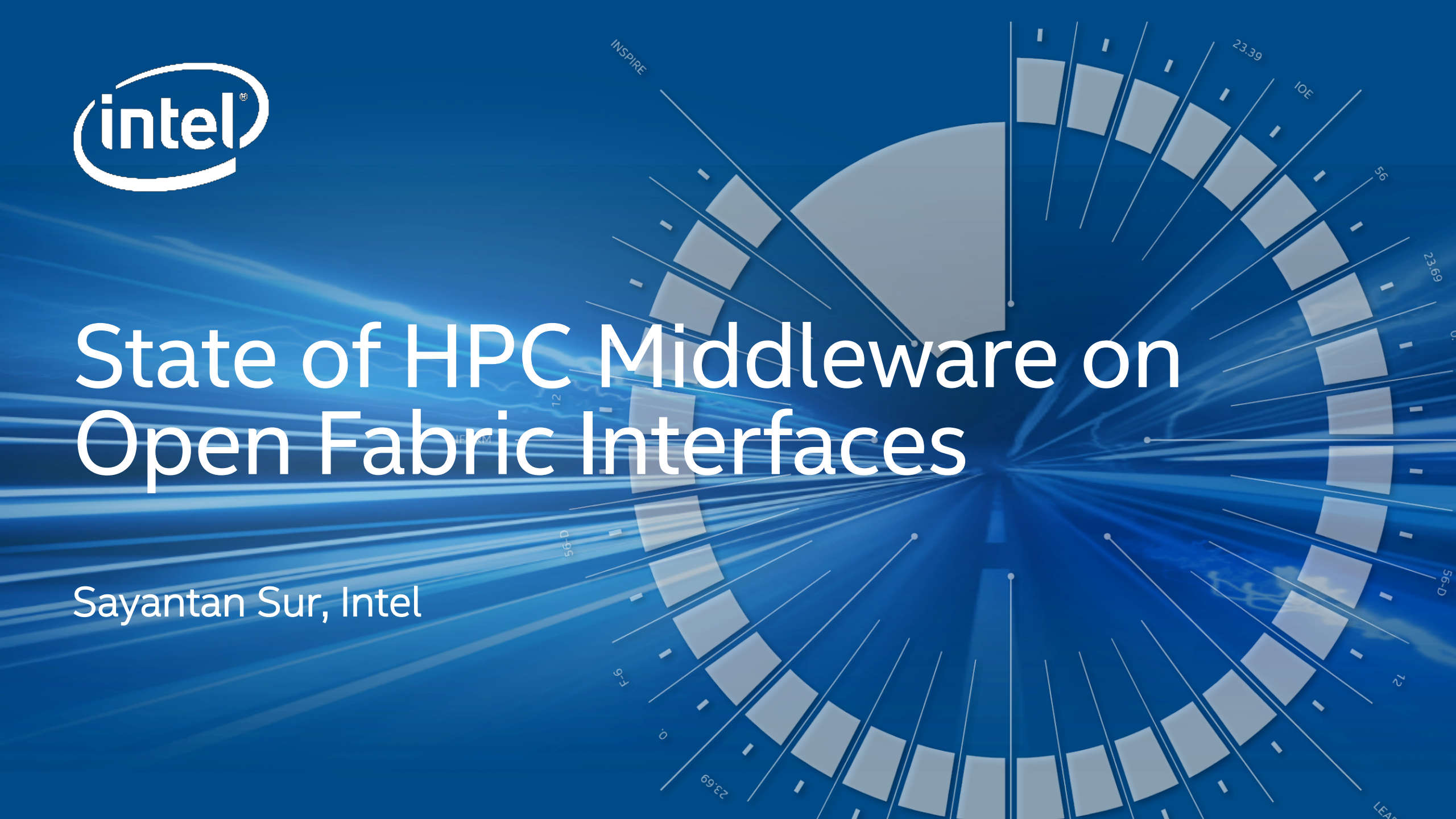


State of HPC Middleware on Open Fabric Interfaces

Sayantan Sur, Intel

Legal Notices and Disclaimers

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY RELATING TO SALE AND/OR USE OF INTEL PRODUCTS, INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT. Intel products are not intended for use in medical, life-saving, life-sustaining, critical control or safety systems, or in nuclear facility applications.

Intel products may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Intel may make changes to dates, specifications, product descriptions, and plans referenced in this document at any time, without notice.

This document may contain information on products in the design phase of development. The information herein is subject to change without notice. Do not finalize a design with this information.

Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

Intel Corporation or its subsidiaries in the United States and other countries may have patents or pending patent applications, trademarks, copyrights, or other intellectual property rights that relate to the presented subject matter. The furnishing of documents and other materials and information does not provide any license, express or implied, by estoppel or otherwise, to any such patents, trademarks, copyrights, or other intellectual property rights.

Wireless connectivity and some features may require you to purchase additional software, services or external hardware.

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, visit Intel Performance Benchmark Limitations

Intel, the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Other names and brands may be claimed as the property of others.

Legal Disclaimer & Optimization Notice

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Optimization Notice

Intel’s compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804

Open Fabric Interfaces

User-centric interfaces lead to innovation and adoption

Open Source

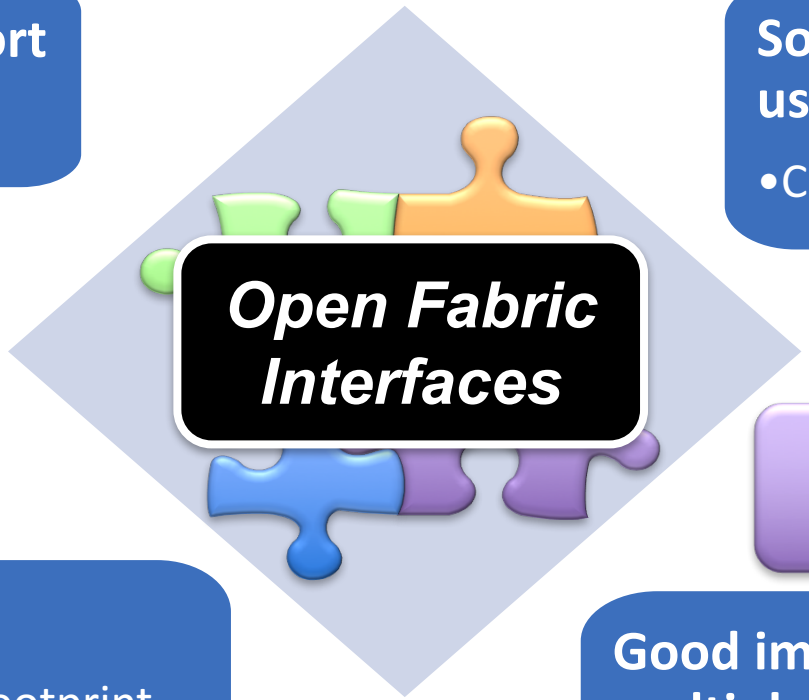
Inclusive development effort

- App and HW developers

User-Centric

Software interfaces aligned with user requirements

- Careful requirement analysis



Scalable

Optimized SW path to HW

- Minimize cache and memory footprint
- Reduce instruction count
- Minimize memory accesses

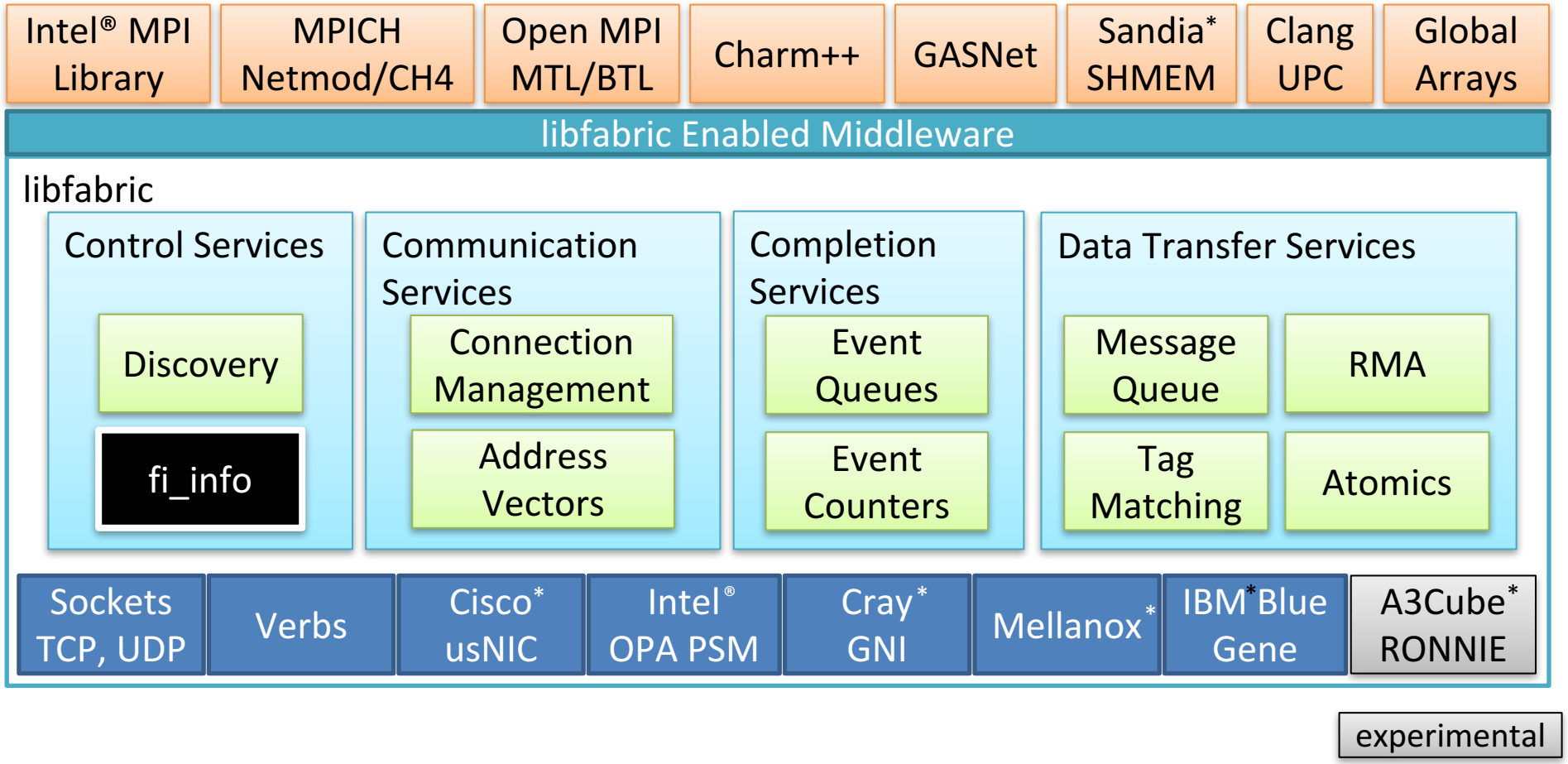
Implementation Agnostic

Good impedance match with multiple fabric hardware

- InfiniBand, iWarp, RoCE, raw Ethernet, UDP offload, Omni-Path, GNI, BGQ, …

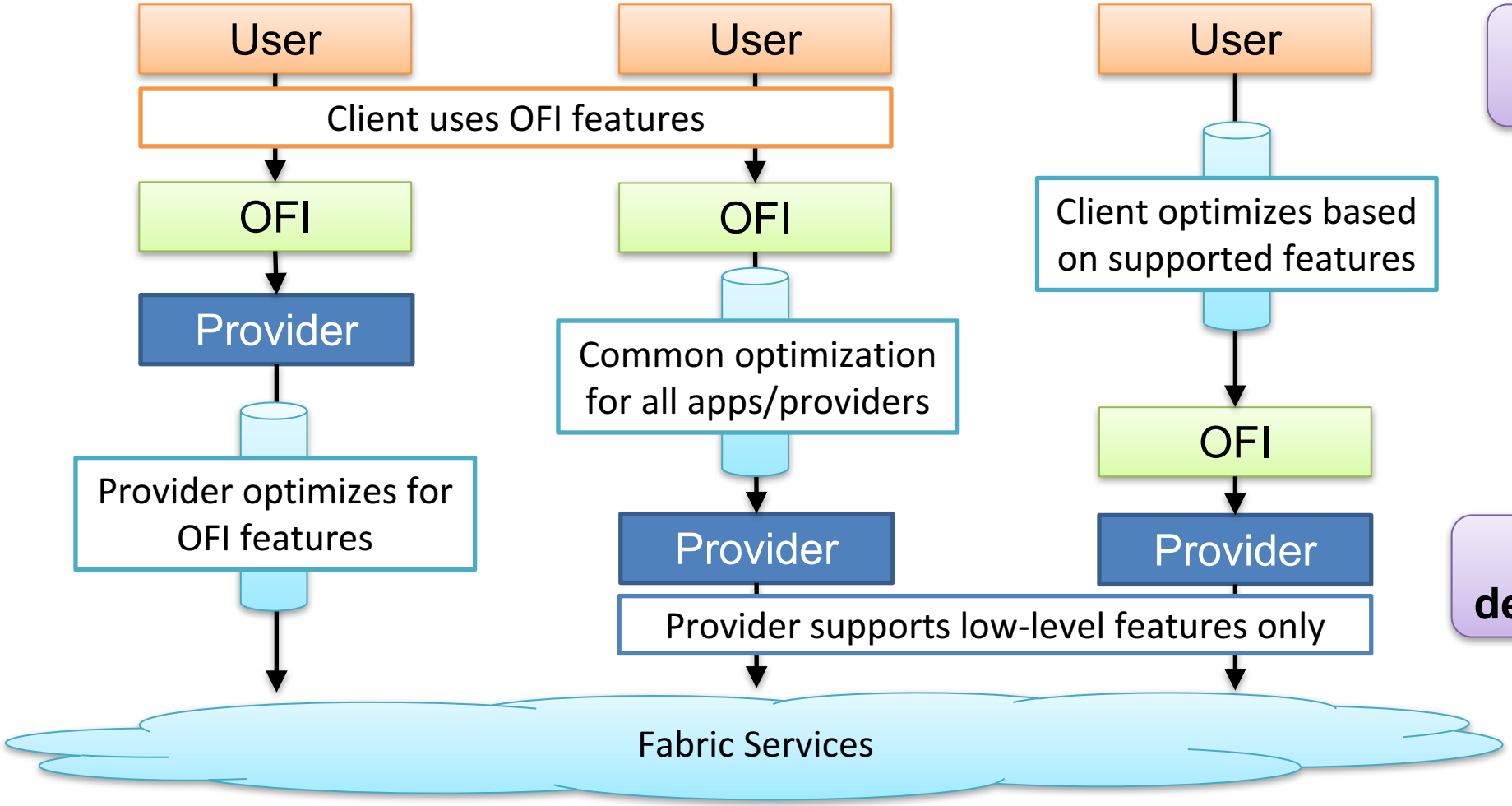
OFI HPC Community

Sampling of HPC Middleware already targeting OFI



OFI Software development strategies

One size does not fit all

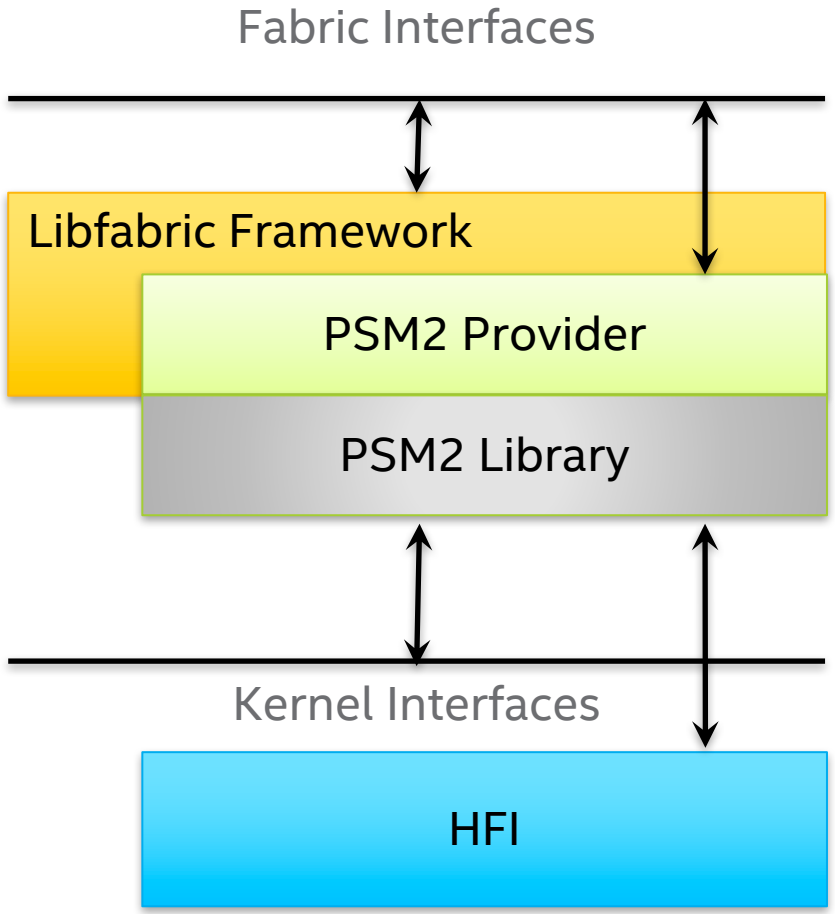


Linux, FreeBSD, OS X, Windows

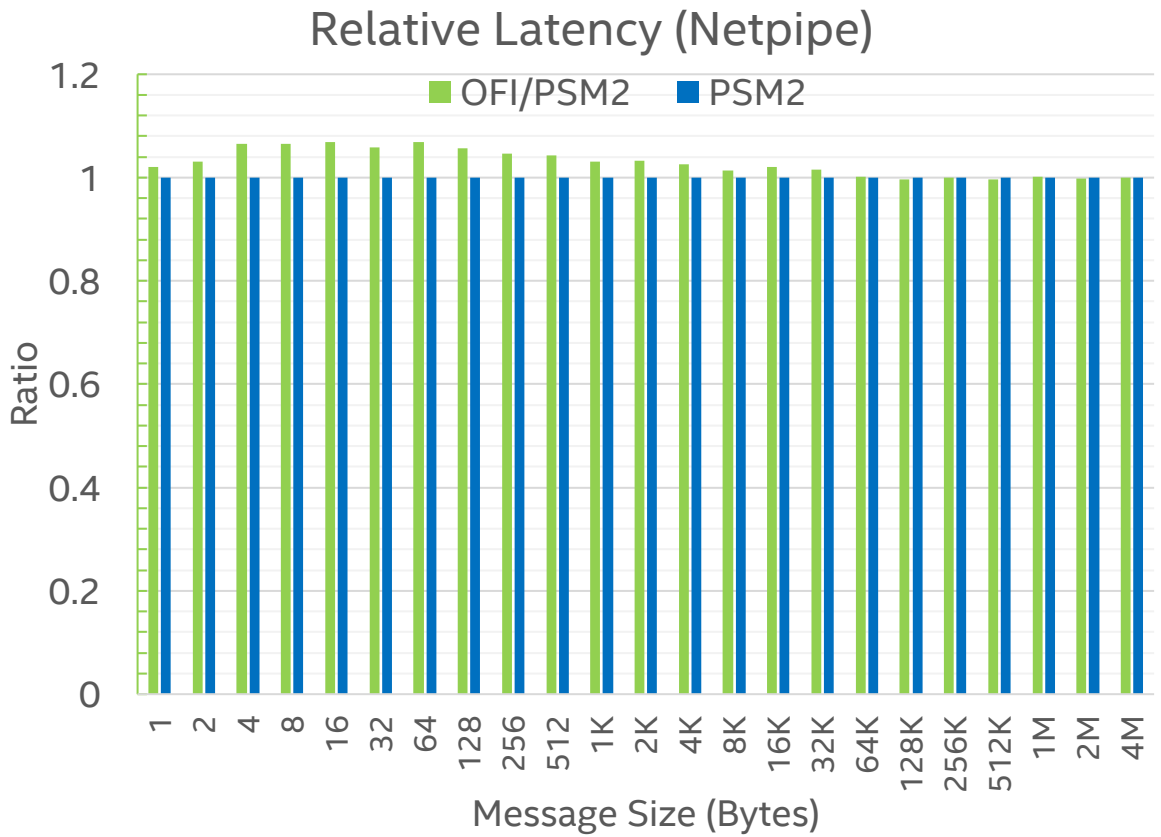
TCP and UDP development support

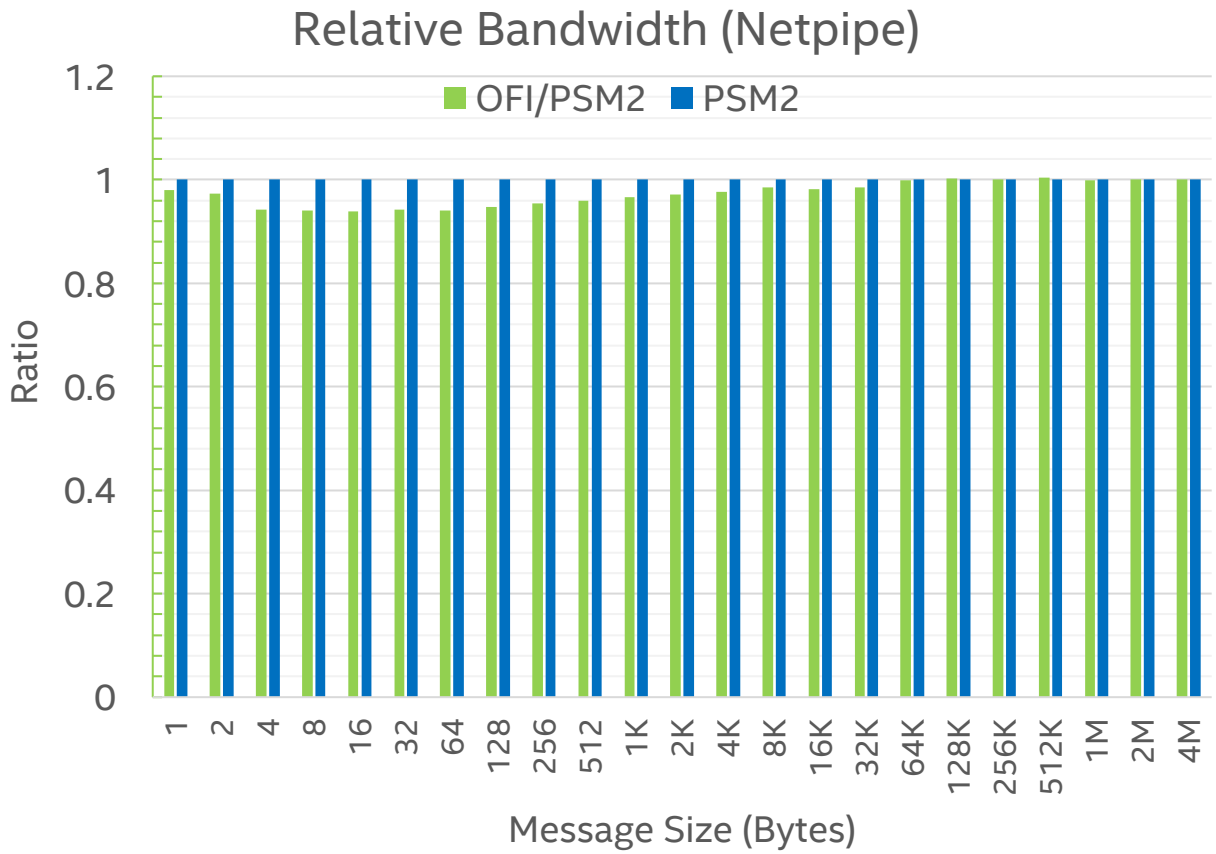
Libfabric on Intel® OPA

- Libfabric PSM2 provider uses the public PSM2 API
- Semantics matches with that of compute middleware such as MPI
  - PSM2 EP maps to HW context, Each EP associated with matched queue
- Regular Endpoints share a single HW context
- Scalable Endpoints provide independent HW contexts
- PSM2 library uses PIO/Eager for small message and DMA/Expected for Bandwidth for large messages
- Current focus is full OFI functionality
- Performance optimizations are possible to reduce call overheads, and provide better semantic mapping of OFI directly to HFI



Intel® OPA/PSM Provider Performance





Performance is comparable for most messages

- 6.7% overhead for 64B latency, and 6.1% overhead for 16B bandwidth
- Optimizations are ongoing

Intel Xeon E5-2697 (Ivy Bridge) 2.6GHz; Intel Omni-Path; RHEL 7.3; libfabric 1.5.0 alpha; IFS 10.5.0.0.115

MPI/OFI Design Principles

Directly map MPI constructs to OFI features as much as possible

Focus on optimizing communication calls

- Avoid branch, function calls, cache misses, memory allocation

MPI	OFI
MPI_Send, MPI_Recv, etc.	fi_tsend, fi_trecv
Window	Memory region (fi_mr(3))
Comm. calls (MPI_Put, MPI_Get, etc…)	fi_write, fi_read, … (fi_rma(3))
Atomic comm. calls (MPI_Accumulate, etc…)	fi_atomic, … (fi_atomic(3))

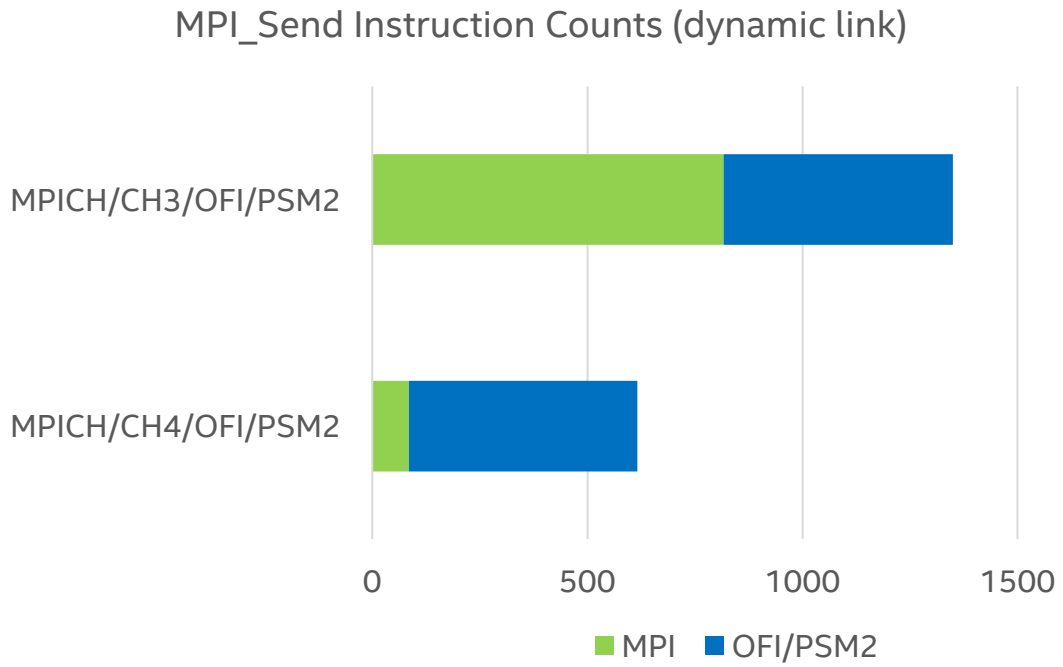
MPICH-OFI

Open-source implementation based on MPICH

- Uses the new CH4 infrastructure
  - Co-designed with MPICH community
- Targets existing and new fabrics via Open Fabrics Interface (OFI)
  - Ethernet/sockets, Intel® Omni-Path, Cray Aries*, IBM BG/Q*, InfiniBand*
- OFI is intended as the default communication interface for future versions MPICH

Improving on MPICH/CH3 Overheads

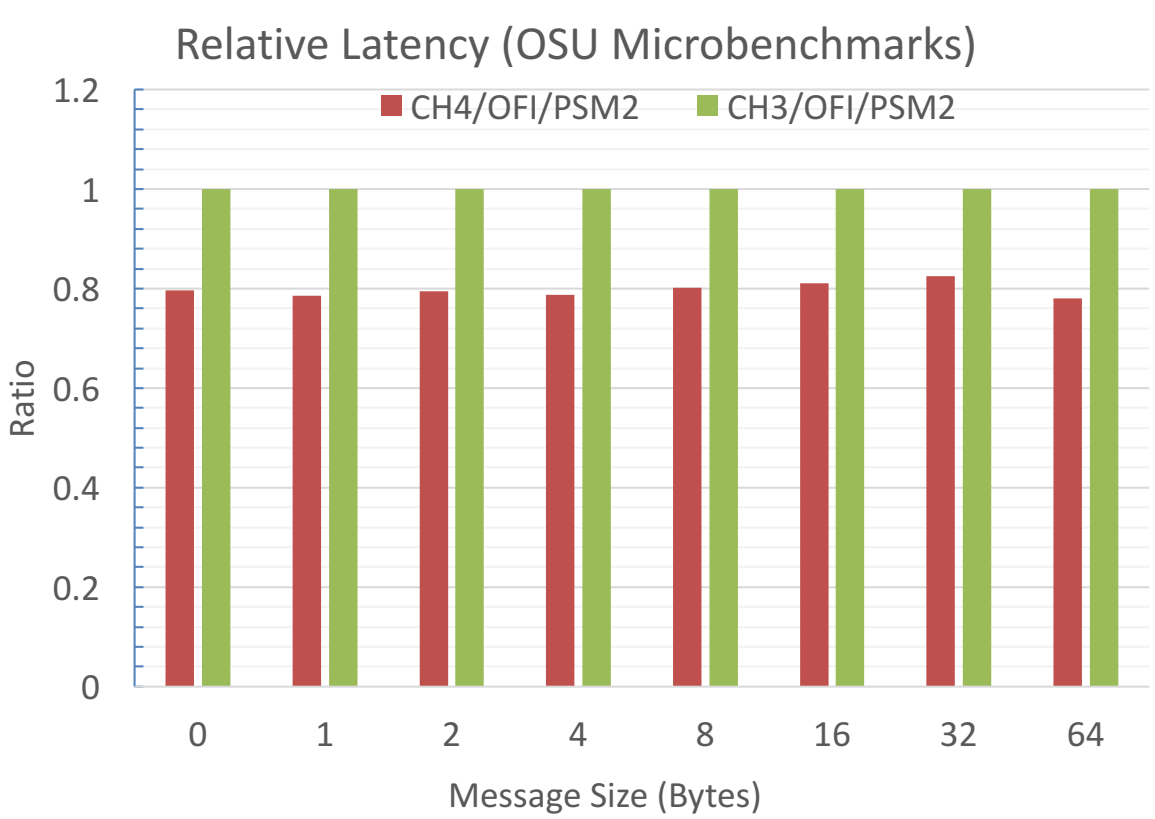
CH4 significantly reduces MPI instructions

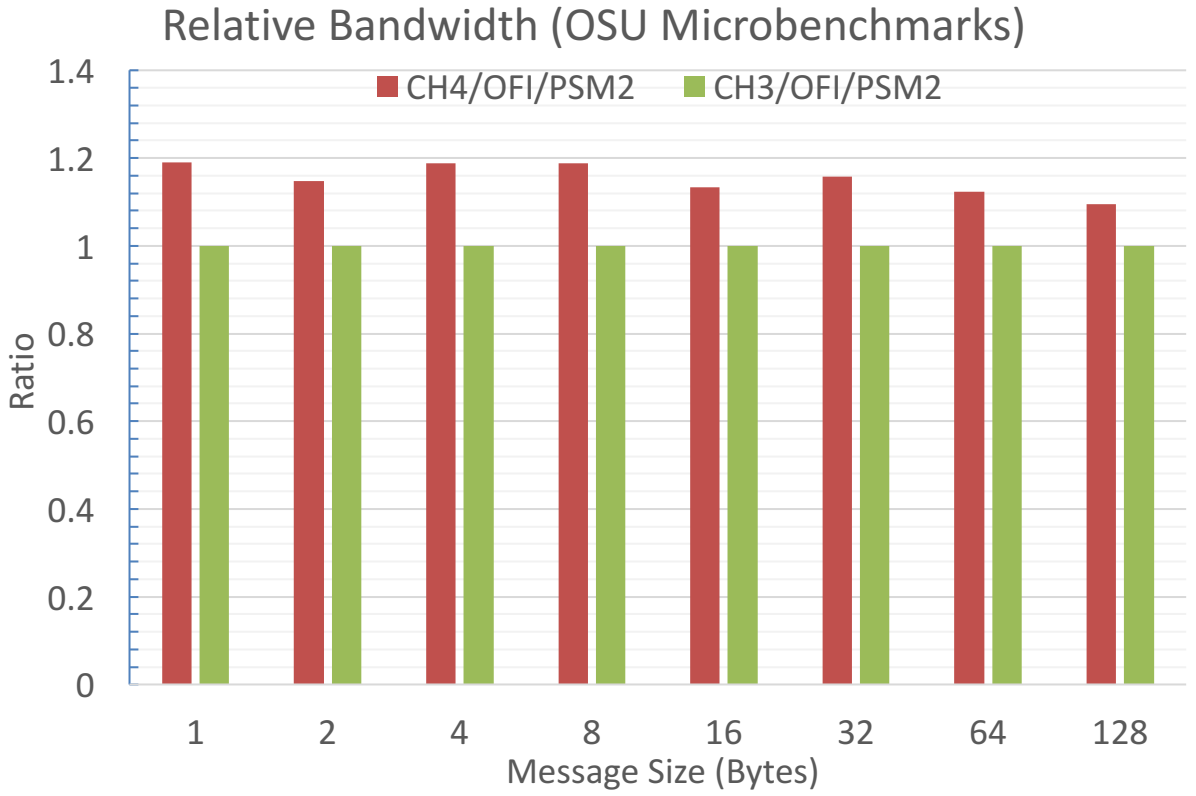


MPICH/CH4:

- 35 instructions with static build
- 85 instructions with dynamic build
- Data collected using GCC v5.2.1on Xeon E5-2697 (Haswell)

MPICH Performance Comparison (KNL + OPA)





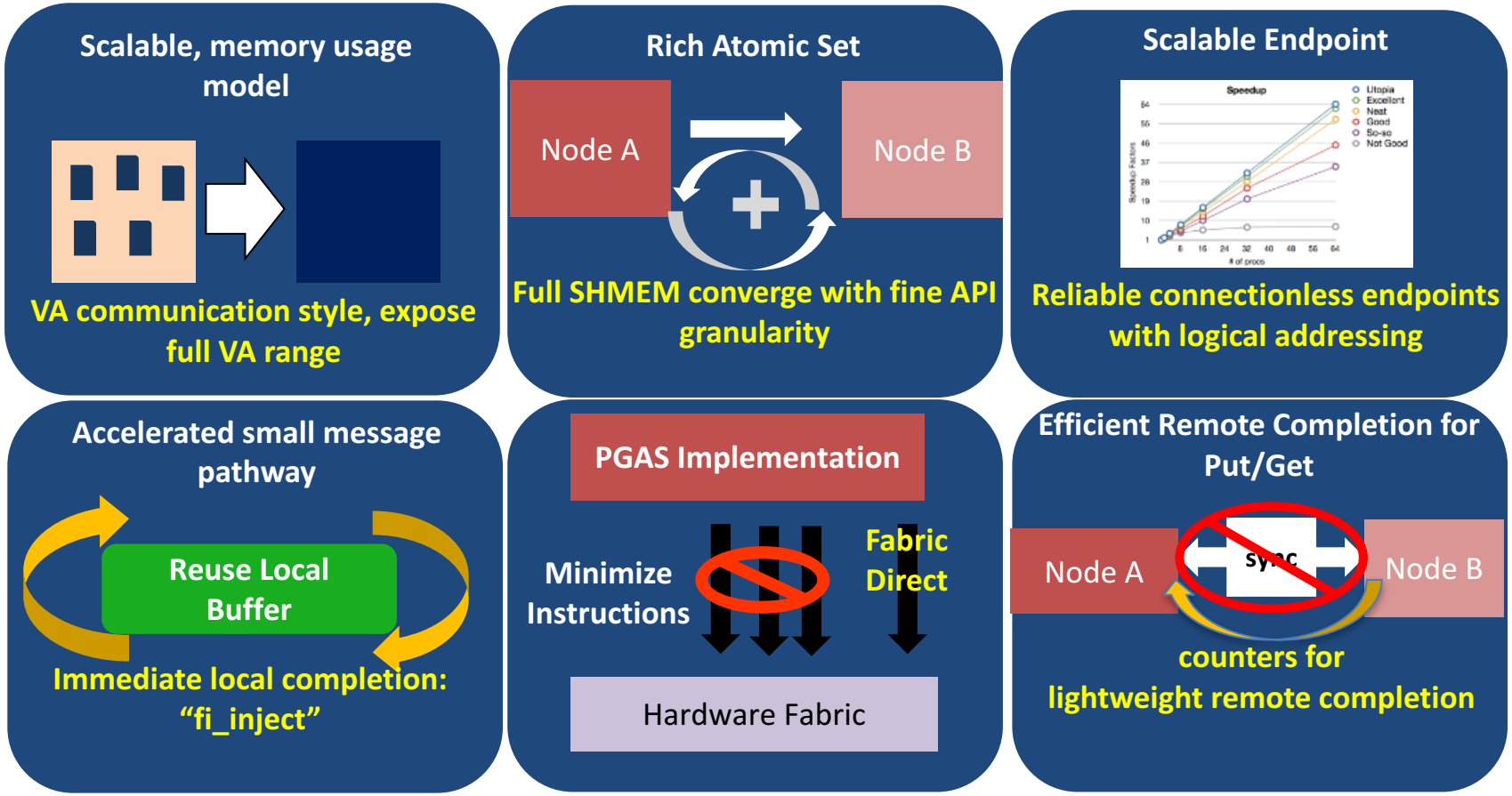
Around 20% reduction in latency, gains in bandwidth for small messages

Intel Xeon Phi 7250 (Knights Landing) 1.4GHz; Intel Omni-Path; RHEL 7.3; GCC v5.3.1; OSU Micro-benchmarks 5.3

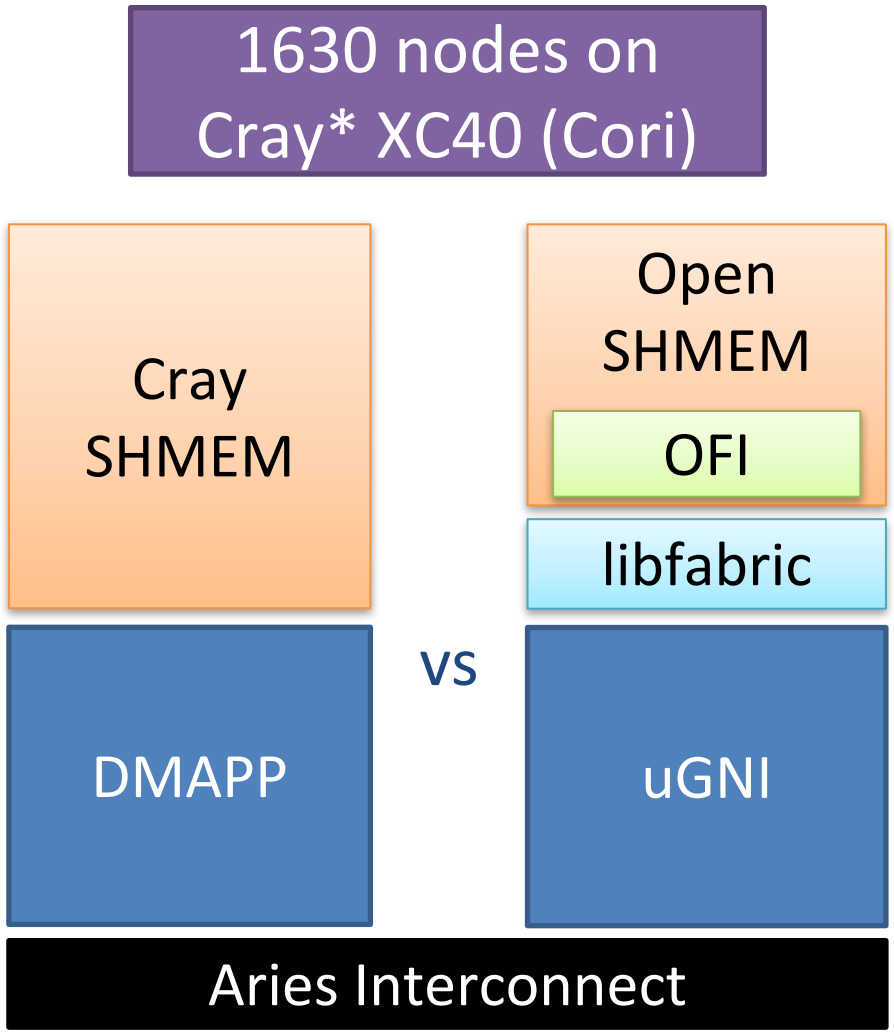
OpenSHMEM mapping on OFI

Design and implementation presented at OpenSHMEM workshop 2016

https://rd.springer.com/chapter/10.1007/978-3-319-50995-2_7



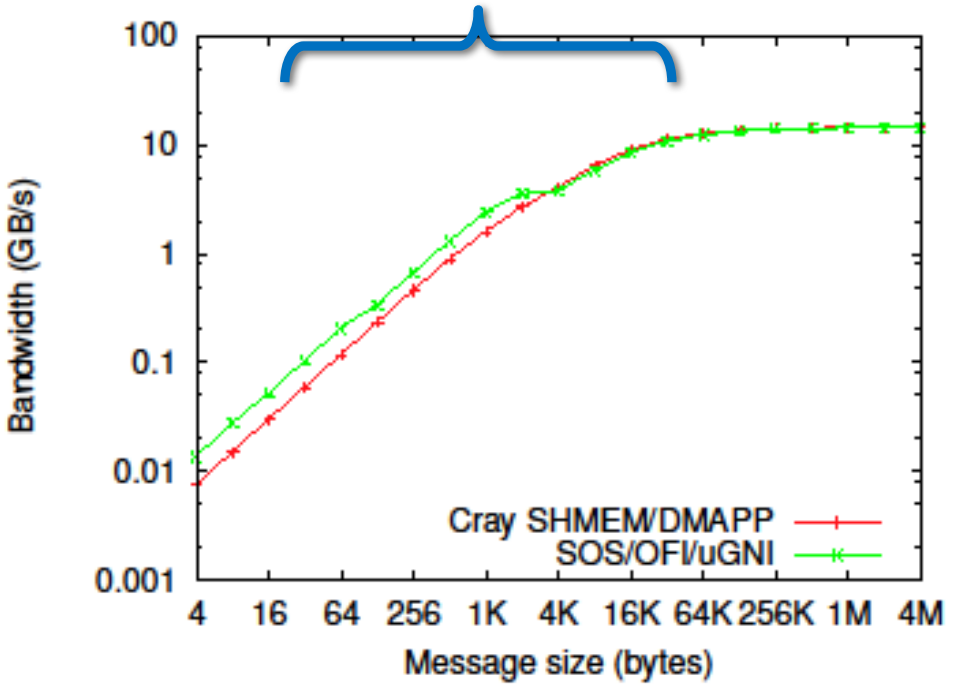
SHMEM/OFI Test Environment



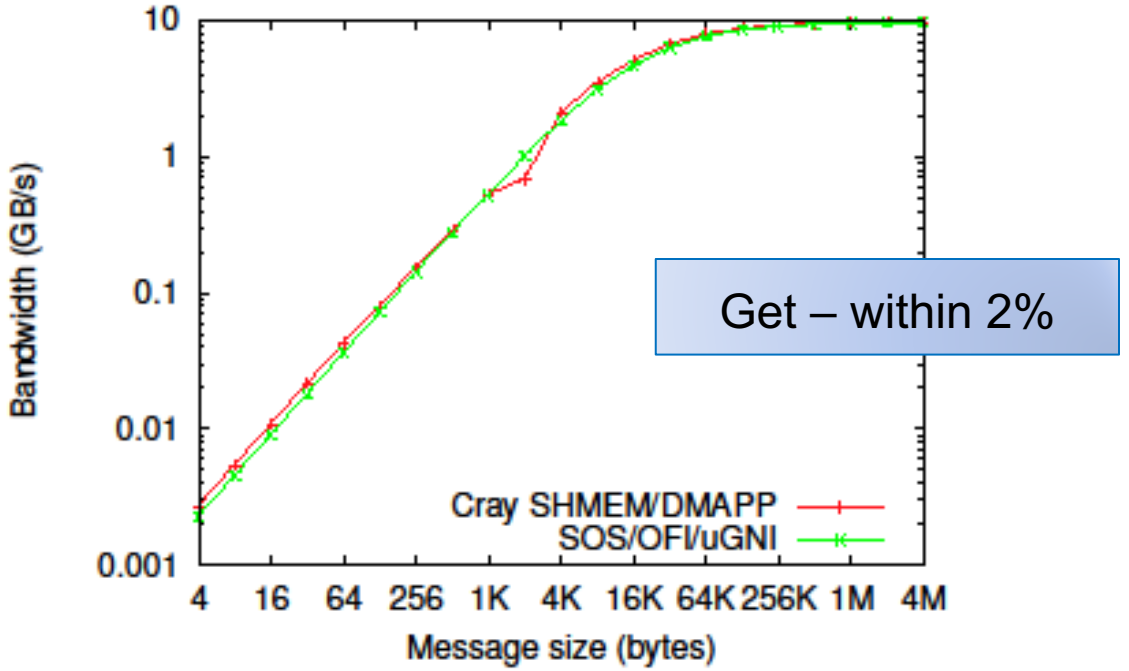
- **All tests run on CORI at NERSC**
- **Cray* SHMEM**
  - Cray* Aries, Dragonfly* topology
  - CLE (Cray* Linux*), SLURM*
  - DMAPP
    - Designed for PGAS
    - Optimized for small messages
- **Sandia* OpenSHMEM / libfabric**
  - uGNI
    - Designed for MPI and PGAS
    - Optimized for large messages

SHMEM Performance on Cray* XC40

Put – up to 61% improvement



Blocking Get/Put B/W



Get – within 2%

Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit http://www.intel.com/performance. Configuration: CORI @ NERSC

State of GASNet Support

Currently GASNet/OFI supports Intel® True Scale Architecture, Intel® Omni-Path Architecture, and TCP/IP.

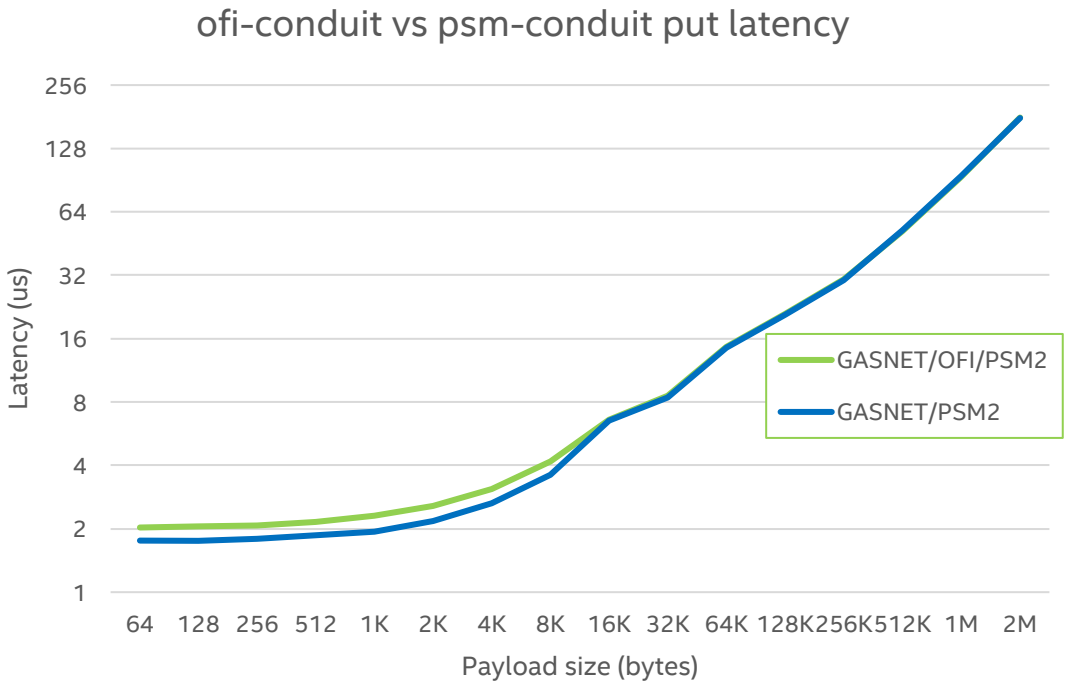
- Experimental support for Cray* XC systems via GNI provider.
- Blue Gene/Q provider supports the implementations requirements, but has not been tested yet.

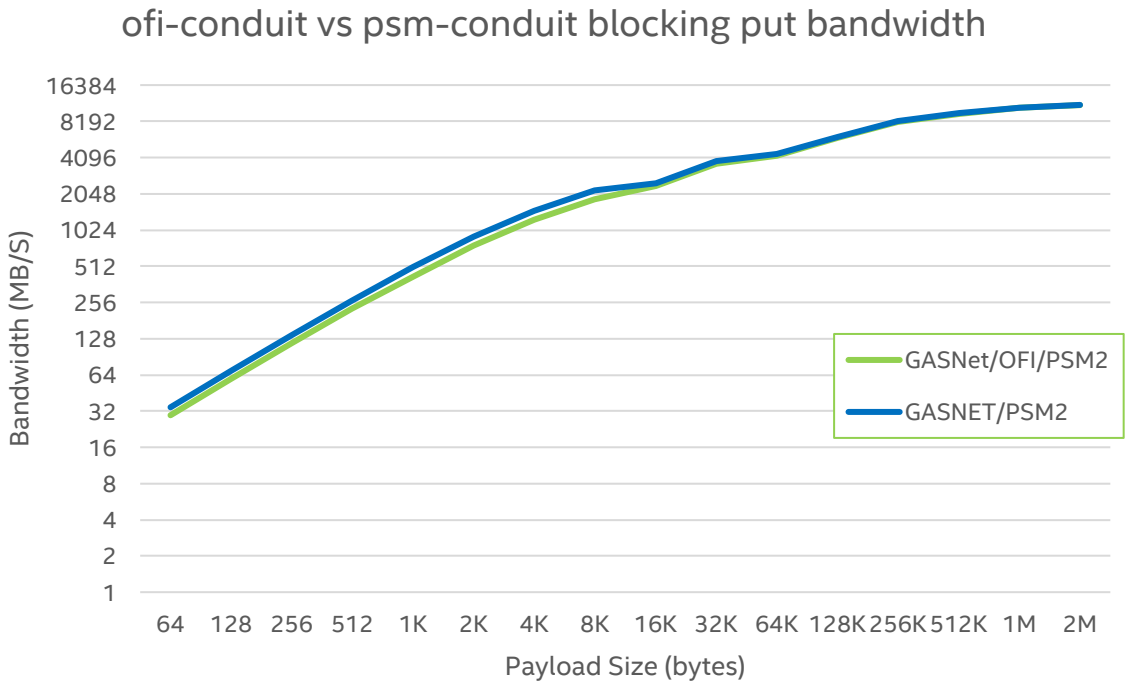
Provider Requirements to support GASNet

- FI_EP_RDM
- (Preferred) FI_MR_SCALABLE, FI_MR_BASIC
- FI_MSG, FI_RMA
- FI_MULTI_RECV

Support on platforms like verbs may be easily achieved through utility providers

GASNet Performance Comparisons





Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit http://www.intel.com/performance. Configuration: Intel(R) Xeon(TM) CPU E5-2697 v3 @ 2.60GHz, RHEL 7.3, libfabric 1.4.0, GASNet 1.28.2, libpsm2-10.2.58-1

GASNet Performance Discussion

At large message sizes, performance is more or less the same

For small message sizes there is a disparity

Reason: native psm-conduit is using different completion mechanism

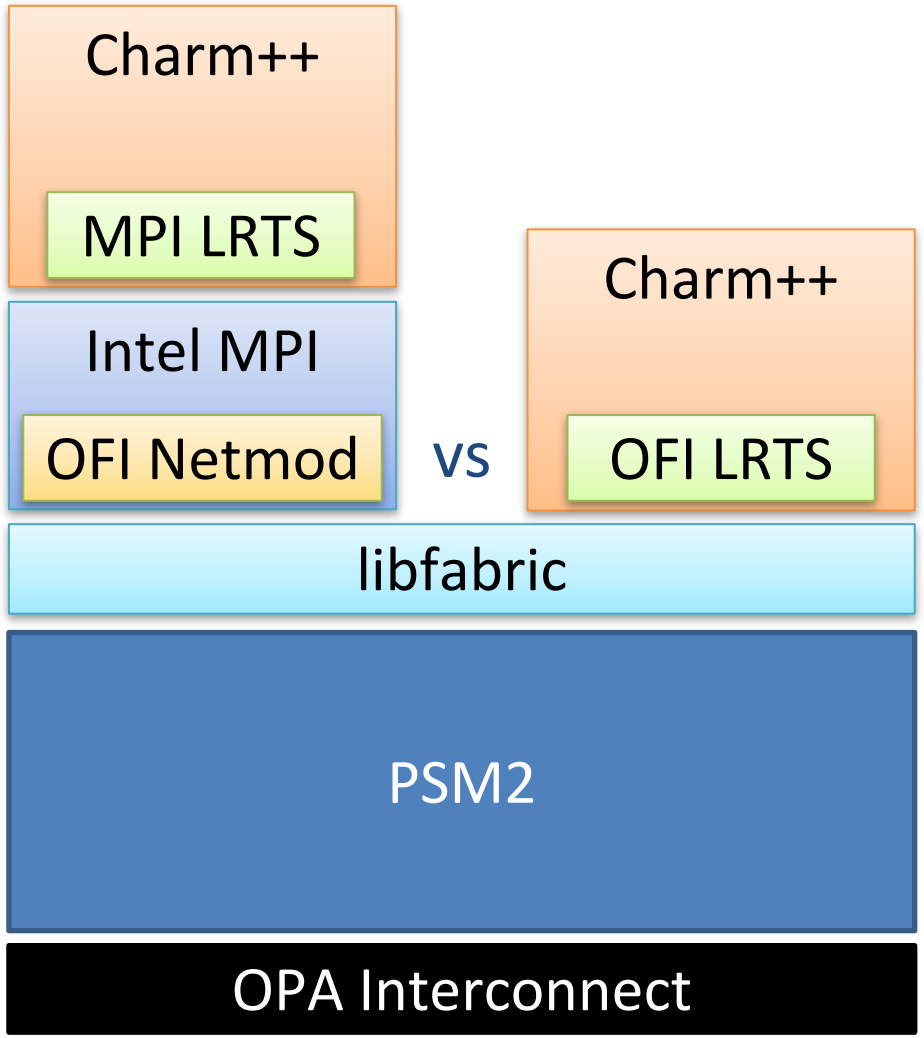
```
psm2_error_t
psm2_am_request_short(psm2_epaddr_t epaddr, psm2_handler_t handler,
        psm2_amarg_t *args, int nargs, void *src,
        size_t len, int flags,
        psm2_am_completion_fn_t completion_fn,
        void *completion_ctxt);
```

Callback function executed when remote completion is finished

- Pros: Better latency, reduces overhead related to completion queue processing
- Cons: Does not return error/success information

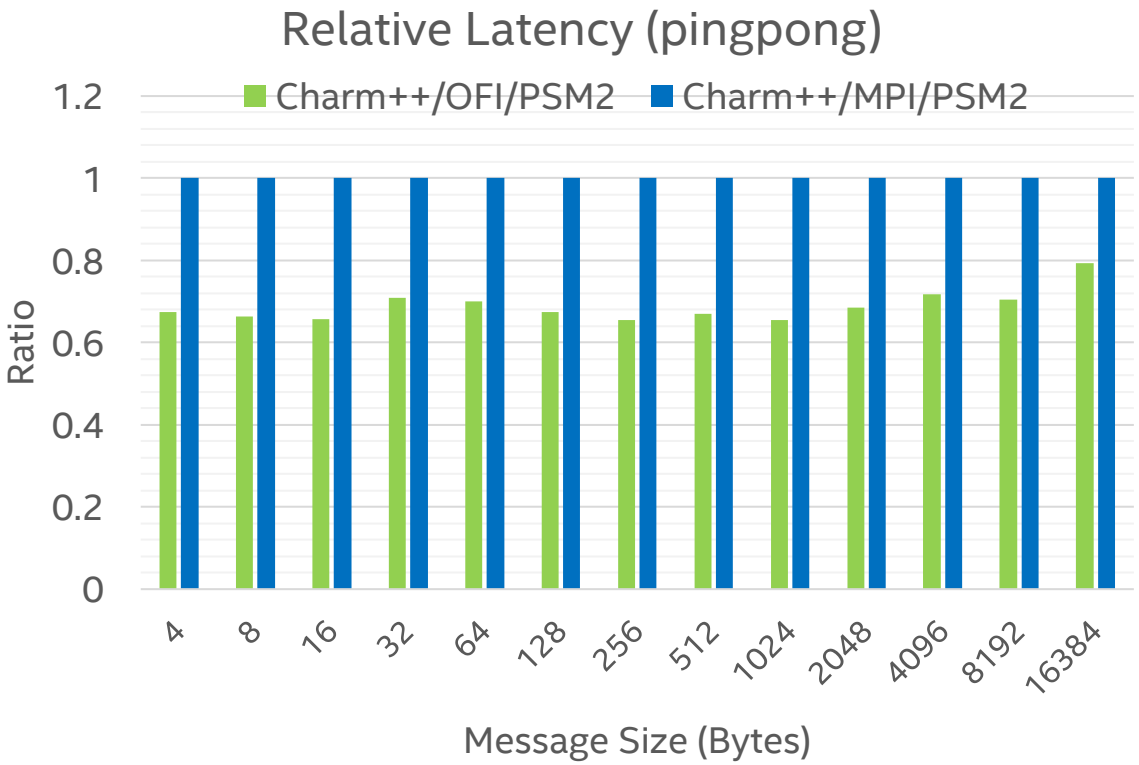
Native OPA provider could map better, as opposed to through PSM2

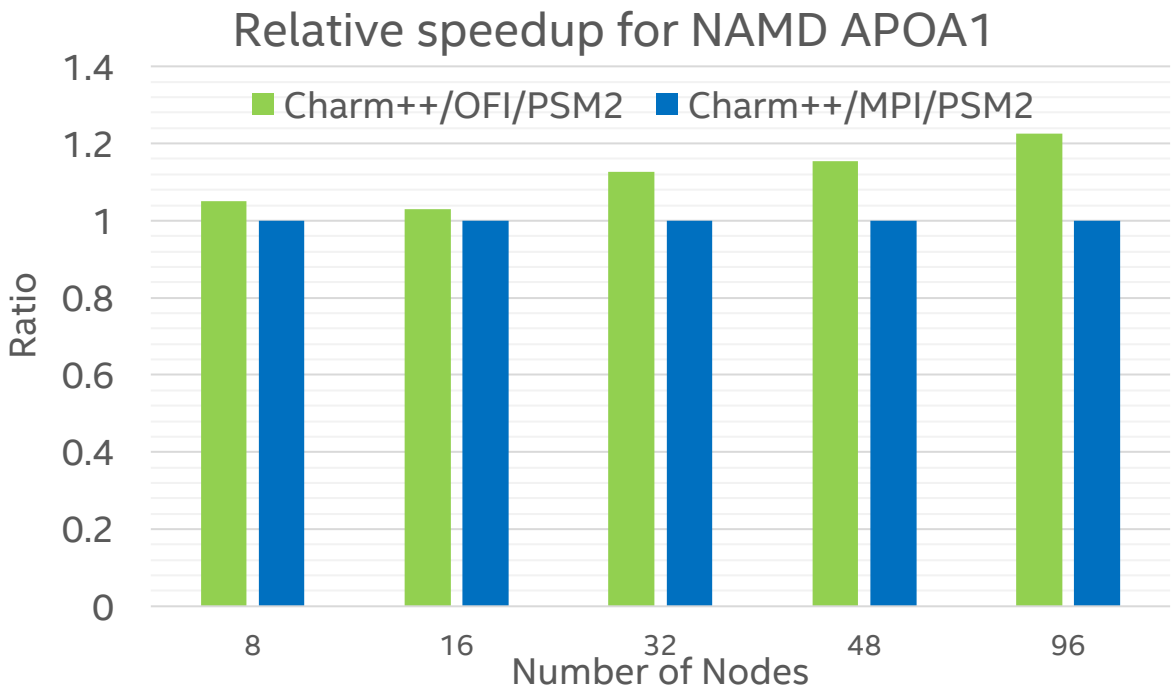
Charm++ on Intel® OPA



- **New OFI LRTS implementation**
  - Removes MPI overhead
  - Uses FI_TAGGED and FI_RMA APIs with FI_EP_RDM endpoint
  - Two data transfer modes
    - Small/medium messages: uses pre-posted buffers on receiver's side to avoid usage of unexpected path
    - Large messages: use RMA Read
  - Future work: multi-threaded use cases can benefit from scalable endpoints

Charm++ Performance Comparisons





Around 30% reduction in latency for small messages

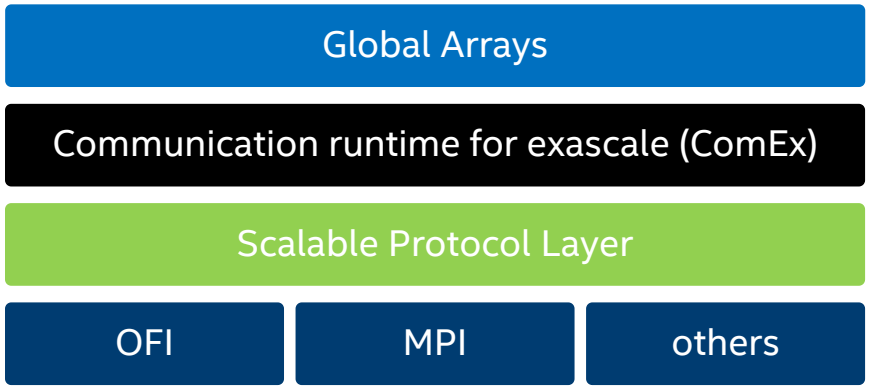
Intel(R) Xeon(R) CPU E5-2695 v3 @ 2.30GHz; data from PSC Bridges, reported by Nitin Bhat of UIUC

Performance increased by 22% at 96 nodes

Intel(R) Xeon Phi(R) 7250F (Knights Landing); IFS 10.2; libfabric 1.4.2; Intel MPI 2017.3.196

Global Arrays

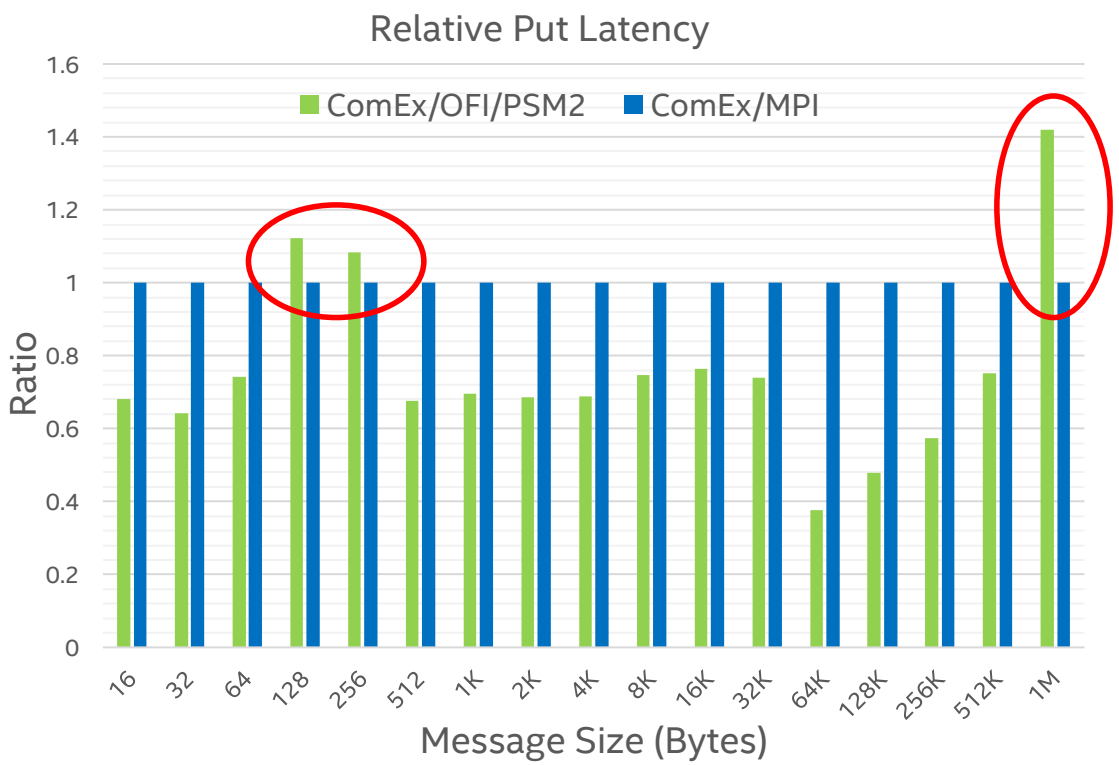
- **New OFI ComEx implementation**
  - Removes MPI overhead
  - Uses FI_TAGGED and FI_RMA with FI_EP_RDM endpoint, FI_MR_SCALABLE
  - Put/Get uses the RMA APIs
  - Lock/Unlock uses the tagged and atomics
  - Accumulate/RMW use the atomics
  - If provider doesn't support atomics, then implement atomics over messages
    - OPA provider emulates atomics internally

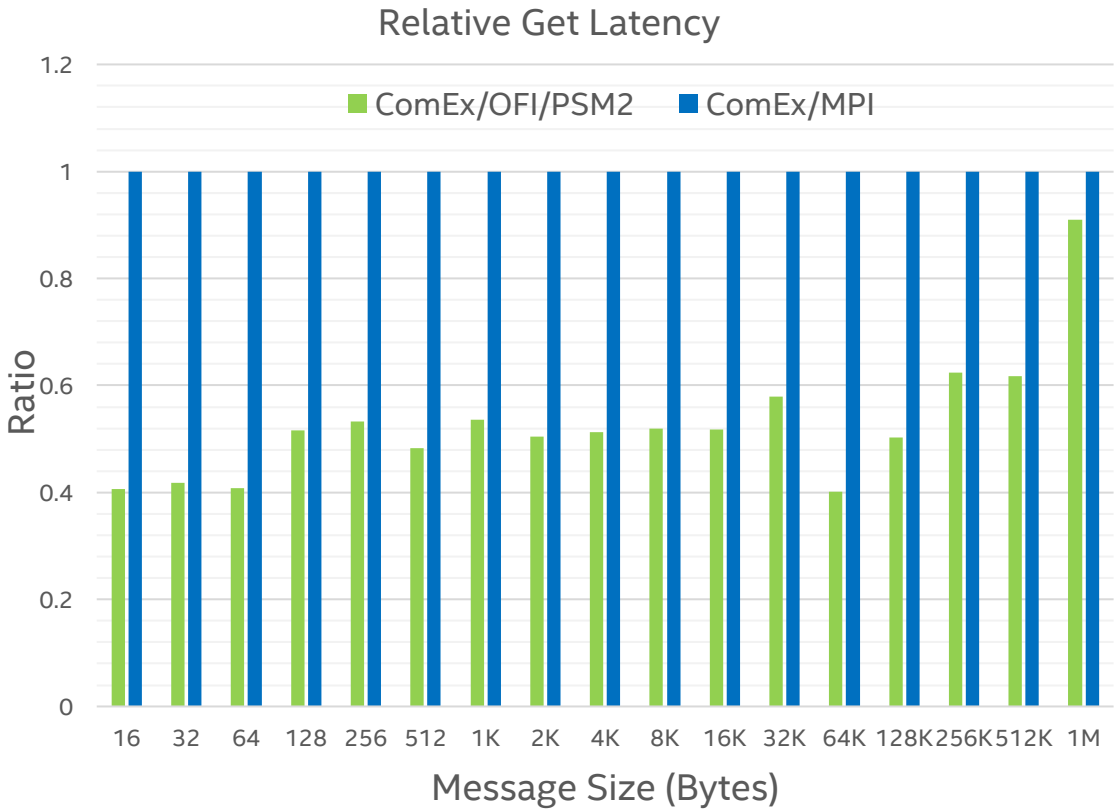


Currently GA is single-threaded. There is effort from Jeff Hammond to add multi-threading support. So there will be the same plans/issues as for Charm++/OFI:

- utilize scalable endpoints API
- improve intra-node communications

ComEx performance over OFI – Put and Get

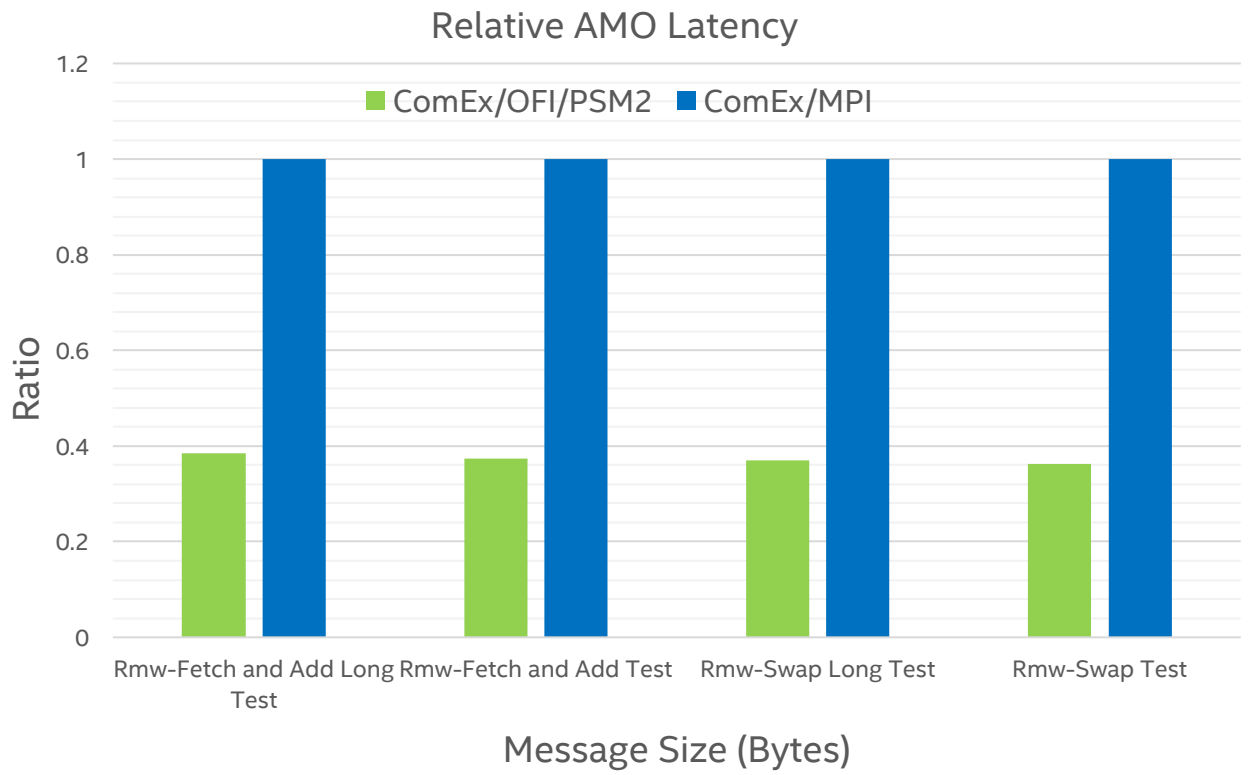


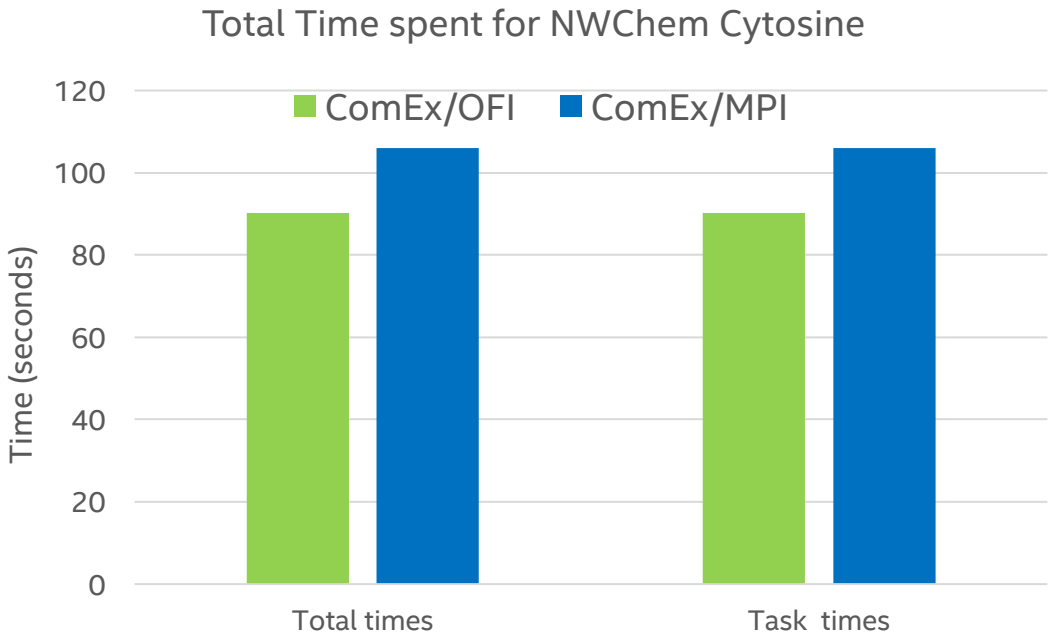


Significant improvements in both Get and Put latency
Some tuning needed for a few data points *(work in early stages)*

Intel(R) Xeon Phi(R) 7250F@ 1.4GHz; Global Arrays 5.6.2; libfabric 1.5.0; Intel MPI 2017.3.196

ComEx performance over OFI – AMO and NWCHEM





8 nodes, 68 ranks per node
Total 4352 ranks

More than 50% improvement in AMO ops
More than 10% improvement at NWChem level with Cytosine

Intel(R) Xeon Phi(R) 7250F@ 1.4GHz; Global Arrays 5.6.2; libfabric 1.5.0; Intel MPI 2017.3.196

Summary

Open Fabrics Interface (OFI) provides a rich set of APIs that are suited for applications and drawn from semantic requirements, not underlying hardware

- Allows portability to different fabrics without needing extensive rework
- *Allows developers to focus on architecture of middleware, which can lead to benefits (like with MPICH/CH4, SHMEM)*

Several prominent HPC middleware like MPI, PGAS, Charm++, GA are already working well on OFI, with application level gains

New semantics like NVM and persistence support are being added to OFI

Never been a better time to get involved: http://libfabric.org/

Intel Confidential

Questions?