GASNet-EX: A High-Performance, Portable Communication Library for Exascale

Dan Bonachea and Paul H. Hargrove

gasnet-staff@lbl.gov  https://gasnet.lbl.gov

Workshop on Languages and Compilers for Parallel Computing (LCPC’18)
October 11, 2018
Abstract

• Present GASNet-EX, the successor to GASNet-1
• Show performance improvements due to the redesign
• Show RMA performance remains competitive w/ MPI-3
  - Better in many cases, across multiple HPC systems
Outline

1. Introduction to GASNet-1 and GASNet-EX
2. Overview of GASNet-EX Improvements
3. Specific GASNet-EX Improvements
4. RMA Microbenchmarks
5. Conclusions
GASNet-1: Overview

• Started in 2002 to provide a portable network communication runtime for three PGAS languages:
  - UPC, CAF and Titanium

• Primary features:
  - Non-blocking RMA (one-sided Put and Get)
  - Active Messages (simplification of Berkeley AM-2)

• Motivated by semantic issues in (then current) MPI-2.0
  - Dan Bonachea, Jason Duell, "Problems with using MPI 1.1 and 2.0 as compilation targets for parallel language implementations", IJHPCN 2004.
GASNet: Adoption and Portability

- **Client runtimes**
  - LBNL UPC++
  - Berkley UPC
  - GCC/UPC
  - Clang UPC
  - Cray Chapel

- **Network conduits**
  - OpenFabrics Verbs (InfiniBand)
  - Mellanox MXM and VAPI (InfiniBand)
  - Cray uGNI (Gemini and Aries)
  - Intel PSM2 (OmniPath)
  - UDP (any TCP/IP network)
  - OFI/libfabric

- **Supported platforms**
  - Over 10 compiler families, 15 operating systems and dozens of architectures

* These lists and counts include both current and past support
GASNet-EX: Overview

• GASNet-EX is the next generation of GASNet
  - Addressing needs of newer programming models such as LBNL UPC++, Stanford Legion and Cray Chapel
  - Incorporating over 15 years of lessons learned
  - Provides backward compatibility for GASNet-1 clients

• Motivating goals include
  - Support more client asynchrony
  - Enable more client adaptation
  - Improve memory footprint
  - Improve threading support
  - Support offload to network h/w
  - Support multi-client applications
  - Support for device memory
GASNet-EX: Status

• GASNet-EX is a work-in-progress
  - Not every new feature has been implemented yet
  - Many have, with benefits this presentation will show

• Three key clients using GASNet-EX
  - UPC++ v1.0 requires GASNet-EX
  - Legion and Chapel are starting work to use EX features

• Will displace legacy GASNet-1 implementation in 2019
Outline

1. Introduction to GASNet-1 and GASNet-EX
2. Overview of GASNet-EX Improvements
3. Specific GASNet-EX Improvements
4. RMA Microbenchmarks
5. Conclusions
Overview of Improvements

GASNet-1:
```c
gasnet_handle_t
gasnet_put_nb(gasnet_node_t node, void *dest_addr,
    void *src_addr, size_t nbytes);
```

GASNet-EX:
```c
gex_Event_t
gex_RMA_PutNB(gex_TM_t tm, gex_Rank_t rank, gex_Addr_t dest_addr,
    void *src_addr, size_t nbytes,
    gex_Event_t *lc_opt, gex_Flags_t flags);
```

A representative example: non-blocking RMA Put

Changes between these two (in red on following slides) illustrate some of the most meaningful changes made in the GASNet-EX design.

They provide the means to address several goals.
Overview of Improvements

\begin{verbatim}
Overview of Improvements

\texttt{gasnet\_handle\_t}
\texttt{gasnet\_put\_nb(gasnet\_node\_t node, void *dest\_addr,}
\texttt{ void *src\_addr, size\_t nbytes);};

\texttt{gex\_Event\_t}
\texttt{gex\_RMA\_PutNB(gex\_TM\_t tm, gex\_Rank\_t rank, gex\_Addr\_t dest\_addr,}
\texttt{ void *src\_addr, size\_t nbytes,}
\texttt{ gex\_Event\_t *lc\_opt, gex\_Flags\_t flags);}
\end{verbatim}

Return Type

GASNet-1: “handle” to test for operation completion
- Thread-specific (only the issuing thread can test/wait for completion)

GASNet-EX: “Event” generalizes handle in two directions
- Not thread-specific (for progress threads, continuation passing, etc.)
- Supports multiple sub-events (e.g. local completion on later slide)
Overview of Improvements

GASNet-1:

```c
void *gasnet_put_nb(gasnet_node_t node, void *dest_addr, void *src_addr, size_t nbytes);
```

GASNet-EX:

```c
void *gex_RMA_PutNB(gex_TM_t tm, gex_Rank_t rank, gex_Addr_t dest_addr, void *src_addr, size_t nbytes,
                     gex_Event_t *lc_opt, gex_Flags_t flags);
```

**Destination**

GASNet-1: an integer **node** identifier to name a process

GASNet-EX: a (**team, rank**) pair to name an “Endpoint”

- “Team” is an ordered sets of Endpoints (also used in collectives)
- Multiple Endpoints for multi-threading and access to device memory
- Multiple Client runtimes for hybrid applications
Overview of Improvements

gasnet_handle_t
gasnet_put_nb(gasnet_node_t node, void *dest_addr,
void *src_addr, size_t nbytes);

gex_Event_t
gex_RMA_PutNB(gex_TM_t tm, gex_Rank_t rank, gex_Addr_t dest_addr,
void *src_addr, size_t nbytes,
gex_Event_t *lc_opt, gex_Flags_t flags);

Destination Address
GASNet-1: a remote virtual address
GASNet-EX: a remote virtual address or an offset
- Offsets can improve scalability of clients using symmetric heaps
- Used with multiple endpoints will enable addressing device memory
Overview of Improvements

gasnet_handle_t
gasnet_put_nb(gasnet_node_t node, void *dest_addr,
                void *src_addr, size_t nbytes);
gex_Event_t
gex_RMA_PutNB(gex_TM_t tm, gex_Rank_t rank, gex_Addr_t dest_addr,
               void *src_addr, size_t nbytes,
               gex_Event_t *lc_opt, gex_Flags_t flags);

Local Completion (when local source buffer may be overwritten)

GASNet-1: ...put_nb() vs. ...put_nb_bulk()
• Local completion can occur separately from remote completion
• Option to conflate it with either injection or remote completion

GASNet-EX: lc_opt selects a local completion behavior
• Both GASNet-1 options, plus an Event the client can test/wait
Overview of Improvements

```
gasnet_handle_t
GASNet-1: gasnet_put_nb(gasnet_node_t node, void *dest_addr,
void *src_addr, size_t nbytes);

gex_Event_t
gex_RMA_PutNB(gex_TM_t tm, gex_Rank_t rank, gex_Addr_t dest_addr,
void *src_addr, size_t nbytes,
gex_Event_t *lc_opt, gex_Flags_t flags);
```

Per-operation Flags

GASNet-EX: introduces extensibility modifiers

- **Require** non-default behaviors, such as offset-based addressing
- **Allow** optional behaviors, such as “Immediate Mode” (later slide)
- **Assert** properties which may eliminate more costly dynamic checks

GASNet-1: has no direct equivalent
Outline

1. Introduction to GASNet-1 and GASNet-EX
2. Overview of GASNet-EX Improvements
3. Specific GASNet-EX Improvements
4. RMA Microbenchmarks
5. Conclusions
Specific GASNet-EX Improvements

• Several new features are already delivering benefits
• This section reports on four of these
  - Local Completion Control
  - Immediate-mode Communication Injection
  - Negotiated-payload Active Messages
  - Remote Atomics
• This section’s results collected on Cray XC40 systems
• The paper provides more detail than can be given here
Local Completion Control

- Figure shows a proxy for how exposing a local completion event can improve overlap:
  - The analogous change within GASNet-EX’s aries-conduit has improved flood bandwidth
- Blue series shows bandwidth prior to utilizing GNI-level local completion
- Red series shows up to 32% increased bandwidth with the local completion event

Non-bulk Put flood bandwidth on Cray Aries with and without use of a local completion event at the GNI level
Immediate-mode Communication Injection

• Lack of resources can stall communication injection
  - Such backpressure may be path-specific
• New feature allows client adaptation to such a scenario
  - E.g. work-stealing could select a different victim
• Immediate-mode is a flag which permits (does not require) implementation to return *without* performing communication, in the presence of backpressure
Immediate-mode Communication Injection

- Figure illustrates performance on a benchmark modeling AM communication with inattentive peers
- Shows reduction in time to complete communication using a “reactive” immediate-mode approach
- The series compare reactive to three distinct static schedules
- Best case is 93% reduction
Negotiated-Payload Active Messages

• “Negotiated-Payload” is a new family of AM interfaces
  - Splits AM injection into distinct Prepare and Commit phases
  - Client and GASNet can negotiate the buffer size and ownership
• Case 1: “chunking” loops may better utilize available buffer resources, allowing fewer larger messages
• Case 2: remove critical-path `memcpy` for some patterns

```c
// Fixed-Payload code, for which most conduits require a memcpy to an internal buffer:
assemble_payload(client_buf, len); // writes client-owned memory
gex_AM_RequestMedium1(team, rank, handler, client_buf, len, GEX_EVENT_NOW, flags, arg);

// Negotiated-Payload avoids the memcpy via payload assembly into a GASNet-owned buffer:
gex_AM_SrcDesc_t sd = gex_AM_PrepareRequestMedium(team, rank, NULL, len, len, NULL, flags, 1);
assemble_payload(gex_AM_SrcDescAddr(sd), len); // writes GASNet-owned memory
gex_AM_CommitRequestMedium1(sd, handler, len, arg);
```
Negotiated-Payload Active Messages

- Figure shows an AM ping-pong bandwidth benchmark using the memcpy-removal pattern on the previous slide
- Normalized to the Fixed-Payload performance
- Shows NP-AM implementation for Cray Aries network delivering up to a 14% improvement

Aries-conduit NP-AM speedup on a ping-pong test with dynamically-generated payload

Bandwidth, normalized to Fixed-Payload

Message Size
Remote Atomics

• “Remote Atomics” is a new family of RMA interfaces
  - Analogous to MPI accumulate operations
• Interface designed with offload in mind
• Uses the “atomics domain” concept
  - Introduced by UPC 1.3
  - Enables efficient offload, even in the presence of concurrent updates to the same location using multiple distinct operations
Remote Atomics

- Offload reduces latency of fetch-and-add by 70% relative to generic AM-based reference
- Figure shows aggregate throughput of a “hot-spot” test of fetch-and-add (all to one)
- Green series shows robust scaling to saturation when offloaded to the Aries NIC
Outline

1. Introduction to GASNet-1 and GASNet-EX
2. Overview of GASNet-EX Improvements
3. Specific GASNet-EX Improvements
4. RMA Microbenchmarks
5. Conclusions
RMA Bandwidth Microbenchmarks

- This section reports unidirectional flood bandwidth measured between two nodes, one process per node.
- Intel MPI Benchmarks v2018.1 to measure MPI-3 RMA
  - IMB-RMA test, Unidir_put and Unidir_get subtests
  - “Aggregate” result category reports bandwidth of
    - Series of many MPI_Put (or Get) operations
    - A single final call to MPI_Win_flush
  - All within a passive-target access epoch established by a call to MPI_Win_lock(SHARED) outside the timed region
- GASNet-EX measures nearest semantic equivalent
RMA Bandwidth Microbenchmarks

• Results collected on four platforms
  - Three different MPI implementation (Cray, IBM and MVAPICH2)
  - Two distinct networks (Cray Aries and Mellanox EDR InfiniBand)
  - Three CPU families (Xeon Haswell, Xeon Phi, and POWER8)
  - Complete details are given in the paper

• Results are collected in “out of the box” configurations
  - Used center’s defaults on the three production systems
  - No tuning knobs used to improve performance
RMA Bandwidth Microbenchmarks

RMA Put on Cori-I (Haswell, Aries, Cray MPI)

Bandwidth (GiB/s)

Transfer Size

GASNet-EX Put
MPI RMA Put

GOOD
RMA Bandwidth Microbenchmarks

RMA Get on Cori-I (Haswell, Aries, Cray MPI)

Bandwidth (GiB/s)

Transfer Size

- GASNet-EX Get
- MPI RMA Get
RMA Bandwidth Microbenchmarks

GASNet-EX / LCPC'18 / Paul H. Hargrove 29
RMA Latency Microbenchmarks

- RMA full-operation latency
  - Same RMA Put or Get operation as flood test
  - But keep just one in-flight instead of many (Win_flush after each)
- Figure reports representative 8-byte latencies
  - GASNet-EX uniformly competitive with MPI-3, better for small sizes

Latency of RMA Put and Get

<table>
<thead>
<tr>
<th>System</th>
<th>GASNet-EX Put</th>
<th>MPI RMA Put</th>
<th>GASNet-EX Get</th>
<th>MPI RMA Get</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cori-I</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Haswell</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aries</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cray MPI</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cori-II</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Haswell</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aries</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cray MPI</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Summitdev</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POWER8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>InfiniBand</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IBM Spectrum</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPI</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MVAPICH2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

GOOD
Outline

1. Introduction to GASNet-1 and GASNet-EX
2. Overview of GASNet-EX Improvements
3. Specific GASNet-EX Improvements
4. RMA Microbenchmarks
5. Conclusions
Conclusions

• GASNet-EX is the next generation of GASNet, addressing needs of newer programming models
  - Asynchrony, adaptively, threading, scalability, device memory, ...
• Already in production use by UPC++
  - Looking for new clients, talk to me over coffee!
• Provides backward compatibility for GASNet-1 clients
• Benefits of new features are already measurable
• Delivers RMA performance competitive with MPI-3 RMA
THANK YOU

gasnet-staff@lbl.gov
https://gasnet.lbl.gov

GASNet-EX and UPC++ have a research poster at SC18
Acknowledgements

• This research was funded in part by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration.

• This research used resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

• This research used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357.

• This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.
Local Completion Control

• GASNet-EX introduces means for client to test (or wait) for local completion *between* injection and completion of a non-blocking Put

• Exposes greater opportunity for communication overlap than possible with the options available in GASNet-1
Non-Contiguous RMA

• GASNet-EX adds Vector-Indexed-Strided (VIS) APIs
  - Express non-blocking Put and Get of non-contiguous data
  - Names reflects the three metadata formats
    • Different trade-offs between size and generality
  - Small modifications to an unofficial GASNet-1 extension

• Implementation uses Active Messages, when appropriate, for pack/unpack of data
  - Benefits from reimplemented using Negotiated-Payload AM
Non-Contiguous RMA

- Figure illustrates performance of Strided Put API
- Red series shows performance using Fixed-Payload AM
- Blue series shows performance using Negotiated-Payload AM
- Both normalized to GASNet-1

Improved Strided Put performance, relative to GASNet-1
Cori-I: Haswell, Aries, Cray MPI

![Graph showing bandwidth (GiB/s) vs transfer size (256 B to 4 MiB) for different operations and MPI libraries.](image)

- **Bandwidth (GiB/s)**
- **Transfer Size**
- **GASNet-EX Put**
- **MPI RMA Put**
- **GASNet-EX Get**
- **MPI RMA Get**
- **MPI ISend/IRecv**
Summitdev (single-rail): POWER8, InfiniBand, IBM Spectrum MPI

Bandwidth (GiB/s) vs. Transfer Size

- GASNet-EX Put
- MPI RMA Put
- GASNet-EX Get
- MPI RMA Get
- MPI ISend/IRrecv

Transfer Sizes:
- 256 B
- 1 kiB
- 4 kiB
- 16 kiB
- 64 kiB
- 256 kiB
- 1MiB
- 4MiB
Gomez: Haswell-EX, InfiniBand, MVAPICH2

Bandwidth (GiB/s) vs. Transfer Size

- GASNet-EX Put
- MPI RMA Put
- GASNet-EX Get
- MPI RMA Get
- MPI ISend/IREcv
## RMA Latency Microbenchmarks

<table>
<thead>
<tr>
<th>System</th>
<th>8-Byte RMA Put Latency</th>
<th>8-Byte RMA Get Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GASNet-EX</td>
<td>MPI3-RMA</td>
</tr>
<tr>
<td>Cori-I</td>
<td>1.07 µs</td>
<td>1.20 µs</td>
</tr>
<tr>
<td>Cori-II</td>
<td>2.15 µs</td>
<td>3.42 µs</td>
</tr>
<tr>
<td>Summitdev</td>
<td>1.61 µs</td>
<td>8.10 µs</td>
</tr>
<tr>
<td>Gomez</td>
<td>1.41 µs</td>
<td>1.51 µs</td>
</tr>
</tbody>
</table>
RMA Latency Microbenchmarks

Cori-I: Haswell, Aries, Cray MPI

Cori-II: Xeon Phi, Aries, Cray MPI

Gomez: Haswell-EX, InfiniBand, MVAPICH2

Summitdev (single-rail): POWER8, InfiniBand, IBM Spectrum MPI