Evaluation of Remote Memory Access Communication on the Cray XT3

V. Tipparaju\textsuperscript{1}, A. Kot\textsuperscript{1}, J. Nieplocha\textsuperscript{1}, M. ten Bruggencate\textsuperscript{2} and N. Chrisochoides\textsuperscript{3}

\textsuperscript{1}. Pacific Northwest National Laboratory
\textsuperscript{2}. Cray, Inc.
\textsuperscript{3}. The College of William and Mary
Remote Memory Access Communication

- Offers support for an intermediate programming model between shared memory and message passing
- Combines advantages of both worlds
  - Direct access to shared/global data (shared memory)
  - Control over locality and data distribution (message passing)
- Increased asynchrony due to absence of receive call
- Many operations supported natively by hardware
- Widely supported by vendors
Cray XT3 Network hardware/software

- Seastar Network
  - System-on-chip design with HyperTransport
  - 3-dimensional torus interconnect
  - 2 DMA engines

- Portals
  - Connectionless for scalability
  - Independent of network hardware for portability
  - User-level flow control and OS bypass for low latency
  - Ability to handle unexpected messages for MPI
3 RMA models

- Compare different models implemented on Portals
  - SHMEM
  - MPI-2
  - ARMCI

- Discuss functionality and implementation

- Compare features

- Evaluate performance
  - Put/Get bandwidth
  - Put/Get/Sendrecv bandwidth of MPI-2
  - Potential overlap for MPI-2 and ARMCI
  - NAS MG benchmark on AMRCI and MPI-1
Portal Addressing Structures

**Data Movement**
- Set of matching bits instead of virtual address allows to support traditional RMA and two-sided send/receive
- Read(get), write(put) and atomic swap(getput) operations

**Receiving Messages**
- Put/get requests (subject to translation), acks and replies
**SHMEM**

- **Functionality**
  - Data transfer (e.g., `shmem_put`)
  - Atomic (e.g., `shmem_swap`)
  - Collective data transfer (e.g., `shmem_sum_to_all`)
  - Initialization and information (e.g., `shmem_init`, `shmem_my_pe`)
  - Synchronization (`shmem_barrier`, `shmem_quiet`)
  - Limited support for non-contiguous data transfer, non-blocking calls, wait for a particular call (present in API but not implemented)

- Different vendors support SHMEM interface and add extensions, e.g., Elan4 provides non-blocking calls
SHMEM implementation

- Allocates all resources on startup
- 2 out of 4 memory segments are symmetric and remotely accessible: symmetric heap and data segment
- Reserves 2 Portal Indices (PI) for each segment
- Allocates 2 EQs, separately for Get and Put
- Addresses and offsets used to determine MD, EQ and PI
- Target PE translated into target Portal Process ID
- After return from Portal call monitors EQ before return
- Strided* calls implemented as sequences of contiguous
- All Put calls request acknowledgements to support global synchronization (on event wait for all acks)
MPI-2 One-sided

- MPI_PUT, MPI_GET, MPI_ACCUMULATE
- RMA *must* take place within synchronization “epoch”
- Synchronization:
  - active target - on both processes
  - passive target - on originating process
- Marks the start of an “epoch” with MPI_Win_fence, MPI_Win_start/MPI_Win_post or MPI_Win_lock
- Marks the end of the “epoch” with MPI_Win_complete, MPI_Win_fence or MPI_Win_unlock
MPI-2 implementation

- Uses Portal-base variant of MPICH2 CH3 ADI 3
- No data transfers occur during Put/Get/Accumulate
- Almost all data calls occur at the end of the epoch:
  - Each process examine posted requests and builds RMA messages
  - Accesses that use derived data-types are packed/unpacked and transferred using approach similar to MPI send/receive
  - Accumulate operations performed on target process

- Cray does not recommend using MPI-2 on XT3 as a high performance solution
ARMCI – a portable remote memory access library

Functionality
- put, get
- accumulate (atomic reduce)
  - used in many apps, performance optimization possible
    - lock-get-daxpy-put-unlock not very efficient
- atomic read-modify-write, mutexes and locks
- memory allocation operations
- noncontiguous data interfaces (vector, strided)

Characteristics
- simple progress rules
- blocking operations ordered w.r.t. target (ease of use)
- compatible with message-passing libraries (MPI, PVM)
ARMCI implementation

- Prepare memory
  - All remote memory for communication should be allocated by ARMCI: use “wildcard” MD, not associated to any EQ
  - Local memory (ARMCI_Malloc, ARMCI_Malloc_local, user) uses different sets of MDs, associated with EQs
  - MDs retained for remote memory and may be unlinked for local

- Data transfer using Portals
  - Looks for appropriate MD, binds or update if retained; data transfer is then done with PtlGetRegion or PtlPutRegion
  - Non-contiguous transfer implemented as sequences of contiguous
  - Accumulate uses owner computes method due to lack of user-level threads in Catamount
Effective Get bandwidth

![Graph showing effective Get bandwidth for ARMCI, SHMEM, and MPI. The x-axis represents message size in bytes, and the y-axis represents bandwidth in MB/s.]
Effective Put bandwidth

- ARMCI
- SHMEM
- MPI

Bandwidth (MB/s) vs. Message size (Bytes)
Effective MPI bandwidth

- **Send-Recv**: Green line
- **Get**: Purple line
- **Put**: Orange line

The graph shows the effective MPI bandwidth with respect to message size in bytes. The x-axis represents message size (Bytes), while the y-axis shows bandwidth in MB/s.
Maximum potential overlap

- Overlapping communications with computations benefits from increased asynchrony of RMA model

- Maximum overlap benchmark
  - Increased computation is gradually inserted between the initiating non-blocking get call and the wait completion call
  - At some point computation would exceed idle CPU time
  - This point gives maximum potential overlap

- Cray SHMEM does not support non-blocking get
Maximum potential overlap (cont)

![Graph showing the overlap percentage against bytes for ARMCI_NbGet and MPI_Get.]
MG NAS benchmark: ARMCI vs. MPI-1
Conclusions and Future work

- Evaluated SHMEM, MPI-2 and ARMCI on Cray XT3
- ARMCI offers superset of RMA operations in all three
- Showed the effectiveness of RMA
  - SHMEM and ARMCI showed good performance
  - MPI-2 did not
  - Additionally, ARMCI showed very good overlap

- We are investigating
  - extension to Portals to make it more general and better support RMA models
  - using accelerated Portal developed by Sandia (Ron Brightwell and Kevin Pedretti)