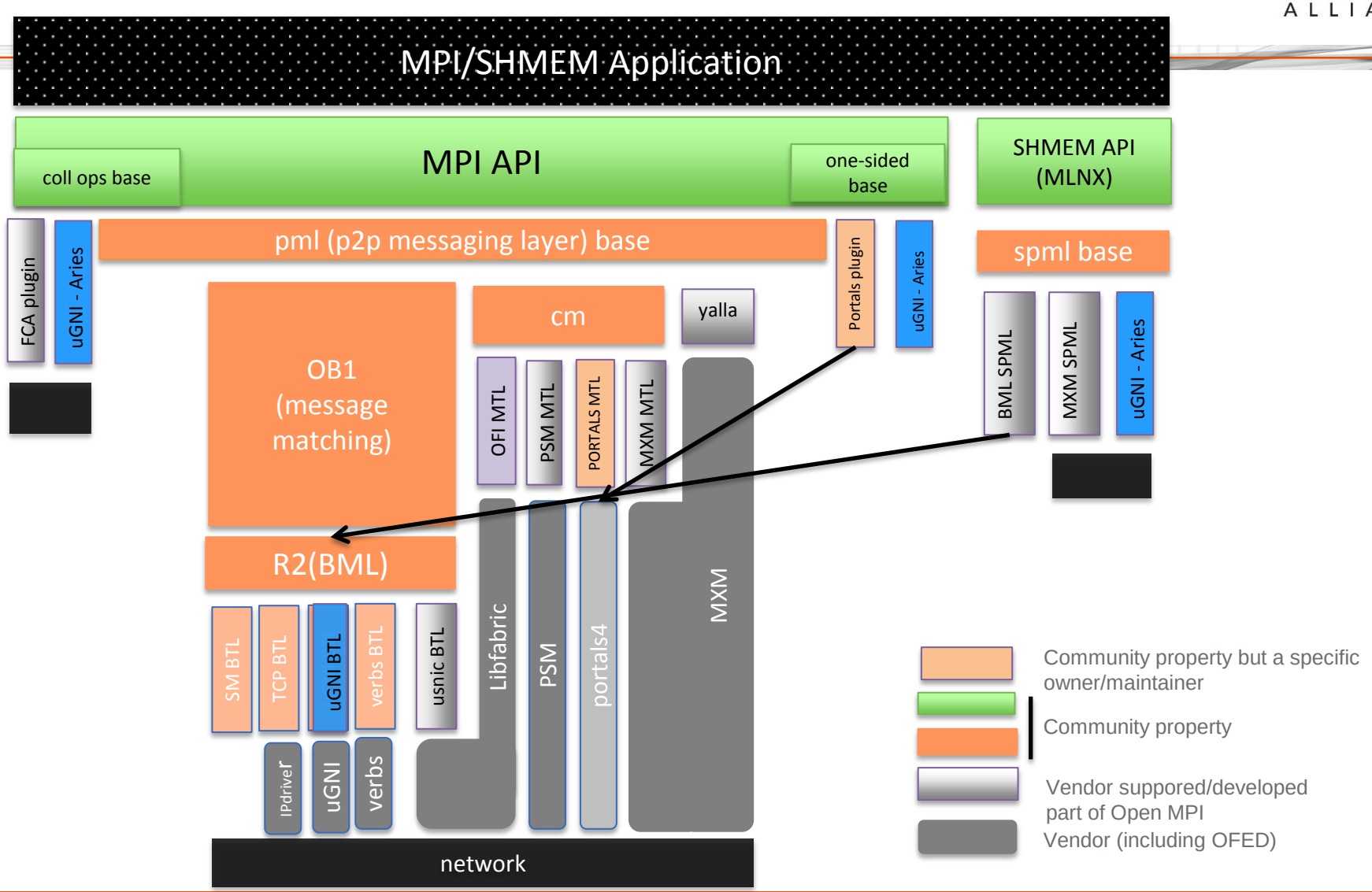


More on Open MPI

Open MPI Requirements for Trinity

- Thread – hot support: multiple threads in an MPI process can use MPI almost as effectively as the case of a single thread per MPI rank
- Asynchronous progress – MPI can make progress on messages (including non-blocking collectives) without an application thread being in an MPI call
- Utilize the Aries collective engine (particularly for MPI_Allreduce)
- Memory scalability – reduce this footprint from the existing uGNI plugin.

Open MPI Project Code Structure



Libfabric at LANL – The Why and The How

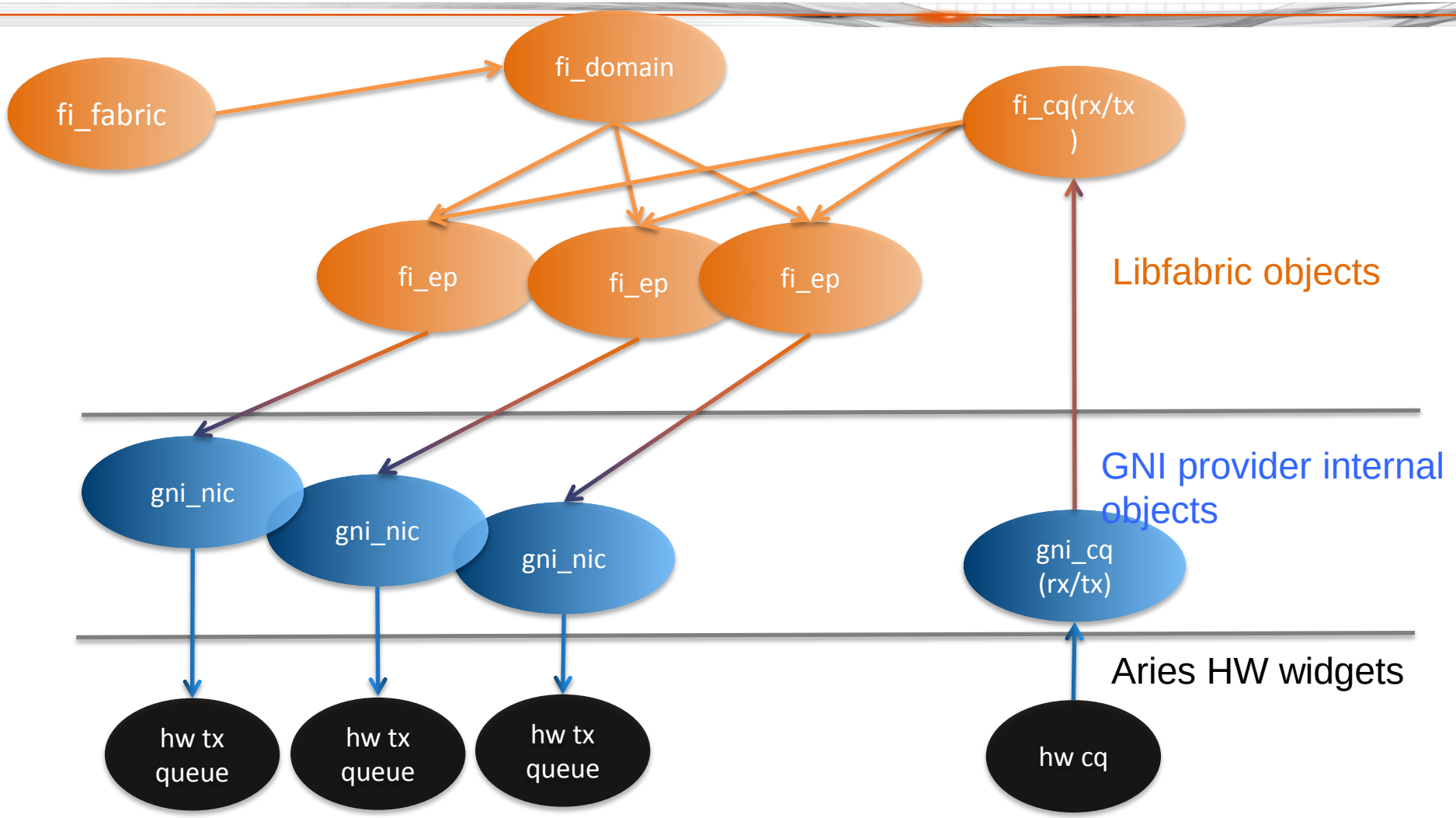
Libfabric – The Why

- A libfabric provider for Trinity/TN8 should allow for more code reuse inside Open MPI when transitioning to post-Trinity systems
- The libfabric provider could be used for applications other than Open MPI on Trinity/TN8
- Adding functionality within the libfabric provider doesn't require buy in from as many players
- Current LANL TLCC2 systems use Infinipath, can do testing at scale with Open MPI and the existing OFI MTL

Libfabric – the how

- Collaborative agreement with Cray to develop a GNI provider
- Open source fork of libfabrichttps://github.com/ofi-cray/libfabric-cray
- Initial implementation will support FI_EP_RDM endpoint type
- Designing to meet goals of MPI thread-hot, asynchronous progress, memory footprint scalability, Aries specific features like collective engine
- Use fi_op ops_open method for specific Aries/GNI functionality

Libfabric – The How



Some observations

- Libfabric interface is proving flexible enough to map well to GNI API for FI_EP_RDM
- GNI provider could use a more flexible mr_key approach (filing an issue)
- Definite need for more unit tests and a framework for testing on clusters with batch schedulers, etc.
- Github mechanics – great and could be better

Thank You

OPENFABRICS ALLIANCE

ANNUAL INTERNATIONAL OPENFABRICS SOFTWARE DEVELOPERS' WORKSHOP