

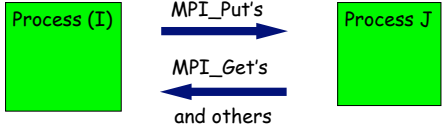
Empowered by Innovation **NEC**

Proposal

(More) Flexible RMA Synchronization for MPI-3

Hubert Ritzdorf
NEC-IT Research Division
ritzdorf@it.neclab.eu

The Problem



How can **Process J** decide that **Process (I)** has finished RMA transfer so that **Process J** can access/change the data ?

MPI-2: Collective approach (Fence, Complete/Wait) on Windows
The group has to be clear in advance; not very flexible

New Proposal

Shared Memory: Atomic counters, ready flags, semaphores, ...
It's not necessarily defined, which process provides the data, decrements the counter, sets the flag, ...

© NEC Europe Ltd 2008 Empowered by Innovation **NEC**

Idea

Window:
Large amount of (shared) memory
Atomic counter (synchronization counters):
To independently handle parts of the (shared) memory

New MPI Type MPI_Sync: MPI Synchronization counter

Special Allocate:
MPI_Win_Sync_alloc_objects (n_sync, sync_counters, win, info)
MPI_Win_Sync_free_objects (n_sync, sync_counters, win)
to be performed by the owner of the window win

© NEC Europe Ltd 2008 Empowered by Innovation **NEC**

Problem

How to transfer the info on MPI_Sync counters

- Every process (in win comm) should be able to get access
- Access to counter may change in run-time
- Processes must have required info to ``attach'' to the counter

Idea:

- New MPI Datatype MPI_Handle_sync
- Transfer data via MPI pt2pt and collectives

Possible alternative : Create n_sync counters in MPI_Win_create
Pro: User doesn't see them; Contra: libraries, less flexibility, scalability

© NEC Europe Ltd 2008 Empowered by Innovation **NEC**

Usage

On processes which perform RMA operations

`MPI_Win_sync_ops_init (target_rank, sync_mode, sync_counter, win, info, req)`

Init

- wait for completion of RMA operations corresponding to `sync_mode` : Bit vector or of `MPI_MODE_WIN_PUT`, `MPI_MODE_WIN_GET`, `MPI_MODE_WIN_ACCUMULATE`
- increment/decrement counter `sync_counter` on process `target_rank`

© NEC Europe Ltd 2008 Empowered by Innovation **NEC**

Usage

On process where counter is located

`MPI_Win_sync_object_init (sync_counter, count, win, info, req)`

Init

- wait for `count` completions (atomic increments) of requests created by `MPI_Win_sync_ops_init`

Designed as persistent request for frequent re-use

- All possibilities of MPI Test/Wait functions
- Works on cache coherent and non cache coherent systems
- Can be implemented by pt2pt functions

© NEC Europe Ltd 2008 Empowered by Innovation **NEC**

Example

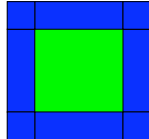
Target

```
MPI_Win_sync_object_init (sync_counter, 8, win, info, req);
MPI_Start (req)
```

8 neighbours

```
MPI_Win_sync_ops_init (target_rank MPI_MODE_PUT, sync_counter, win, info, req); MPI_Start (req)
```

`MPI_Wait (req, ...)`



`MPI_Put (target, ...)`

`MPI_Wait (req, ...)`

© NEC Europe Ltd 2008 Empowered by Innovation **NEC**

Wait for completion of which RMA ops ?

With current MPI-2 Interface:

- Wait for completion of all RMA requests since last synchronization

Not really save programming style.
Difficult for threads or several counters

Additional functions for MPI-3 interface (useful for threads and several synchronization counters)

Add synchronization counter request to RMA functions for example:

`MPI_Iput (... , win, req), MPI_Iget (... , win, req), ...`

© NEC Europe Ltd 2008 Empowered by Innovation **NEC**

Window as Shared Memory

More dynamic usage of window required
(for example for dynamic process spawning)

`MPI_Win_change_comm (new_comm, peer_rank, win)`

Allow processes to [join](#) or [leave](#) an already existing window.

Possible Extensions

Synchronization counters == atomic counters

- Wait/Test that counter completed on remote node
- Change final counter value (increment, decrement)

Automatic restart of request if
final counter value is reached and
test/wait function was executed