STATUS OF OPENFABRICS INTERFACES (OFI) SUPPORT IN MPICH

Yanfei Guo, Computer Scientist

Argonne National Laboratory
OVERVIEW

- What is MPICH?
- Why OFI?
- Current Support
- Future Plan
WHAT IS MPICH?

- MPICH is a high-performance and widely portable open-source implementation of MPI
- It provides all features of MPI that have been defined so far (up to MPI-4.0)
- Active development lead by Argonne National Laboratory and University of Illinois at Urbana-Champaign
  - Several close collaborators who contribute features, bug fixes, testing for quality assurance, etc.
    - IBM, Microsoft, Cray, Intel, Ohio State University, Queen’s University, Mellanox, RIKEN AICS and others
- www.mpich.org
MPICH: GOAL AND PHILOSOPHY

- MPICH aims to be the preferred MPI implementation on the top machines in the world
- Our philosophy is to create an “MPICH Ecosystem”
Why OFI/OFIWG?

- Support for diverse hardware through a common API
- Actively, openly developed
  - Hosted on Github
- Close abstraction for MPI
  - MPI community engaged from the start
- Fully functional sockets provider
  - Prototype code on a laptop
- Strong Vendor Support
- MPIX Stream prototype
- Standalone PMI Library
- MPICH Testsuite
  - Comprehensive testsuite for MPI implementations in general
  - Now available as separate release target
- Accelerate CI builds
  - CI is key for productivity, we do hundreds of CI builds daily
  - Projects are getting more complex, and slower to build
  - Option to prebuild submodules, ./autogen.sh -quick to avoid repeated rebuild
- Improved ch4 and yaksa stability
• **Native GPU Data Movement**
  – Multiple forms of “native” data movement
  – GPU Direct RDMA is generally achieved through Libfabrics or UCX (we work with these libraries to enable it)
  – GPU Direct IPC is integrated into MPICH

• **GPU Fallback Path**
  – GPU Direct RDMA may not be available due to system setup (e.g. library, kernel driver, etc.)
  – GPU Direct IPC might not be possible for some system configurations
  – GPU Direct (both forms) might not work for noncontiguous data
  – Datatype and Active Message Support
  – New GPU-aware datatype engine

The GPU support in MPICH is developed in close collaboration with vendor partners including AMD, Cray, Intel, Mellanox and NVIDIA.

OSU_LATENCY internode fastpath (w/ gdrcopy)
MPIX_STREAM – EXPLICITLY TELL MPI ABOUT THREAD CONTEXT

- **MPIX_Stream** identifies a serial execution context

```c
int MPIX_Stream_create(MPI_Info info, MPIX_Stream *stream)
int MPIX_Stream_free(MPIX_Stream *stream)
```

- *info* can be `MPI_INFO_NULL`, identifies a generic thread context
- In the case of threads, it is the application’s responsibility to ensure access to an `MPIX_Stream` is serialized. Essentially `MPI_THREAD_SERIAL`, but at the object-level, rather than all of MPI.

---

Stream communicator is a communicator with local streams attached.

```c
int MPIX_Stream_comm_create(MPI_Comm parent_comm,
                             MPIX_Stream stream, MPI_Comm *stream_comm)
```

- MPIX streams are local, but communications are between pairs of them
- Otherwise, synchronization is unavoidable at receiver or sender.
- It okay for `stream` to be `MPIX_STREAM_NULL`.
- Conventional communicators are the same as stream communicators with `MPIX_STREAM_NULL` on every process.
STREAM COMMUNICATION IMPROVES ON IMPLICIT VCI PERFORMANCE

- Implicit VCI mapping in MPICH-4.0 scales well with multiple threads
- Advice to users
  - Use different communicators
  - Same communicator, use different tags and set hints
- Explicit VCI with MPIX_Stream communicator in MPICH-4.1 removes thread safety overhead

© OpenFabrics Alliance
STANDALONE PMI

- **PMI remains an internal component in MPICH**
- **Supporting both PMI-1 and PMI-2 is confusing**
  - PMI-1 is the default in MPICH/Hydra, well tested
  - PMI-2 is/was experimental, not feature-complete, less stable
  - Slurm documents PMI-2, but supports PMI-1
  - Cray supports PMI-2
- **Interest in using PMI/Hydra independently from MPICH**
  - PMI interface is a universal interface that works everywhere MPI works
  - Hydra is a robust and versatile launcher
  - PMI/Hydra works well for multi-process runtimes, e.g. OpenSHMEM, NVSHMEM
- **Need to extend PMI/Hydra to support modern PMI features**
  - To (partially) support PMIx
Better configure options
• --with-pmi={pmi1,pmi2,pmix}
• --with-pmilib={mpich,slurm,cray,pmix}
• --with-pm={no,hydra,gforker,remshell}
• --with-pmi={slurm,cray} also works

Separate release targets
• pmi-4.1.tar.gz and hydra-4.1.tar.gz
• Provide libpmi.so

Consistent PMI headers
• Third party PMI implementation should support the same pmi.h and pmi2.h

Internal refactoring
• PMI-1 and PMI-2 are internally unified
• Wire protocol layer and semantic layer are separated
MPICH 4.2 ROADMAP

- MPICH-4.2a1 2H 2023
- MPICH-4.2b1 targeted for SC23
  - 4.2.x branch will be created
- GA release in early 2024
- Critical bug fixes will be backported to 4.1.x
MPICH 4.2 FEATURES

- Experimental Thread Communicator for interthread communication
- Experimental support for MPI-5 ABI proposal
- PMIx support in libpmi and Hydra
- Improved support for MPI-4 partitioned communication
- More in plan
 MPIX THREAD COMMUNICATOR

- Make MPI available for inter-thread communication within parallel regions.
- Break down thread and process silos for a unified communication API.

```c
int MPIX_Threadcomm_init(MPI_Comm comm, int num_threads,
                          MPI_Comm threadcomm)
#pragma omp parallel {
    MPIX_Threadcomm_start(threadcomm);
    /* use threadcomm within parallel region */
    MPIX_Threadcomm_finish(threadcomm);
}
```
MPIX THREAD COMMUNICATOR - STATUS

- Open pull request, will merge soon
- ✓ Point-to-point functions (blocking and non-blocking, intra- and inter-node)
- ✓ Blocking collectives (single algorithm)
- Non-blocking collectives
- Collective algorithm tuning
- Communicator functions
- ? One-sided communication
SUPPORT MPI-5 ABI

- A working proposal currently being developed in MPI Forum
- Build once, work with either MPICH or Open MPI derivatives
- MPICH-4.2 will support both MPICH ABI and optionally MPI-5 ABI
  - mpicc builds MPICH ABI, mpicc_abi builds MPI-5 ABI
  - libmpi.so implements MPICH ABI, libmpi_abi.so implements MPI-5 ABI
  - mpi.h will effectively become mpi_abi.h when mpicc_abi is used. User code always #include <mpi.h>
THANK YOU

Yanfei Guo, Computer Scientist

Argonne National Laboratory