Endpoints Plenary

James Dinan
Hybrid Working Group
December 10, 2013
Status of Endpoints

1. Proposal text #380 ready
2. Explored interoperability story [EuroMPI ‘13]
3. Exploring performance story [IJHPCA in prep]
4. Working on implementation in MPICH
   – Will be open source
5. Target: Formal reading in March
Motivation for Endpoints

1. Interoperability argument
   – On-node programming model
   – Multi-node models that use threads

2. Performance argument
   – Increase communication concurrency
     • Preserve shared memory/node-level programming
     • Make number of VA spaces free parameter
   – Reduce synchronization penalties
     • Privatize thread communication state and resources
Achievable Network Performance (Dramatization)

- Network endpoint design evolving to support many cores
- *Not real data, represents my personal views*
- Gathering real data for paper, will present at next meeting
Impact of Queue Depth on Message Rate

- Brian Barrett, et al. [EuroMPI ‘13]
- Threads sharing a rank increase posted receive queue depth (x-axis)
Mapping of Ranks to Processes

• MPI provides a 1-to-1 mapping of ranks to processes
• This was good in the past
• Usage models and systems have evolved
  – Hybrid MPI+Threads programming
  – Ratio of core to network endpoint performance decreasing
Endpoints Model

- Many-to-one mapping of ranks to processes
  - Threads act as first-class participants in MPI operations
  - Improve programmability of MPI + X
  - Threads drive independent network endpoints

- Endpoint: Set of resources that supports the independent execution of MPI communications
  - Endpoints have process semantics
Current THREAD_MULTIPLE Usage

- MPI message matching space: <communicator, sender, tag>
- Two approaches to using THREAD_MULTIPLE

1. Match specific thread using the tag:
   - Partition the tag space to address individual threads
   - Limitations:
     - Collectives – Multiple threads at a process can’t participate concurrently
     - Wildcards – Multiple threads concurrently requires care

2. Match specific thread using the communicator:
   - Split threads across different communicators (e.g. Dup and assign)
   - Can use wildcards and collectives
   - However, limits connectivity of threads with each other
Implementation of Endpoints

- Two implementation strategies
  1. Each rank is a network endpoint
  2. Ranks are multiplexed on endpoints
     - Effectively adds destination rank to matching
  3. Combination of the above

- Potential to reduce threading overheads
  - Separate resources per thread
    - Rank can represent distinct network resources
    - Increase HFI/NIC concurrency
  - Separate software state per thread
    - Per-endpoint message queues/matching
    - Enable per-communicator threading levels

- FG-MPI implements “static” endpoints
  - A little different, still demonstrates implementation and performance benefits
Endpoints Interface

```c
int MPI_Comm_create_endpoints(
    MPI_Comm parent_comm, int my_num_ep,
    MPI_Info info, MPI_Comm *out_comm_hdl[])
```

– Out handle array takes TLS out of the implementation and off the critical path
– Each process requests an independent number of endpoints
– MPI_ERR_ENDPOINTS – Endpoints could not be created
Endpoints Proposal

https://svn.mpi-forum.org/trac/mpi-forum-web/ticket/380