Mysteries of the Deep: What happens inside of MPI on Blue Gene/Q and why it matters

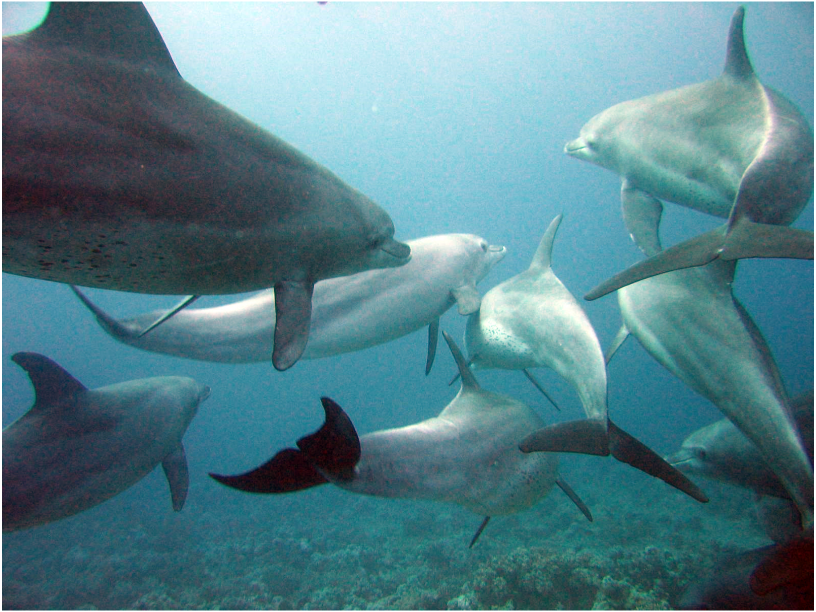
Jeff Hammond

Leadership Computing Facility
Argonne National Laboratory

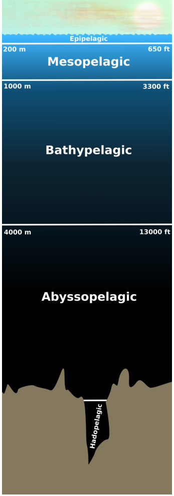
May 23, 2012

The view from the boat

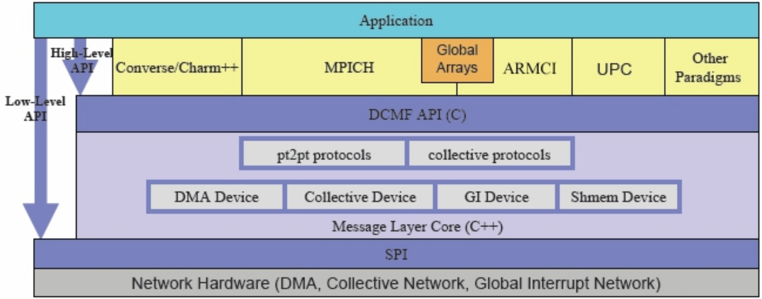
A reason to dive



But not too deep

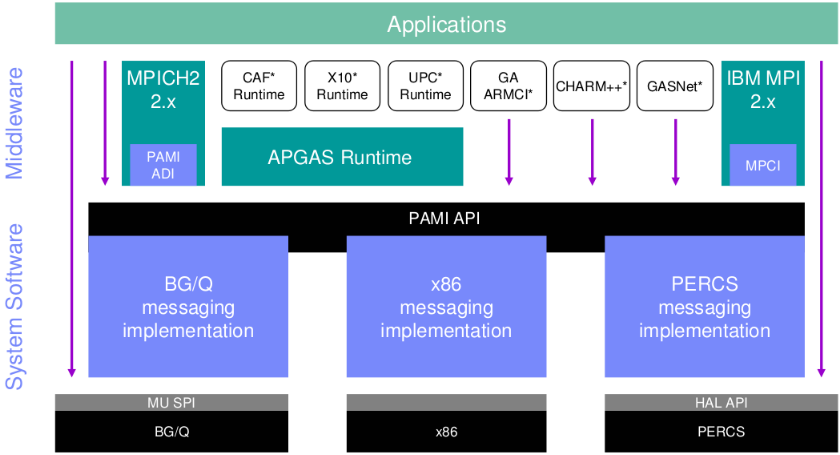


Blue Gene/P Communication architecture



Source: IBM

Blue Gene/Q Communication architecture



*all runtimes are not supported on all platforms

Performance results

Neighbor exchange

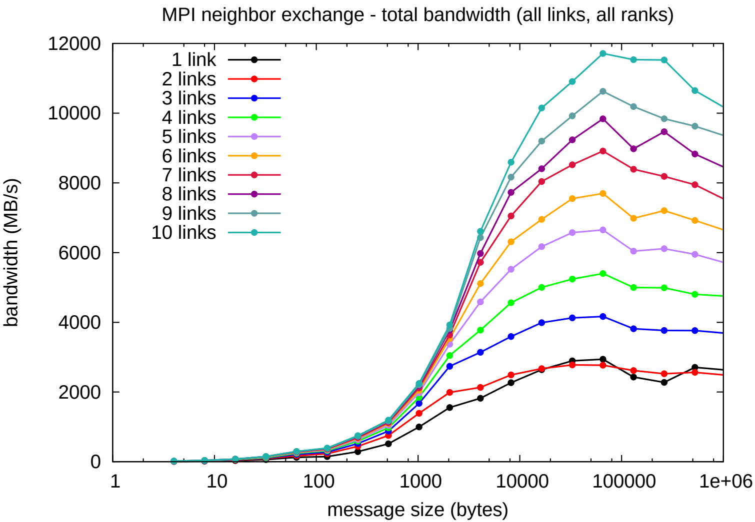
Testing injection (send) bandwidth along 1 to 10 links.

Explicitly mapped to torus.

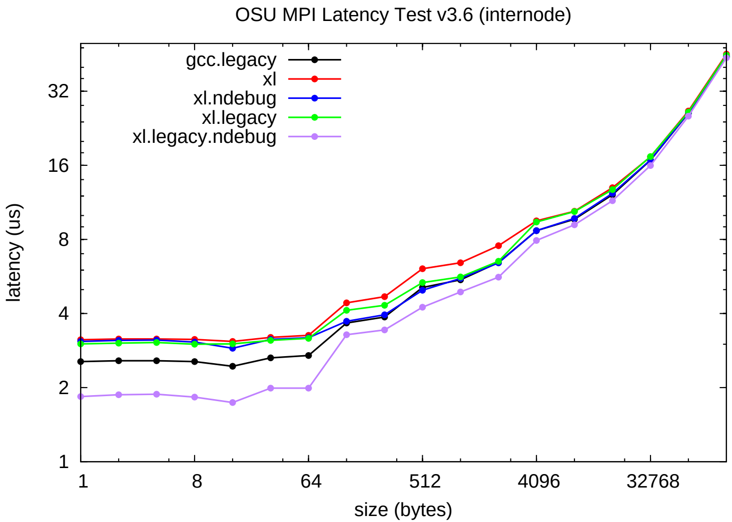
Nonblocking recv and send followed by waitall.

No repetition in test but warmup along all 6 links done first.

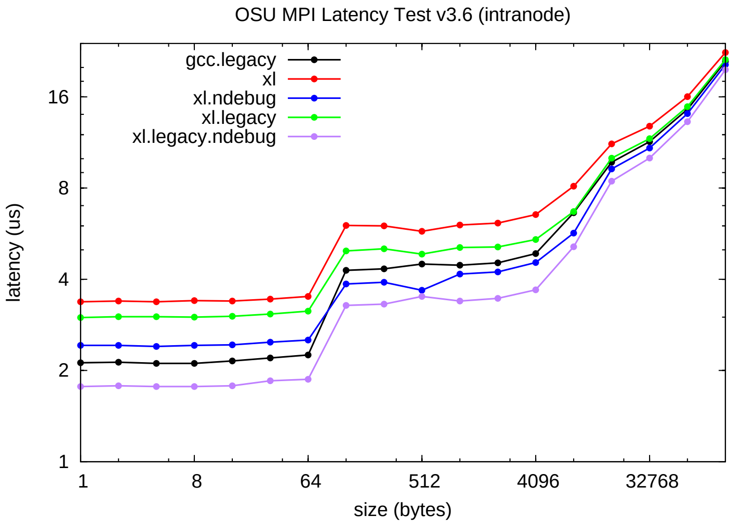
Neighbor exchange



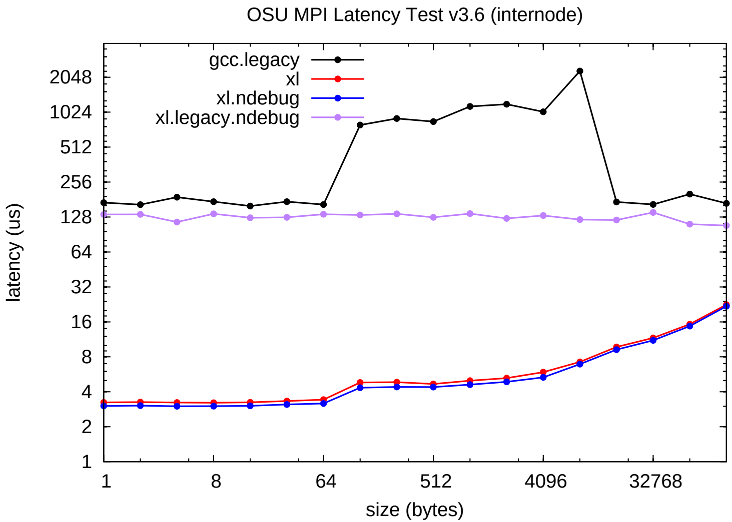
OSU MPI Benchmarks



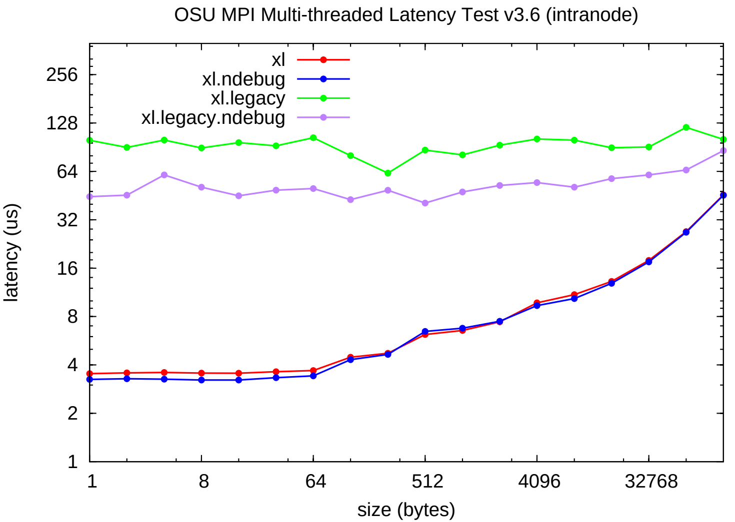
OSU MPI Benchmarks



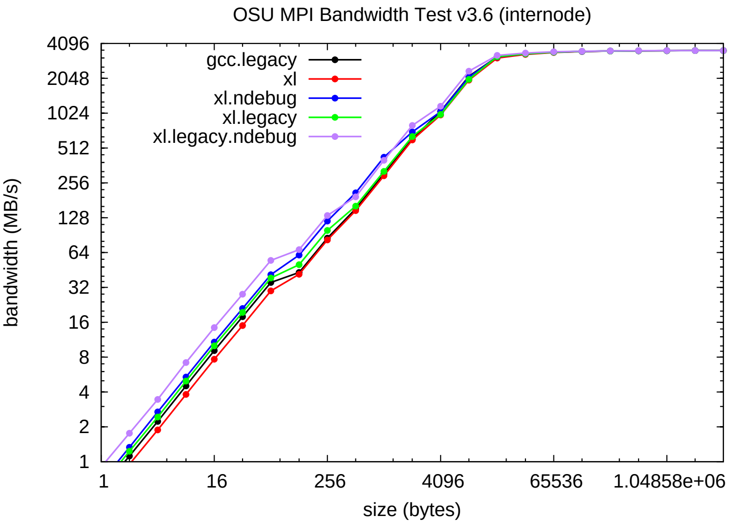
OSU MPI Benchmarks



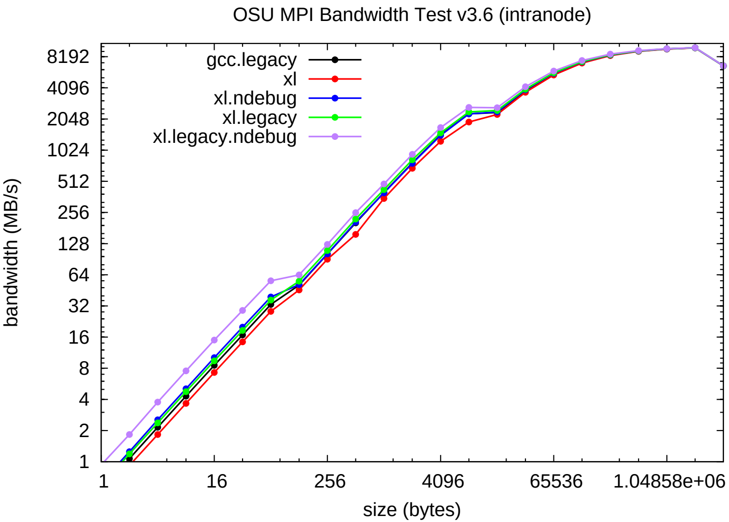
OSU MPI Benchmarks



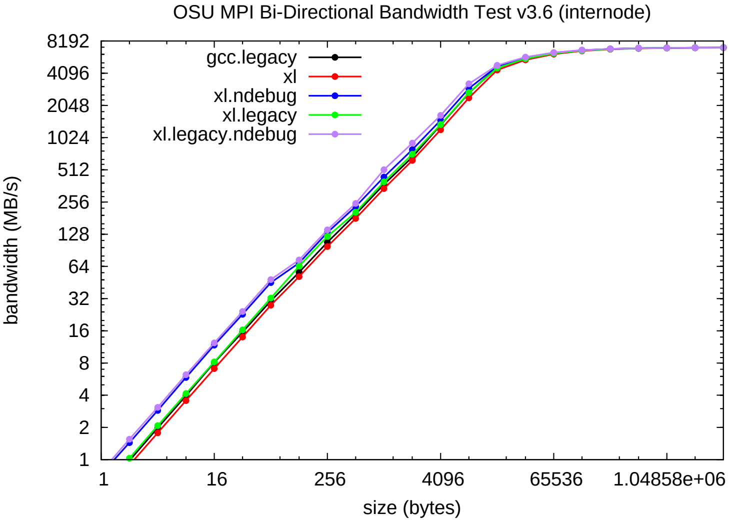
OSU MPI Benchmarks



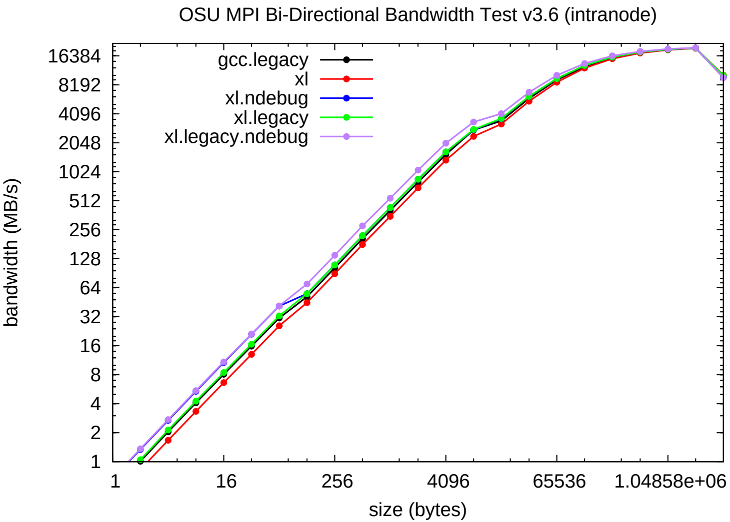
OSU MPI Benchmarks



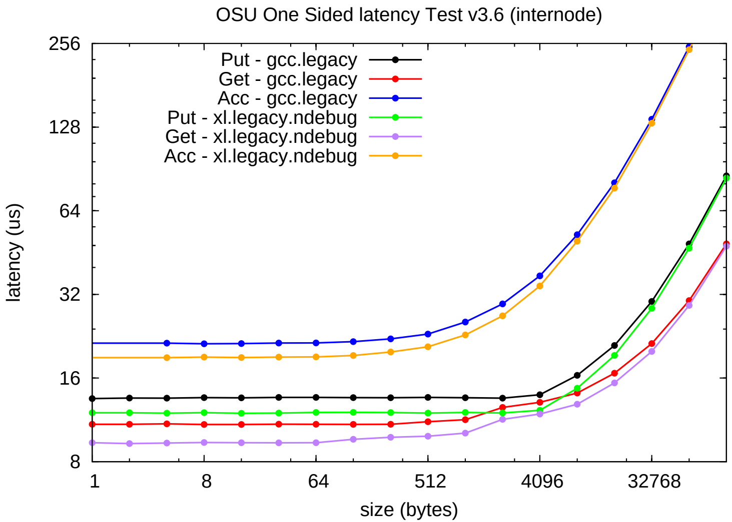
OSU MPI Benchmarks



OSU MPI Benchmarks



OSU MPI Benchmarks



Topology and MPIX

Topology API

From /bgsys/drivers/ppcfloor/comm/*/include/mpix.h

```
#define MPIX_TORUS_MAX_DIMS 5
typedef struct
{
    unsigned ppn;
    unsigned coreID;
    unsigned torus_dimension;
    unsigned Size[MPIX_TORUS_MAX_DIMS];
    unsigned Coords[MPIX_TORUS_MAX_DIMS];
    unsigned isTorus[MPIX_TORUS_MAX_DIMS];
} MPIX_Hardware_t;
```

Topology API

```
MPIX_Hardware_t hw;
MPIX_Hardware(&hw);

int hopCoord = (hw.Coords[0]+1) % (hw.Size[0]);
int tempCoords[MPIX_TORUS_MAX_DIMS+1] =
    { hopCoord, hw.Coords[1], hw.Coords[2],
      hw.Coords[3], hw.Coords[4], hw.coreID};
MPIX_Torus2rank(tempCoords, &rank_ap);
```

Environment variables

PAMI Verbosity

PAMID_STATISTICS
Turns on statistics printing for the message layer such as the maximum receive queue depth. *Disabled by default.*

PAMID_VERBOSE
Increases the amount of information dumped during an MPI_Abort() call. *Disabled by default.*

PAMI High-level tuning options

PAMID_CONTEXT_POST
Handoff-based communication. *Enabled by default (I think).*

Optimizing collectives through thinking

MPI_Reduce_scatter: reduce a buffer to root, then scatter from root. This MPI-2.1 function requires vector arguments, so this is really reduce, then scatterv. MPI_Scatterv is not optimized. In addition, the arguments to scatterv must be allocated internally. At scale, this consumes a lot of memory (perhaps as much as 8x).

MPI_Reduce_scatter_block: reduce a buffer to root, then scatter from root. This MPI-2.2 function takes scalar arguments and therefore uses less memory at scale.

MPI_Allreduce: reduce a buffer everywhere, then copy out my portion. This MPI-1 function requires scalar arguments and is highly optimized on Blue Gene/P. Using MPI_IN_PLACE means no extra memory is used (although the input buffer is modified).

On BGQ, allreduce will be faster than reduce at least some of the time.

Closing thoughts

"People first, then money, then things" – Suze Orman

"Science first, then algorithms, then communication" – Me

However, you know your algorithms are good and communication is holding you back, understanding the internals of MPI will help you write faster and more scalable code on Blue Gene/P.

P.S. If profiling shows you spend too much time in MPI_Barrier, your *algorithm* is the problem (load imbalance), not communication.