The Message Passing Interface: MPI 3.1 Released, Next Stop MPI 4.0

Martin Schulz
LLNL / CASC
Chair of the MPI Forum
MPI Forum BOF @ SC15, Austin, TX

http://www.mpi-forum.org/

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

Agenda

- **Current State of MPI**
  - Features and Implementation status of MPI 3.1
- **Working group updates**
  - Fault Tolerance (Wesley Bland)
  - Hybrid Programming (Pavan Balaji)
  - Persistence (Anthony Skjellum)
  - Point to Point Communication (Daniel Holmes)
  - One Sided Communication (Rajeev Thakur)
  - Tools (Kathryn Mohror)
- **How to contribute to the MPI Forum?**

Let's keep this interactive – Please feel free to ask questions!

Where We Are

- **MPI 3.0 ratified in September 2012**
  - Available at http://www.mpi-forum.org/
  - Several major additions compared to MPI 2.2

Notable Additions to MPI 3.0

- Non-blocking collectives
- Neighborhood collectives
- RMA enhancements
- Shared memory support
- MPI Tool Information Interface
- Non-collective communicator creation
- Fortran 2008 Bindings
- New Datatypes
- Large data counts
- Matched probe

Where We Are

Available through HLRS
-> MPI Forum Website

- **MPI 3.0 ratified in September 2012**
  - Available at http://www.mpi-forum.org/
  - Several major additions compared to MPI 2.2
- **MPI 3.1 ratified in June 2015**
  - Inclusion for errata (mainly RMA, Fortran, MPI_T)
  - Minor updates and additions (address arithmetic and non-block. I/O)
  - Adaption in most MPIs progressing fast

MPI-3.1 Impl. as of November 2015 (thanks to Pavan Balaji)

	MPICH	MVAPICH	Open MPI	Cray MPI	Tianhe MPI	Intel MPI	IBM BG/Q MPI[1]	IBM PE MPICH[2]	IBM Platform	SGI MPI	Fujitsu MPI	MS MPI	MPC
NBC	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	(*)	Q4'15
Nbrhood collectives	✔	✔	✔	✔	✔	✔	✔	✔		✔			Q4'15
RMA	✔	✔	✔	✔	✔	✔	✔	✔		✔			*
Shared memory	✔	✔	✔	✔	✔	✔	✔	✔		✔		✔	*
Tools Interface	✔	✔	✔	✔	✔	✔	✔	✔		✔		*	Q4'16
Comm-creat group	✔	✔	✔	✔	✔	✔	✔	✔		✔			*
F08 Bindings	✔	✔	✔	✔	✔		✔			✔			Q2'16
New Datatypes	✔	✔	✔	✔	✔	✔	✔	✔		✔		✔	Q4'15
Large Counts	✔	✔	✔	✔	✔	✔	✔	✔		✔		✔	Q2'16
Matched Probe	✔	✔	✔	✔	✔	✔	✔	✔		✔	✔	✔	Q2'16
NBC I/O	✔	Q1'16	✔	Q4'15						Q2'16			

Release dates are estimates and are subject to change at any time.

Empty cells indicate no *publicly announced* plan to implement/support that feature.

Platform-specific restrictions might apply for all supported features

[1] Open Source but unsupported [2] **No** MPI_T variables exposed * Under development (*) Partly done

Where We Are

- **MPI 3.0 ratified in September 2012**
  - Available at http://www.mpi-forum.org/
  - Several major additions compared to MPI 2.2
- **MPI 3.1 ratified in June 2015**
  - Inclusion for errata (mainly RMA, Fortran, MPI_T)
  - Minor updates and additions (address arithmetic and non-block. I/O)
  - Adaption in most MPIs progressing fast
- **Parallel to MPI 3.1, the forum started working towards MPI 4.0**
  - Schedule tbd. (depends on features)
  - Several active working groups

Current Working Groups and Topics

- **Collectives & Topologies**
  - Torsten Hoefler, ETH
  - Andrew Lumsdaine, Indiana
- **Fault Tolerance**
  - Wesley Bland, ANL
  - Aurelien Bouteiller, UTK
  - Rich Graham, Mellanox
- **Fortran**
  - Craig Rasmussen, U. of Oregon
- **Generalized Requests**
  - Fab Tillier, Microsoft
- **Hybrid Models**
  - Pavan Balaji, ANL
- **I/O**
  - Quincey Koziol, HDF Group
  - Mohamad Chaarawi, HDF Group
- **Large count**
  - Jeff Hammond, Intel
- **Persistence**
  - Anthony Skjellum, Auburn Uni.
- **Point to Point Comm.**
  - Dan Holmes, EPCC
  - Rich Graham, Mellanox
- **Remote Memory Access**
  - Bill Gropp, UIUC
  - Rajeev Thakur, ANL
- **Tools**
  - Kathryn Mohror, LLNL
  - Marc-Andre Hermans, RWTH Aachen

Current Working Groups and Topics

- **Collectives & Topologies**
  - Torsten Hoefler, ETH
  - Andrew Lumsdaine, Indiana
- **Fault Tolerance**
  - Wesley Bland, ANL
  - Aurelien Bouteiller, UTK
  - Rich Graham, Mellanox
- **Fortran**
  - Craig Rasmussen, U. of Oregon
- **Generalized Requests**
  - Fab Tillier, Microsoft
- **Hybrid Models**
  - Pavan Balaji, ANL
- **I/O**
  - Quincey Koziol, HDF Group
  - Mohamad Chaarawi, HDF Group
- **Large count**
  - Jeff Hammond, Intel
- **Persistence**
  - Anthony Skjellum, Auburn Uni.
- **Point to Point Comm.**
  - Dan Holmes, EPCC
  - Rich Graham, Mellanox
- **Remote Memory Access**
  - Bill Gropp, UIUC
  - Rajeev Thakur, ANL
- **Tools**
  - Kathryn Mohror, LLNL
  - Marc-Andre Hermans, RWTH Aachen

Current Working Groups and Topics

- **Collectives & Topologies**
  - Torsten Hoefler, ETH
  - Andrew Lumsdaine, Indiana
- **Fault Tolerance**
  - Wesley Bland, ANL
  - Aurelien Bouteiller, UTK
  - Rich Graham, Mellanox
- **Fortran**
  - Craig Rasmussen, U. of Oregon
- **Generalized Requests**
  - Fab Tillier, Microsoft
- **Hybrid Models**
  - Pavan Balaji, ANL
- **I/O**
  - Quincey Koziol, HDF Group
  - Mohamad Chaarawi, HDF Group
- **Large count**
  - Jeff Hammond, Intel
- **Persistence**
  - Anthony Skjellum, Auburn Uni.
- **Point to Point Comm.**
  - Dan Holmes, EPCC
  - Rich Graham, Mellanox
- **Remote Memory Access**
  - Bill Gropp, UIUC
  - Rajeev Thakur, ANL
- **Tools**
  - Kathryn Mohror, LLNL
  - Marc-Andre Hermans, RWTH Aachen

Fault Tolerance in the MPI Forum

MPI Forum BoF
Supercomputing 2015
Wesley Bland

What is the working group doing?

- Decide the best way forward for fault tolerance in MPI.
  - Currently looking at User Level Failure Mitigation (ULFM), but that's only part of the puzzle.
- Look at all parts of MPI and how they describe error detection and handling.
  - Error handlers probably need an overhaul
  - Allow clean error detection even without recovery
- Consider alternative proposals and how they can be integrated or live alongside existing proposals.
  - Reinit, FA-MPI, others
- Start looking at the next thing
  - Data resilience?

User Level Failure Mitigation Main Ideas

- Enable application-level recovery by providing minimal FT API to prevent deadlock and enable recovery
- Don't do recovery for the application, but let the application (or a library) do what is best.
- Currently focused on process failure (not data errors or protection)

ULFM Progress

- BoF going on **right now** in 13A
- Making minor tweaks to main proposal over the last year
  - Ability to disable FT if not desired
  - Non-blocking variants of some calls
- Solidifying RMA support
  - When is the right time to notify the user of a failure?
- Planning reading for March 2016

Is ULFM the only way?

- No!
  - Fenix, presented at SC ‘14 provides more user friendly semantics on top of MPI/ULFM
- Other research discussions include
  - Reinit (LLNL) - Fail fast by causing the entire application to roll back to MPI_INIT with the original number of processes.
  - FA-MPI (Auburn/UAB) - Transactions allow the user to use parallel try/catch-like semantics to write their application.
    - Paper in the SC ‘15 Proceedings (ExaMPI Workshop)
- Some of these ideas fit with ULFM directly and others require some changes
  - We’re working with the Tools WG to revamp PMPI to support multiple tools/libraries/etc. which would enable nice fault tolerance semantics.

How Can I Participate?

Website: http://www.github.com/mpiwg-ft

Email: mpiwg-ft@lists.mpi-forum.org

Conference Calls: Every other Tuesday at 3:00 PM Eastern US

In Person: MPI Forum Face To Face Meetings

Current Working Groups and Topics

- **Collectives & Topologies**
  - Torsten Hoefler, ETH
  - Andrew Lumsdaine, Indiana
- **Fault Tolerance**
  - Wesley Bland, ANL
  - Aurelien Bouteiller, UTK
  - Rich Graham, Mellanox
- **Fortran**
  - Craig Rasmussen, U. of Oregon
- **Generalized Requests**
  - Fab Tillier, Microsoft
- **Hybrid Models**
  - Pavan Balaji, ANL
- **I/O**
  - Quincey Koziol, HDF Group
  - Mohamad Chaarawi, HDF Group
- **Large count**
  - Jeff Hammond, Intel
- **Persistence**
  - Anthony Skjellum, Auburn Uni.
- **Point to Point Comm.**
  - Dan Holmes, EPCC
  - Rich Graham, Mellanox
- **Remote Memory Access**
  - Bill Gropp, UIUC
  - Rajeev Thakur, ANL
- **Tools**
  - Kathryn Mohror, LLNL
  - Marc-Andre Hermans, RWTH Aachen

MPI Forum: Hybrid Programming WG

Pavan Balaji

Hybrid Programming Working Group Chair

balaji@anl.gov

MPI Forum Hybrid WG Goals

- Ensure interoperability of MPI with other programming models
  - MPI+threads (pthreads, OpenMP, user-level threads)
  - MPI+CUDA, MPI+OpenCL
  - MPI+PGAS models

MPI-3.1 Performance/Interoperability Concerns

- Resource sharing between MPI processes
  - System resources do not scale at the same rate as processing cores
    - Memory, network endpoints, TLB entries, …
    - Sharing is necessary
  - MPI+threads gives a method for such sharing of resources
- Performance Concerns
  - MPI-3.1 provides a single view of the MPI stack to all threads
    - Requires all MPI objects (requests, communicators) to be shared between all threads
    - Not scalable to large number of threads
    - Inefficient when sharing of objects is not required by the user
  - MPI-3.1 does not allow a high-level language to interchangeably use OS processes or threads
    - No notion of addressing a single or a collection of threads
    - Needs to be emulated with tags or communicators

Single view of MPI objects

- MPI-3.1 specification requirements
  - It is valid in MPI to have one thread generate a request (e.g., through MPI_IRECV) and another thread wait/test on it
  - One thread might need to make progress on another's requests
  - Requires all objects to be maintained in a shared space
  - When a thread accesses an object, it needs to be protected through locks/atomics
    - Critical sections become expensive with hundreds of threads accessing it
- Application behavior
  - Many (but not all) applications do not require such sharing
  - A thread that generates a request is responsible for completing it
    - MPI guarantees are safe, but unnecessary for such applications

```
P0 (Thread 1)
MPI_Irecv(…, comm1, &req1);
pthread_barrier();

pthread_barrier();
MPI_Wait(&req1, …);
```

```
P0 (Thread 2)
MPI_Irecv(…, comm2, &req2);
pthread_barrier();
MPI_Wait(&req2, …);
pthread_barrier();
```

```
P1
MPI_Ssend(…, comm1);
MPI_Ssend(…, comm2);
```

Interoperability with High-level Languages

- In MPI-3.1, there is no notion of sending a message to a thread
  - Communication is with MPI processes – threads share all resources in the MPI process
  - You can emulate such matching with tags or communicators, but some pieces (like collectives) become harder and/or inefficient
- Some high-level languages do not expose whether their processing entities are processes or threads
  - E.g., PGAS languages
- When these languages are implemented on top of MPI, the language runtime might not be able to use MPI efficiently

MPI Endpoints: Proposal for MPI-4

- Idea is to have multiple addressable communication entities within a single process
  - Instantiated in the form of multiple ranks per MPI process
- Each rank can be associated with one or more threads
- Lesser contention for communication on each "rank"
- In the extreme case, we could have one rank per thread (or some ranks might be used by a single thread)

MPI Endpoints Semantics

```
MPI_Comm_create_endpoints(MPI_Comm parent_comm, int my_num_ep,
        MPI_Info info, MPI_Comm out_comm_handles[])
```

- Creates new MPI ranks from existing ranks in parent communicator
  - Each process in parent comm. requests a number of endpoints
  - Array of output handles, one per local rank (i.e. endpoint) in endpoints communicator
  - Endpoints have MPI process semantics (e.g. progress, matching, collectives, …)
- Threads using endpoints behave like MPI processes
  - Provide per-thread communication state/resources
  - Allows implementation to provide process-like performance for threads

MPI Endpoints

Relax the 1-to-1 mapping of ranks to threads/processes

Hybrid MPI+OpenMP Example
With Endpoints

```
int main(int argc, char **argv) {
  int world_rank, tl;
  int max_threads = omp_get_max_threads();
  MPI_Comm ep_comm[max_threads];

  MPI_Init_thread(&argc, &argv, MPI_THREAD_MULTIPLE, &tl);
  MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);

#pragma omp parallel
  {
    int nt = omp_get_num_threads();
    int tn = omp_get_thread_num();
    int ep_rank;
#pragma omp master
    {
      MPI_Comm_create_endpoints(MPI_COMM_WORLD, nt, MPI_INFO_NULL, ep_comm);
    }
#pragma omp barrier
    MPI_Comm_rank(ep_comm[tn], &ep_rank);
    ... // Do work based on 'ep_rank'
    MPI_Allreduce(..., ep_comm[tn]);

    MPI_Comm_free(&ep_comm[tn]);
  }
  MPI_Finalize();
}
```

Additional Notes

- Useful for more than just avoiding locks
  - Semantics that are "rank-specific" become more flexible
    - E.g., ordering for operations from a process
    - Ordering constraints for MPI RMA accumulate operations
- Supplementary proposal on thread-safety requirements for endpoint communicators
  - Is each rank only accessed by a single thread or multiple threads?
  - Might get integrated into the core proposal
- Implementation challenges being looked into
  - Simply having endpoint communicators might not be useful, if the MPI implementation has to make progress on other communicators too

More Info

- Endpoints:
  - https://svn.mpi-forum.org/trac/mpi-forum-web/ticket/380
- Hybrid Working Group:
  - https://svn.mpi-forum.org/trac/mpi-forum-web/wiki/MPI3Hybrid

Current Working Groups and Topics

- **Collectives & Topologies**
  - Torsten Hoefler, ETH
  - Andrew Lumsdaine, Indiana
- **Fault Tolerance**
  - Wesley Bland, ANL
  - Aurelien Bouteiller, UTK
  - Rich Graham, Mellanox
- **Fortran**
  - Craig Rasmussen, U. of Oregon
- **Generalized Requests**
  - Fab Tillier, Microsoft
- **Hybrid Models**
  - Pavan Balaji, ANL
- **I/O**
  - Quincey Koziol, HDF Group
  - Mohamad Chaarawi, HDF Group
- **Large count**
  - Jeff Hammond, Intel
- **Persistence**
  - Anthony Skjellum, Auburn Uni.
- **Point to Point Comm.**
  - Dan Holmes, EPCC
  - Rich Graham, Mellanox
- **Remote Memory Access**
  - Bill Gropp, UIUC
  - Rajeev Thakur, ANL
- **Tools**
  - Kathryn Mohror, LLNL
  - Marc-Andre Hermans, RWTH Aachen

Persistent Collective Operations in MPI

Daniel Holmes, EPCC
Anthony Skjellum, Auburn University
Matthew Shane Farmer, Auburn University
Purushotham V. Bangalore, UAB

Performance, Portability, Productivity

- MPI has been the most effective parallel programming model for the past 20+ years
- 3 P's (or 4)
  - Performance is emphasized
  - Portability is co-equal
  - Productivity is desirable
  - There is now a 4th P – Predictability!
  - Choose 2 phenomenon applies
- There is a cost of portability (CoP)
- Improvements to the API can lower the CoP without reducing portability

Definitions

- Persistence
  - Setup once – may be expensive
  - Reuse N times
- Planned Transfers
  - Transfers for which resources are locked down in advance (DMA, descriptors, buffers…)
- These rely on parameters being static or relatively static
- Goal: cut out mid-level code, tests, setups, teardowns

Motivations

- Non-blocking
  - Supports asynchronous progress independent of the user thread
- Persistent
  - Supports “amortization” of cost to pick best operation and schedule over many uses
  - Requires operations to freeze arguments
- Performance and Predictability are key drivers
- Persistence allows for static resource allocation
- Ex: FFTW uses “planning” to pick algorithms – slow to choose, fast to execute
- Ex: Paper by Jesper Larsson Träff illustrates value

Standardization History

- Point-to-point
  - Persistent and non-persistent since MPI-1
- Collective operations
  - MPI-1 & 2 – blocking (some split phase)
  - MPI-3 – some non-blocking
  - MPI-Next – more non-blocking,

    Persistent nonblocking (proposed)

Concept

- An all-to-all transfer is done many times in an app
- The specific sends and receives represented never change (size, type, lengths, transfers)
- A non-blocking persistent collective operation can take the time to apply a heuristic and choose a faster way to move that data
- Fixed cost of making those decisions (could be high) are amortized over all the times the function is used
- Static resource allocation can be done
- Choose fast(er) algorithm, take advantage of special cases
- Reduce queueing costs
- Special, limited hardware can be allocated if available
- Choice of multiple transfer paths could also be performed

Basics

- Mirror regular nonblocking collective operations
- For each nonblocking MPI collective (including neighborhood collectives), add a persistent variant
- For every MPI_I<coll>, add MPI_<coll>_init
- All parameters for the new functions will be identical to those for the corresponding nonblocking function
- All arguments "fixed" for subsequent uses
- Persistent collective calls cannot be matched with blocking or nonblocking collective calls

Init/Start

- The *init* function calls only perform initialization actions for their particular (collective) operation and do not start the communication needed to effect the operation
- Ex: MPI_Allreduce_init()
  - Produces a persistent request (not destroyed by completion)
- Works with MPI_Start (but NOT Startall)
- Only inactive requests can be started
- MPI_REQUEST_FREE can be used to free an inactive persistent collective request (similar to freeing persistent point-to-point requests)

Ordering of Init and Start

- Inits are non-blocking collective calls and must be ordered
- Similarly, persistent collective operations must be started in the same order at all processes in the corresponding communicator
- Startall cannot be used with persistent nonblocking collectives due to arbitrary ordering in the Startall operation

Example

Nonblocking Collective APIs	Persistent Collective APIs
	MPI_Bcast_init(..., &req[0]); MPI_Reduce_init(..., &req[1]);

for (i=0; i<MAXITER; i++) {
compute(bufA);
MPI_Ibcast(bufA,...,rowcomm, &req[0]);
compute(bufB);
MPI_Ireduce(bufB,...,colcomm, &req[1]);
MPI_Waitall(2, req, ...);
}
	for (i=0; i<MAXITER; i++) {
	compute(bufA);
	MPI_Start(req[0]);
	compute(bufB);
	MPI_Start(req[1]);
	MPI_Waitall(2, req, ...);
	}

Standardization Status

- Open ticket 466 – to be moved to Git
- Target: MPI-Next first standard release
- First reading – June, 2015
- Got feedback from the Forum
- Second reading – December
- Init/Start semantics have been clarified since first reading

Prototyping underway

- OpenMPI-based
- Replicate the libNBC library approach to interfacing with OpenMPI with persistence
- Explore improving persistent performance of point to point communication as well

Future Work

- Orthogonalization of the standard desirable
- We will extend ticket to non-blocking I/O collective operations
  - Other areas TBD
- We will explore missing non-persistent collective operations in standard too

Summary

- We want to get maximum performance when there are repetitive operations
- Evidence in the literature of efficacy
- Other approaches (*e.g.,* with Info arguments) are possible too
- Persistent, nonblocking collective operations provides the path to applications raising performance and predictability when there is reuse

Current Working Groups and Topics

- **Collectives & Topologies**
  - Torsten Hoefler, ETH
  - Andrew Lumsdaine, Indiana
- **Fault Tolerance**
  - Wesley Bland, ANL
  - Aurelien Bouteiller, UTK
  - Rich Graham, Mellanox
- **Fortran**
  - Craig Rasmussen, U. of Oregon
- **Generalized Requests**
  - Fab Tillier, Microsoft
- **Hybrid Models**
  - Pavan Balaji, ANL
- **I/O**
  - Quincey Koziol, HDF Group
  - Mohamad Chaarawi, HDF Group
- **Large count**
  - Jeff Hammond, Intel
- **Persistence**
  - Anthony Skjellum, Auburn Uni.
- **Point to Point Comm.**
  - Dan Holmes, EPCC
  - Rich Graham, Mellanox
- **Remote Memory Access**
  - Bill Gropp, UIUC
  - Rajeev Thakur, ANL
- **Tools**
  - Kathryn Mohror, LLNL
  - Marc-Andre Hermans, RWTH Aachen

MPI Working Group

Point to Point Communication

Current Topics

- MPI_Cancel for send requests
  - Discussion planned for December meeting
- Assertions as INFO keys
  - Active discussions ongoing in teleconferences
  - First formal reading planned for December meeting
- Allocating receive and freeing send
  - On hold

MPI_Cancel for send requests

- Current text on cancelling send requests
  - Does not capture original intent
  - Is vague and hard to implement
  - Only present for symmetry with receive requests
  - Never needed/used by users [sub: please check]
- Proposal to deprecate and eventually remove
  - See tickets #478, #479, and #480 (old trac system)
  - Soon to be replaced by issue in new git repository

Assertions as INFO keys

- Current INFO keys are hints to MPI
  - MPI cannot change semantics due to INFO key
  - User can lie, or set INFO keys to nonsense values
  - Some optimisations require stronger guarantees
- Proposal to allow assertions via INFO keys
  - User must comply with semantics for INFO keys
  - MPI is allowed to rely on validity of INFO keys
  - Implications for propagating INFO keys on dup
- See issue #11 in new git repository
  - https://github.com/mpi-forum/mpi-issues/issues/11

Propagating INFO keys

- MPI_COMM_[I]DUP propagate INFO keys
  - Libraries must query or remove INFO keys
  - Assertion INFO keys on parent communicator
    - Restrict behaviour of library using child communicator
- Proposal to prevent propagation of INFO keys
  - Probably should never have been required
  - Can still use MPI_COMM_DUP_WITH_INFO
  - Or (new) MPI_COMM_IDUP_WITH_INFO

New assertion INFO keys

- mpi_assert_no_any_tag
  - The process will not use MPI_ANY_TAG
- mpi_assert_no_any_source
  - The process will not use MPI_ANY_SOURCE
- mpi_assert_exact_length
  - Receive buffers must be correct size for messages
- mpi_assert_overtaking_allowed
  - All messages are logically concurrent

Meeting details

- Teleconference calls
  - Fortnightly on Monday at 11:00 central US
  - Next on 23rd November 2015
- Email list:
  - mpiwg-p2p@lists.mpi-forum.org
- Face-to-face meetings
  - meetings.mpi-forum.org/Meeting_details.php
  - Next on 7th-10th December in San Jose, CA

Current Working Groups and Topics

- **Collectives & Topologies**
  - Torsten Hoefler, ETH
  - Andrew Lumsdaine, Indiana
- **Fault Tolerance**
  - Wesley Bland, ANL
  - Aurelien Bouteiller, UTK
  - Rich Graham, Mellanox
- **Fortran**
  - Craig Rasmussen, U. of Oregon
- **Generalized Requests**
  - Fab Tillier, Microsoft
- **Hybrid Models**
  - Pavan Balaji, ANL
- **I/O**
  - Quincey Koziol, HDF Group
  - Mohamad Chaarawi, HDF Group
- **Large count**
  - Jeff Hammond, Intel
- **Persistence**
  - Anthony Skjellum, Auburn Uni.
- **Point to Point Comm.**
  - Dan Holmes, EPCC
  - Rich Graham, Mellanox
- **Remote Memory Access**
  - Bill Gropp, UIUC
  - Rajeev Thakur, ANL
- **Tools**
  - Kathryn Mohror, LLNL
  - Marc-Andre Hermans, RWTH Aachen

Remote Memory Access (RMA) Working Group Update

Rajeev Thakur

Brief Recap: What's New in MPI-3 RMA

- Substantial extensions to the MPI-2 RMA interface
- New window creation routines:
  - MPI_Win_allocate: MPI allocates the memory associated with the window (instead of the user passing allocated memory)
  - MPI_Win_create_dynamic: Creates a window without memory attached. User can dynamically attach and detach memory to/from the window by calling MPI_Win_attach and MPI_Win_detach
  - MPI_Win_allocate_shared: Creates a window of shared memory (within a node) that can be can be accessed simultaneously by direct load/store accesses as well as RMA ops
- New atomic read-modify-write operations
  - MPI_Get_accumulate
  - MPI_Fetch_and_op (simplified version of Get_accumulate)
  - MPI_Compare_and_swap

What's new in MPI-3 RMA contd.

- A new "unified memory model" in addition to the existing memory model, which is now called "separate memory model"
- The user can query (via MPI_Win_get_attr) whether the implementation supports a unified memory model (e.g., on a cache-coherent system), and if so, the memory consistency semantics that the user must follow are greatly simplified.
- New versions of put, get, and accumulate that return an MPI_Request object (MPI_Rput, MPI_Rget, …)
- User can use any of the MPI_Test/Wait functions to check for local completion, without having to wait until the next RMA sync call

MPI-3 RMA can be implemented efficiently

- "Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One Sided" by Robert Gerstenberger, Maciej Besta, Torsten Hoefler (SC13 Best Paper Award)
- They implemented complete MPI-3 RMA for Cray Gemini (XK5, XE6) and Aries (XC30) systems on top of lowest-level Cray APIs
- Achieved better latency, bandwidth, message rate, and application performance than Cray's MPI RMA, UPC, and Coarray Fortran

(a) Latency inter-node Put

(b) Message Rate inter-node

Application Performance with Tuned MPI-3 RMA

(a) Inserts per second for inserting 16k elements per process including synchronization.

Distributed Hash Table

(b) Time to perform one dynamic sparse data exchange (DSDE) with 6 random neighbors

Dynamic Sparse Data Exchange

(c) 3D FFT Performance. The annotations represent the improvement of FOMPI over MPI-1.

3D FFT

Figure 8: MILC: Full application execution time. The annotations represent the improvement of FOMPI and UPC over MPI-1.

MILC

Gerstenberger, Besta, Hoefler (SC13)

MPI RMA is Carefully and Precisely Specified

- To work on both cache-coherent and non-cache-coherent systems
  - Even though there aren't many non-cache-coherent systems, it is designed with the future in mind

- There even exists a *formal model* for MPI-3 RMA that can be used by tools and compilers for optimization, verification, etc.
  - See "Remote Memory Access Programming in MPI-3" by Hoefler, Dinan, Thakur, Barrett, Balaji, Gropp, Underwood. ACM TOPC, July 2015.
  - http://htor.inf.ethz.ch/publications/index.php?pub=201

Some Issues Currently Being Considered

- Clarifications to shared memory semantics
- New assertions for passive target epochs
- Nonblocking RMA epochs
- RMA notification at the target process
- Many others

Current Working Groups and Topics

- **Collectives & Topologies**
  - Torsten Hoefler, ETH
  - Andrew Lumsdaine, Indiana
- **Fault Tolerance**
  - Wesley Bland, ANL
  - Aurelien Bouteiller, UTK
  - Rich Graham, Mellanox
- **Fortran**
  - Craig Rasmussen, U. of Oregon
- **Generalized Requests**
  - Fab Tillier, Microsoft
- **Hybrid Models**
  - Pavan Balaji, ANL
- **I/O**
  - Quincey Koziol, HDF Group
  - Mohamad Chaarawi, HDF Group
- **Large count**
  - Jeff Hammond, Intel
- **Persistence**
  - Anthony Skjellum, Auburn Uni.
- **Point to Point Comm.**
  - Dan Holmes, EPCC
  - Rich Graham, Mellanox
- **Remote Memory Access**
  - Bill Gropp, UIUC
  - Rajeev Thakur, ANL
- **Tools**
  - Kathryn Mohror, LLNL
  - Marc-Andre Hermans, RWTH Aachen

MPI Tools Working Group Update

Kathryn Mohror Lawrence Livermore National Lab
Marc-Andre Hermanns Jülich Aachen Research Alliance

What is the Tools WG about?

- Tools support
  - Debuggers
  - Correctness tools
  - Performance analysis tools
  - Could really be anything
- As of MPI-3.0
  - Profiling Chapter -> Tool Support Chapter
    - PMPI interface still exists
    - New section for new Tool Information Interface, MPI_T
  - Documented the “MPIR” and “MQD” interfaces for debuggers
    - Ad hoc, largely undocumented interface
    - *Documented*, not standardized

MPI_T interface appears to be in good shape

- On the way to MPI 3.1
  - Lots of feedback from community
    - Tools people and MPI implementors
  - Errata
    - 19 Errata to MPI 3.0
    - Good thing! People are using the interface!
  - Feature updates
    - Handful of small changes
      - Quick look up of variables
      - Add a couple new return codes
      - Minor clarifications
      - Specify that some function parameters are optional

What's happening now?

- New interface to replace PMPI
  - Known, longstanding problems with the current profiling interface PMPI
    - One tool at a time can use it
    - Forces tools to be monolithic (a single shared library)
    - The interception model is OS dependent
  - New interface
    - Callback design
    - Multiple tools can potentially attach
    - Maintain all old functionality
- New feature for event notification in MPI_T
  - PERUSE
  - Tool registers for interesting event and gets callback when it happens

What's happening now?

- Debugger support - MPIR interface
  - Fixing some bugs in the original "blessed" document
    - Missing line numbers!
  - Support non-traditional MPI implementations
    - Ranks are implemented as threads
  - Support for dynamic applications
    - Commercial applications/ Ensemble applications
    - Fault tolerance
  - Handle Introspection Interface
    - See inside MPI to get details about MPI Objects
      - Communicators, File Handles, etc.

I have ideas. Can I join in the fun?

- Yes!
- Join the mailing list
  - http://lists.mpi-forum.org/
  - mpiwg-tools
- Join our meetings
  - https://github.com/mpiwg-tools/tools-issues/wiki/Meetings
- Look at the wiki for current topics
  - https://github.com/mpiwg-tools/tools-issues/wiki

The MPI-Forum Organization and Call for Participation

Martin Schulz
LLNL / CASC
Chair of the MPI Forum
MPI Forum BOF @ SC15, Austin, TX

http://www.mpi-forum.org/

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

The MPI Forum Drives MPI

- **Standardization body for MPI**
  - Discusses additions and new directions
  - Oversees the correctness and quality of the standard
  - Represents MPI to the community
- **Organization consists of chair, secretary, convener, steering committee, and member organizations**
- **Open membership**
  - Any organization is welcome to participate
  - Consists of working groups and the actual MPI forum
  - Physical meetings 4 times each year (3 in the US, one with EuroMPI/Asia)
    - Working groups meet between forum meetings (via phone)
    - Plenary/full forum work is done mostly at the physical meetings
  - Voting rights depend on attendance
    - An organization has to be present two out of the last three meetings (incl. the current one) to be eligible to vote

Typical Way to Add Items to MPI

1. **New items should be brought to a matching working group for discussion**
   - Creation of preliminary proposal
   - Simple (grammar) changes are handled by chapter committees

Typical Way to Add Items to MPI

1. **New items should be brought to a matching working group for discussion**
   - Creation of preliminary proposal
   - Simple (grammar) changes are handled by chapter committees
2. **Socializing of idea driven by the WG**
   - Could include plenary presentation to gather feedback
     - Focused on concepts not details like names or formal text
   - Make proposal easily available through WG wiki
   - Important to keep overall standard in mind
3. **Development of full proposal**
   - Latex version that fits into the standard
   - Creation of a github issue to track voting and a matching pull request
4. **MPI forum reading/voting process**

The MPI Voting Process

- **Quorum**
  - 2/3 of eligible organizations have to be present
  - 3/4 of present organization have to vote yes
  - Goal: standardize only if there is consensus
- **Steps**
  1. Reading: "Word by word" presentation to the forum
  2. First vote
  3. Second vote
- **Each step has to be at a separate physical meeting**
  - Ensure people have time to think about additions
  - Avoid hasty mistakes, which are hard to fix
  - Prototypes are encouraged and helpful to convince people

How To Get Involved

- **Submit comments to the MPI forum**
  - mpi-comments@mpi-forum.org
  - Feedback on prototypes / proposals as well as the existing standard
- **Subscribe to email lists to see what's going on**
  - Each working group has its own mailing list
- **Join a working group**
  - Check out the respective Wiki pages
  - Participate in WG meetings (typically phone conference)
  - Contact the WG chairs to introduce yourself
- **Participate in physical MPI forum meetings**
  - December 2015, San Jose, CA, USA
  - March 2016, Chicago, IL, USA
  - Logistics and agendas available through the MPI forum website
  - Drop me an email if you have questions or are interested
- **More information at: http://www.mpi-forum.org/**

http://www.mpi-forum.org/docs/docs.html

MPI Documents

www.mpi-forum.org/docs/docs.html

MPI Documents

The official version of the MPI documents are the English Postscript versions (for MPI 1.0 and 1.1) and PDF (for the other versions). In several cases, a translation or HTML version is also available for convenience. The HTML version was made with automated tools. In case of a difference between these two sources, the Postscript or PDF version of MPI standard documents are always considered the official version. In the case of multiple PDF versions, only the one described as the "MPI x.y document as PDF" is the official version; the versions provided with alternate formatting are provided as a convenience and are not official (every effort has been taken to make them "the same", but no guarantee is made).

Those who prefer to get the documents via anonymous ftp may do so at ftp.mpi-forum.org in pub/docs/.

Some translations of MPI documents are available.

MPI-3.1

MPI-3.1 was approved by the MPI Forum on June 4, 2015.

- MPI 3.1 document as PDF
- Versions of MPI 3.1 with alternate formatting
- Errata for MPI 3.1
- Unofficial HTML version of MPI 3.1. This version was produced with tohtml.

The complete, official MPI-3.1 Standard (June 2015) is available in one book (hardcover, 868 pages, sewn binding).
For all MPI programmers, the standard can be used as a complete MPI reference manual with many examples and advices to users. With MPI-3.1, a general index was added that supplements the other indices.
It was printed and is sold at cost by HLRS; see http://www.hlrs.de/mpi/mpi31/ .

Previous Versions of the MPI Standard

MPI-3.0

MPI-3.0 was approved by the MPI Forum on September 21, 2012.

http://meetings.mpi-forum.org/

MPI Forum Meetings

meetings.mpi-forum.org

MPI Forum Meetings

Meetings home

Search

- The MPI Forum
  - Home
- Meetings
  - Logistics/Agendas
  - Secretary notes (past mtgs.)
- Ongoing Technical Efforts
  - MPI 4.0 effort
  - Wiki
- Completed Technical Efforts
  - MPI 3.1 effort
  - MPI 3.0 effort
  - MPI 2.2 effort
  - MPI 2.1 effort
- Other MPI Sites
  - Official MPI documents
  - MPI errata
  - MPI Forum mailing lists

This website contains information about the activities of the MPI Forum, which is the standardization forum for the Message Passing Interface (MPI).

Updates

- **Implementation Status (MPI 3.1), as of November 2015**
- **BoF at SC 15, Nov. 18th, 2015**
  - Room 15, Austin Convention Center, Austin, TX
  - News from the MPI Forum, incl. MPI 3.1
  - Activities from all MPI Forum Working Groups
  - How can you get involved?"
- **Implementation Status (MPI 3.0), as of March 2015**
- **BoF at SC 14, Nov. 19th, 2014**
  - Introduction and General Update - Martin Schulz
  - Proposals for MPI Fault Tolerance - Wesley Bland
  - The MPI Endpoints Proposal for better MPI+X Support - Pavan Balaji
  - Ideas for persistant Pt-to-Pt Communication - Daniel Holmes (Editable Version of the Document to Add Comments)
  - How to Participate in the MPI Forum

Meeting Coordinators:

- Martin Schulz, Meeting Chair
- Richard Graham, Convener
- Jeff Squyres, Meeting Secretary
- Bill Gropp, Editor

Previous Efforts:

- Martin Schulz, MPI 3.1

GitHub Organization: https://github.com/mpi-forum

MPI Forum

https://github.com/mpi-forum

Search GitHub | Pull requests | Issues | Gist

MPI Forum

Home of the MPI Forum's Repositories. See mpi-issues for general issues and wiki pages.

http://www.mpi-forum.org mpi-forum@lists.mpi-forum.org

Repositories | People 16 | Teams 4

Filters | Find a repository...

meetings.mpi-forum.org — PHP ★ 0 ⑂ 0
The meetings.mpi-forum.org web site
Updated 4 days ago

mpi-standard PRIVATE — TeX ★ 0 ⑂ 13
Official repository for the MPI Standard source
Updated 11 days ago

mpir PRIVATE — TeX ★ 0 ⑂ 4
Repository for the MPIR Interface

People 16 >

https://github.com/mpi-forum/mpi-issues/wiki

Home · mpi-forum/mpi-is

https://github.com/mpi-forum/mpi-issues/wiki

This repository Search Pull requests Issues Gist

mpi-forum / mpi-issues

Unwatch 17 Star 2 Fork 1

Home

Edit New Page

Jeff Squyres edited this page 16 days ago · 34 revisions

Welcome

Welcome to the MPI Standard issues repository.

This repository is public to allow anyone to participate in the MPI Forum by viewing and contributing to issues. The associated source code for the MPI Standard however, is kept private to avoid allowing unofficial copies of the MPI Standard to be easily created (see below for how to request access to this private repository).

Getting Started

If you're just getting started with Git, GitHub, and/or the MPI Forum's usage of Git+GitHub, you should probably read the following wiki pages in order.

1. Git and GitHub definitions and notation
2. Overview of the MPI Forum's GitHub usage

Pages 18

Find a Page...

- Home
- Access to the MPI Forum private repository
- Branch Management
- Clone the main MPI standards document repo
- Creating Simple Pull Request
- Fork the MPI Forum mpi standard repo (on GitHub)
- Git and GitHub definitions and notation
- How to file an MPI Forum issue ("ticket")
- Initial Git setup

Open Tickets/Issues

Search/filter for: “is:open is:issue user:mpi-forum”

Issues

https://github.com/issues?utf8=✓&q=is%3Aopen+is%3Aissue+user%3Ampi-forum+

Search GitHub | Pull requests | Issues | Gist

Created | Assigned | Mentioned

is:open is:issue user:mpi-forum

17 Open | 5 Closed | Visibility | Organization | Sort

- mpi-forum/mpi-issues **User-Level Failure Mitigation: Files** mpi-4.0 not ready wg-ft
  #22 opened 7 days ago by wesbland 2016-02 Chicago, USA
- mpi-forum/mpi-issues **User-Level Failure Mitigation: RMA** mpi-4.0 not ready wg-ft
  #21 opened 7 days ago by wesbland 2016-02 Chicago, USA
- mpi-forum/mpi-issues **User-Level Failure Mitigation** mpi-4.0 not ready wg-ft 1
  #20 opened 7 days ago by wesbland 2016-02 Chicago, USA
- mpi-forum/mpi-issues **make MPI_THREAD_MULTIPLE a requirement for all implementations** 5
  #19 opened 9 days ago by jeffhammond
- mpi-forum/mpi-issues **Longer types for use with MPI_MINLOC and MPI_MAXLOC** not ready 2
  #18 opened 9 days ago by jeffhammond
- mpi-forum/mpi-issues **clarify what functions can be called on uncommitted datatypes** not ready 6
  #17 opened 9 days ago by jeffhammond
- mpi-forum/mpi-issues **deprecate MPI_GRAPH_CREATE** not ready
  #16 opened 9 days ago by jeffhammond
- mpi-forum/mpi-issues **MPI_Reduce_scatter advice to users is wrong** errata 3
  #15 opened 9 days ago by jeffhammond
- mpi-forum/mpi-issues **deprecate MPI_COMM_JOIN** mpi <next> not ready

www.EuroMPI2016.ed.ac.uk

- Call for papers open by end of November 2015
- Full paper submission deadline: 1st May 2016

- Associated events: tutorials, workshops, training
- Focusing on: benchmarks, tools, applications, parallel I/O, fault tolerance, hybrid MPI+X, and alternatives to MPI and reasons for not using MPI

Summary

http://www.mpi-forum.org/

- **MPI Forum is an open forum**
  - Everyone / every organization can join
  - Want/Need/Encourage community feedback
- **Major work in the next few years on MPI 4**
  - Several major initiatives, incl.
    - Fault Tolerance
    - Better support for hybrid programming
    - Performance Hints and Assertions
  - Many smaller proposals as well
- **Get involved**
  - Let us know what you or your applications need
  - Let us know where MPI is lacking for your needs
  - Help close these gaps!