INTRODUCING SANDIA* OPENSHMEM WITH MULTI-FABRIC SUPPORT USING LIBFABRIC:

Kayla Seager, Software Engineer, Intel
Ryan Grant, Senior Member of Technical Staff, SNL
Special thanks to Sayantan Sur and Jim Dinan, Intel

[ April 5th, 2016 ]
OUTLINE

- **Background**
  - Libfabric and SHMEM co-design
  - Sandia SHMEM support and software structure

- **Scalable Design**
  - Elements leveraged from Libfabric

- **Portability**
  - SOS support for many fabrics

- **Results**
  - Initial performance results
BACKGROUND
OFA’s OFIWG Goals for Libfabric:

- “tight semantic map between applications and underlying fabric services” – OFIWG
  [https://ofiwg.github.io/libfabric/](https://ofiwg.github.io/libfabric/)

**Network Interface**

**Applications**

**Influence**

**Leads to**

**Libfabric**

**App Centric Design = Performance, Scalability, Portability**
Application mapping example embodies Libfabric’s goal:

SHMEM Application

Influenced

Libfabric Framework

Lead to

Thin SHMEM-OFI layer

Libfabric

SHMEM-Libfabric = Performance, Scalability, & Portability
WHAT IS OPENSHMEM?

- HPC Communication Programming Model API
  - RMA & Atomic Pt-Pt
  - Distributed shared memory model (symmetric addressing)
  - Collectives
    - barrier, broadcast, reduce, all-to-all, strided all-to-all

Node A

Node B

SHMEM RMA WRITE (PUT)
Sandia OpenSHMEM

[GitHub repository URL]

**NEW**

- Optional: PMI
- RMA Collectives
- Locking implementation
- Heap Management
- Transport Layer
- malloc
- SHM

- CMA
- XPMEM
- Portals4
- Libfabric
OpenSHMEM 1.2.0 compliant release

OpenSHMEM 1.3.0 compliant release (1st)
- includes true non-blocking API and all-to-all collectives

Open Source BSD license

Vehicle to…
- establish interface between Libfabric and OpenSHMEM
- Drive both specifications
Software Quality

- Continuous integration testing
- Rich test suite x Multiple interfaces
- Supports independent run environments
- Easy issue tracking -> bug resolution
PR TESTING FRAMEWORK: TRAVIS CI

Testing for every incoming PR

Tests multiple environment combinations
SCALABILITY AND PERFORMANCE
OFIWG’S SHMEM REQUIREMENTS BAKED INTO OFI
USED BY SHMEM-OFI

Scalable Endpoint Enumeration

Efficient Remote Completion for Put/Get

Reliable connectionless endpoints with logical addressing

enabled counters for lightweight remote completion
OIFIWG’S SHMEM REQUIREMENTS BAKED INTO OFI
USED BY SHMEM-OFI

Accelerated small message pathway

Reuse Local Buffer

Immediate local completion: “fi_inject”

Minimize Instructions

Fabric Direct

PGAS Implementation

Hardware Fabric
OFIWG’S SHMEM REQUIREMENTS BAKED INTO OFI
USED BY SHMEM-OFI

Scalable, memory usage model

VA communication style, expose full VA range

Rich Atomic Set

Node A

Node B

Full SHMEM converge with fine API granularity
PORTABILITY
PORTABILITY SCOPE

Sandia OpenSHMEM

Portals 4

libfabric

Sockets Provider

PSM/PSM2 Provider

uGNI Provider

BG/Q Provider

Supports maximum portability

Intel® Omni-Path Architecture generations

Under Active Development

Software (IB)

Hardware (BXI)
**MEMORY MODEL BEFORE**

<table>
<thead>
<tr>
<th>Libfabric Semantic</th>
<th>Remote Virtual Addr. (OS support)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MR_SCALABLE</strong></td>
<td><strong>Symmetric Addressing</strong></td>
</tr>
<tr>
<td><em>user defined keys</em></td>
<td>(Address on Node A = Address on Node B)</td>
</tr>
<tr>
<td><em>Full VA Space</em></td>
<td>+full VA space addressing</td>
</tr>
<tr>
<td>+no key exchange</td>
<td>+minimal footprint (don’t track addresses)</td>
</tr>
<tr>
<td>+single registration for full VA space</td>
<td></td>
</tr>
<tr>
<td>+minimal footprint (don’t track keys)</td>
<td></td>
</tr>
</tbody>
</table>

**MR_SCALABLE and symmetric VA = good SHMEM semantic match**
## MEMORY MODEL **AFTER**

<table>
<thead>
<tr>
<th>OFI Semantic</th>
<th>Remote Offset</th>
<th>Remote Virtual Addr. (OS support)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MR_SCALABLE *user defined keys</td>
<td>Offset addressing, -track and exchange offset</td>
<td>Symmetric addressing</td>
</tr>
<tr>
<td>MR_BASIC *fabric defined keys</td>
<td>Offset Addressing -track and exchange offset -track and exchange keys</td>
<td>Symmetric Addressing -track and exchange keys</td>
</tr>
</tbody>
</table>

**Adding MR Basic & offset enables broader provider support**
- **Require full atomic fabric support**
  - No current hardware supports long double atomics

```
fi_getinfo (fabric discovery)
```

- Fabric 1
  - No long double atomics

- Fabric 2
  - Full Software Atomics
Enabled “hybrid” atomic support for portability
- detect and allow long double (only) limitation

- **fi** _atomic valid (fine grain support detection)

- **fi** _getinfo (fabric discovery)

- **Enable**

- **NEW**

- **Software reduction collective**

- **Full Software Atomics**

- **Fabric 1**
  - No long double atomics

- **Fabric 1**
  - No long double atomics

- **Fabric 2**
  - Full Software Atomics

- **SHMEM Request**
  - Full Atomics
EARLY RESULTS

- **Scalability:**
  - Sandia-SHMEM-OFI-PSM2: scaling to 2,048 PEs (16ppn)
  - Using ISx (integer sort) with 134,217,728 keys per PE
  - Sandia-SHMEM-OFI-Sockets: scaling 512 PEs (14ppn)
    - Full OSU* SHMEM test-suite

- **Performance:**
  - Shmem_int_p -> fi_inject_write: **16 instructions**
  - Shmem_fence -> fi_cntr_wait: **11 instructions**
Performance and Scalability

- Enabled by SHMEM requirements baked into libfabric framework
- `shmем_int_p`: 16 instructions
- scaled to 2,048 processes

Portability

- 4 different fabrics/providers
- SHMEM-libfabric changes for portability
  - Flexible memory model (compile time choice)
  - Software long double atomic support
12th ANNUAL WORKSHOP 2016

THANK YOU

Kayla Seager, Software Engineer, Intel
Ryan Grant, Senior Member of Technical Staff, SNL
Special thanks to Sayantan Sur and Jim Dinan

Part of this work was funded through the Computational Systems and Software Environment sub-program of the Advanced Simulation and Computing Program funded by the National Nuclear Security Administration