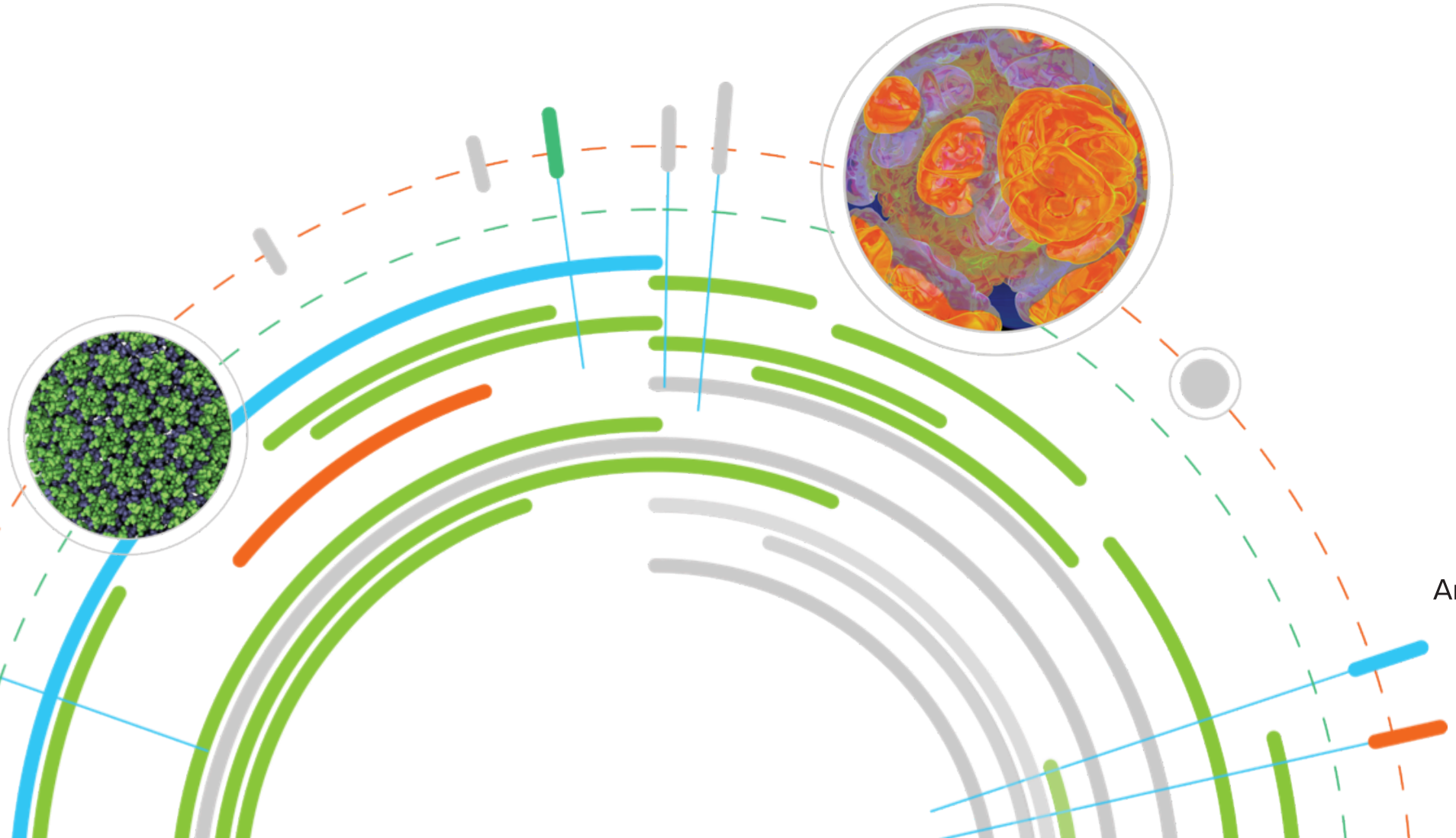
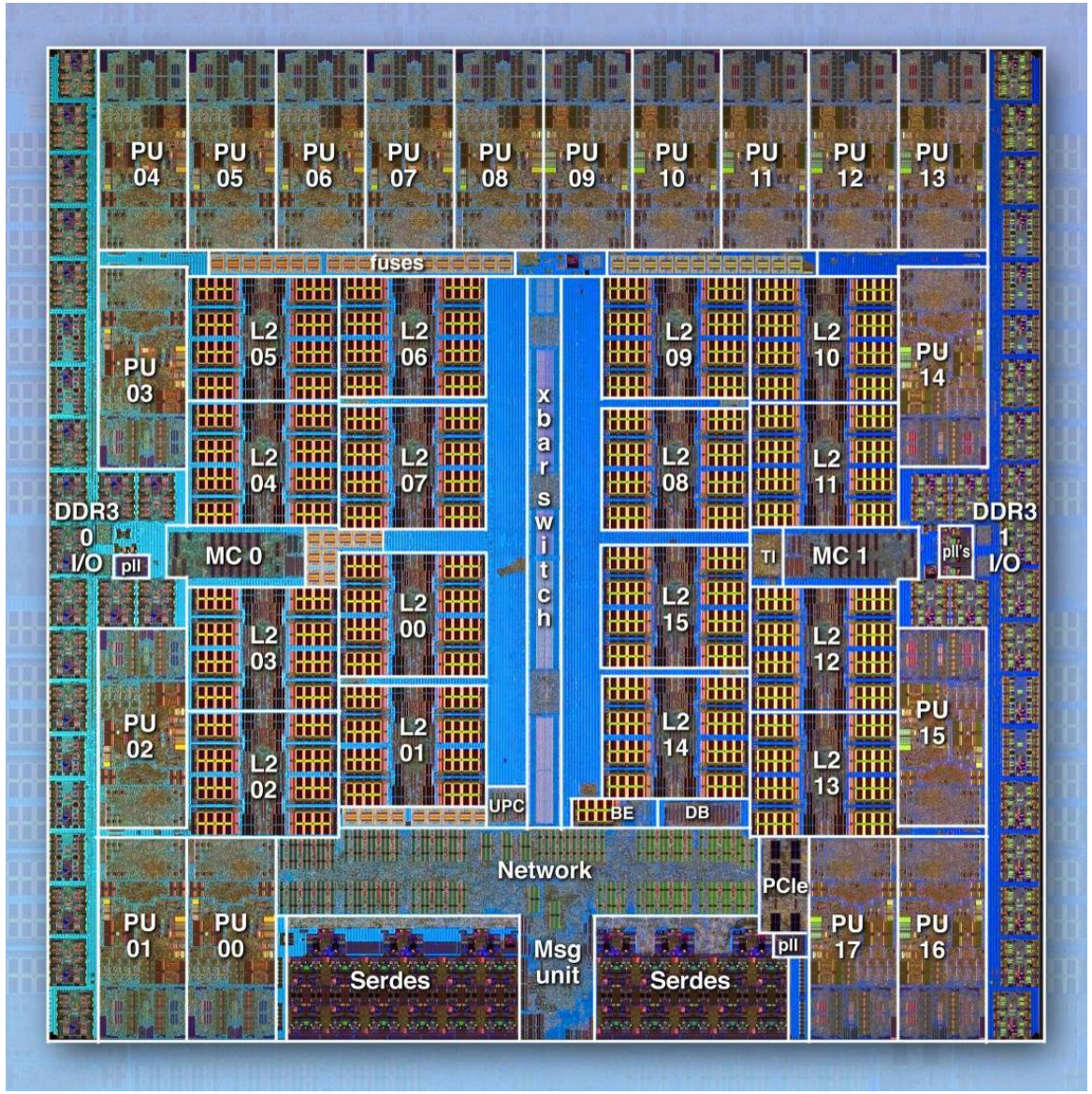
ALCF Systems 2: Inter-Node



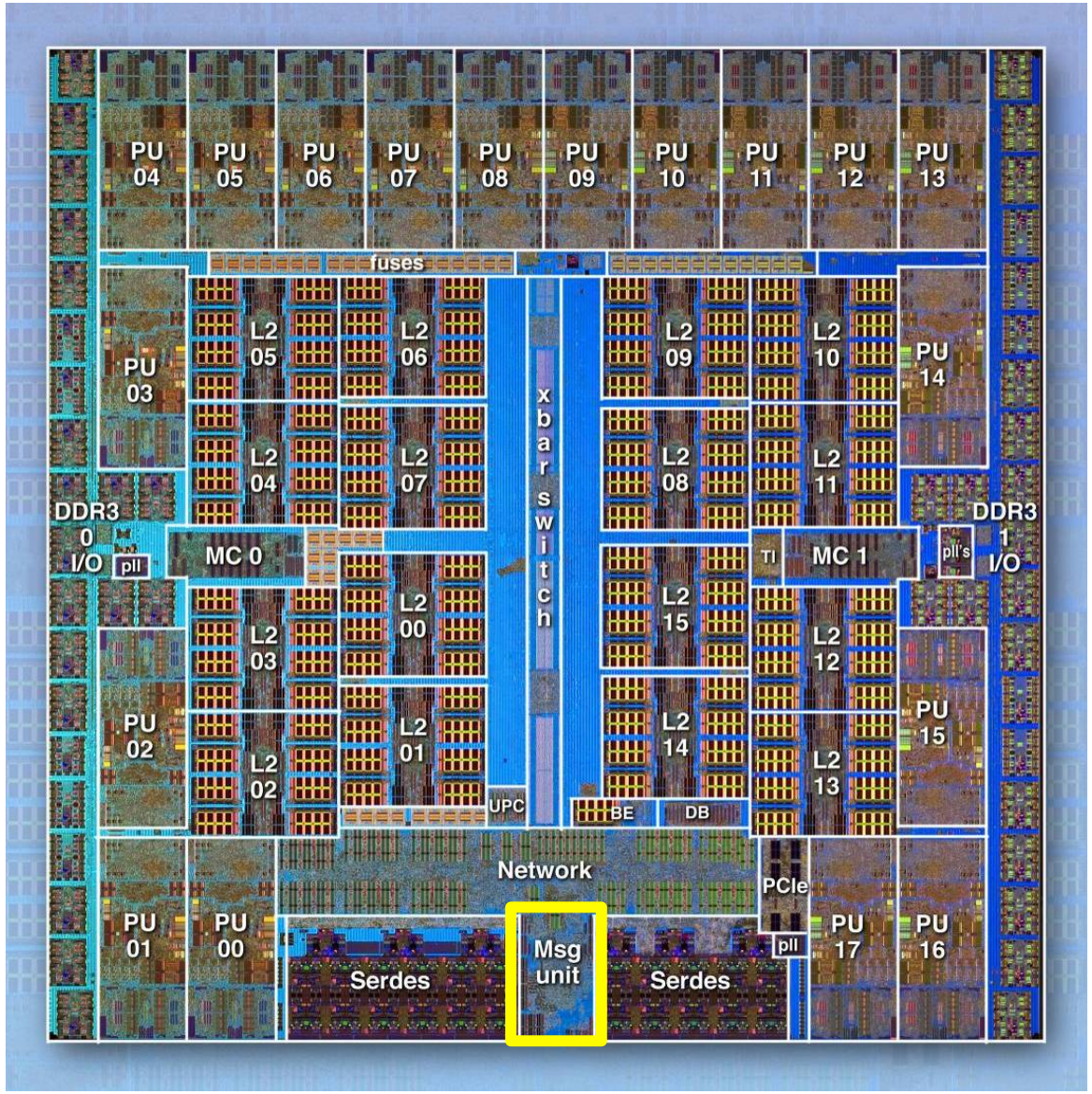
Argonne **Leadership Computing** Facility

Argonne NATIONAL LABORATORY

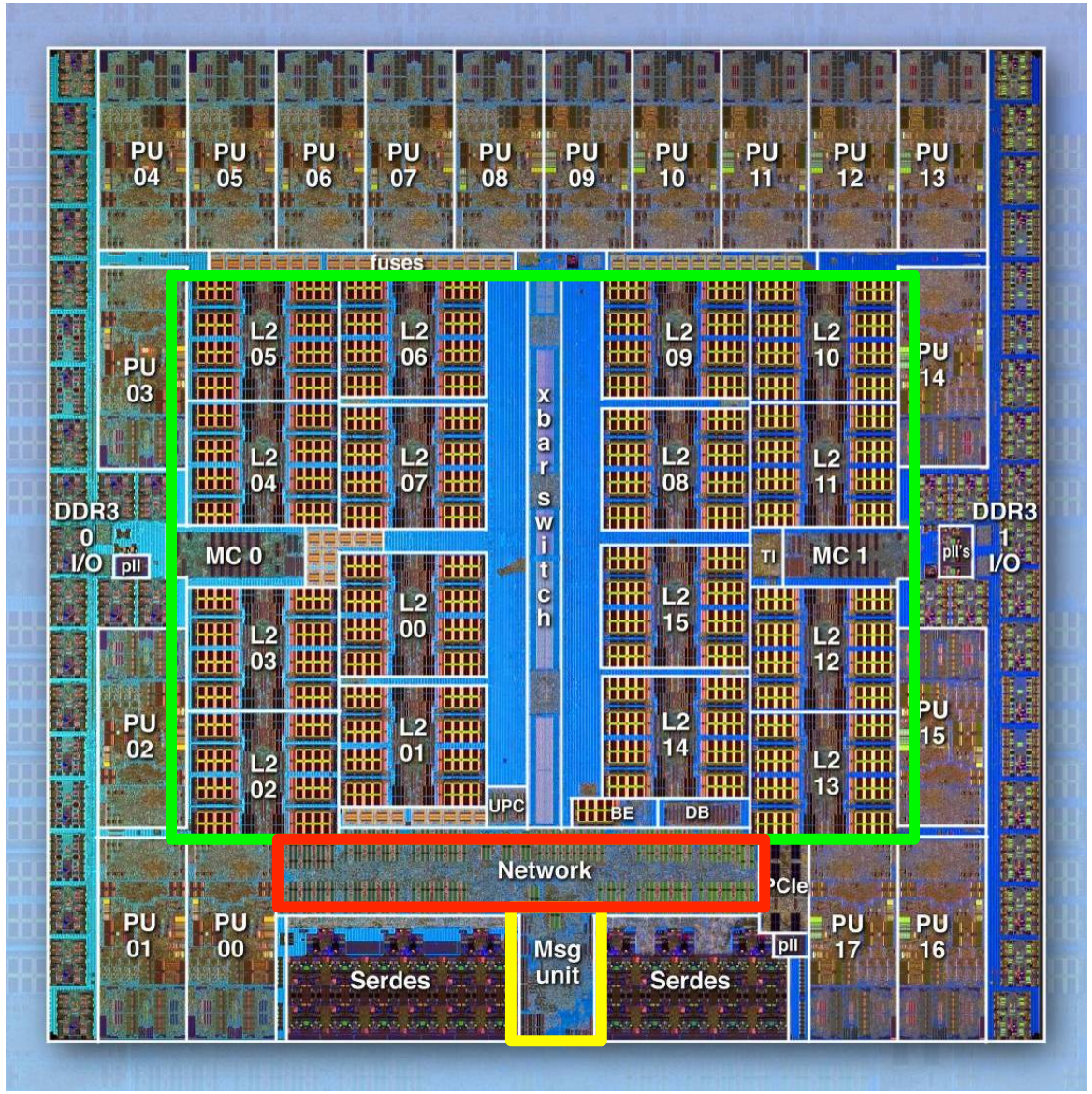
BG/Q Compute Chip



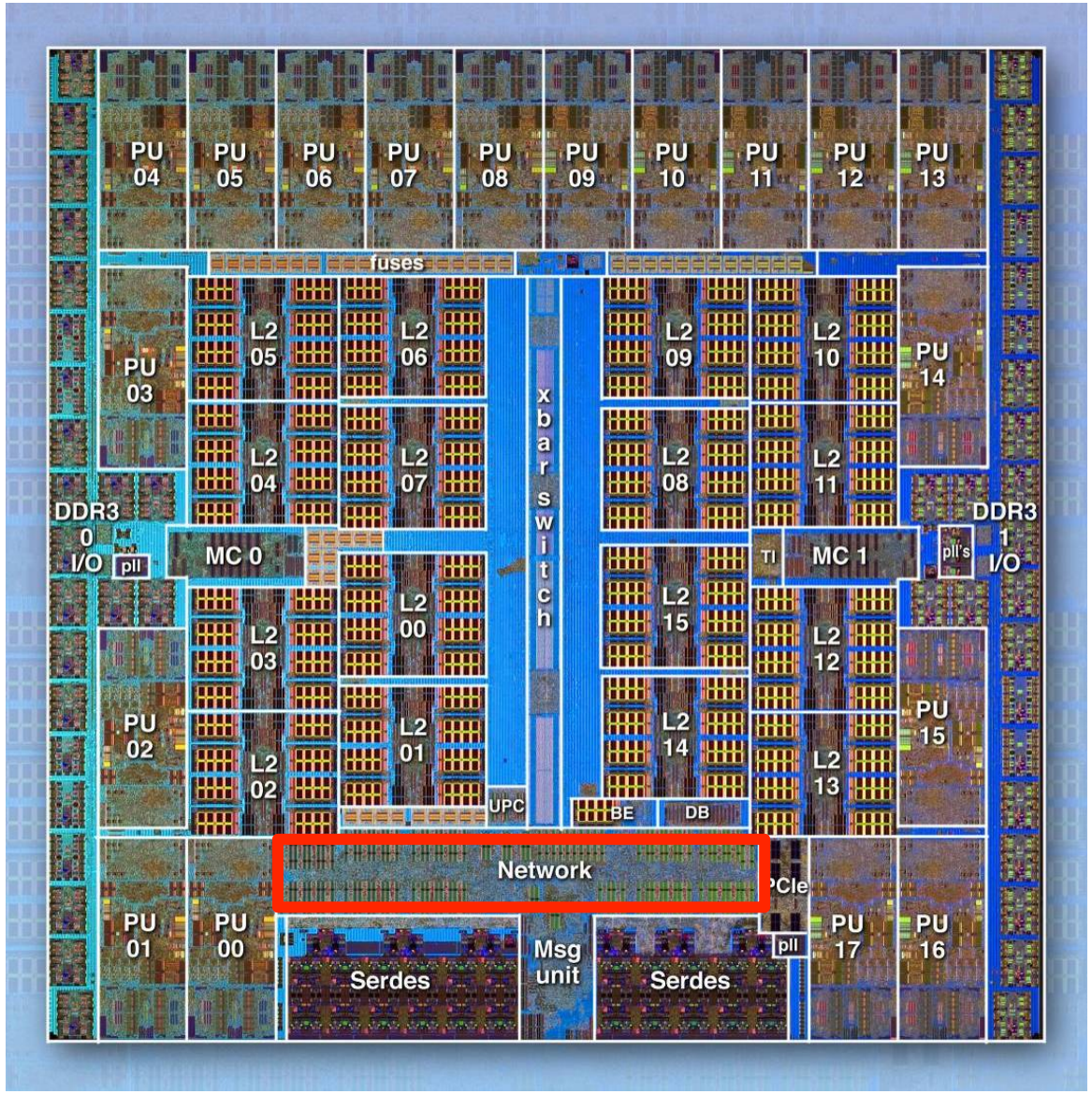
BG/Q Compute Chip



BG/Q Compute Chip



BG/Q Compute Chip

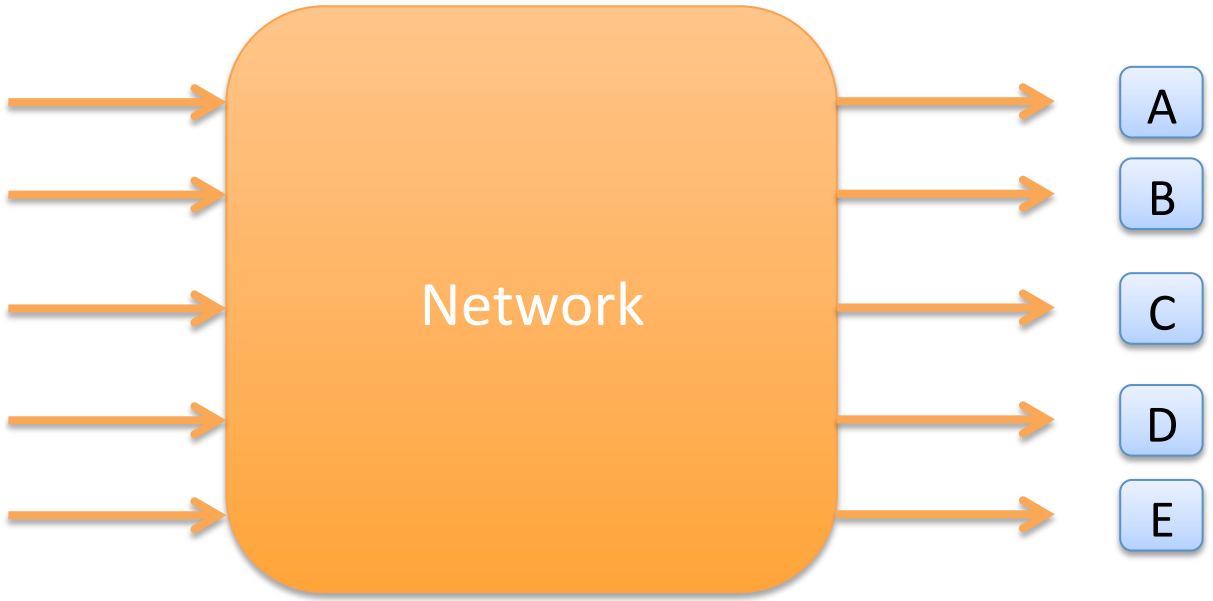


BG/Q Network

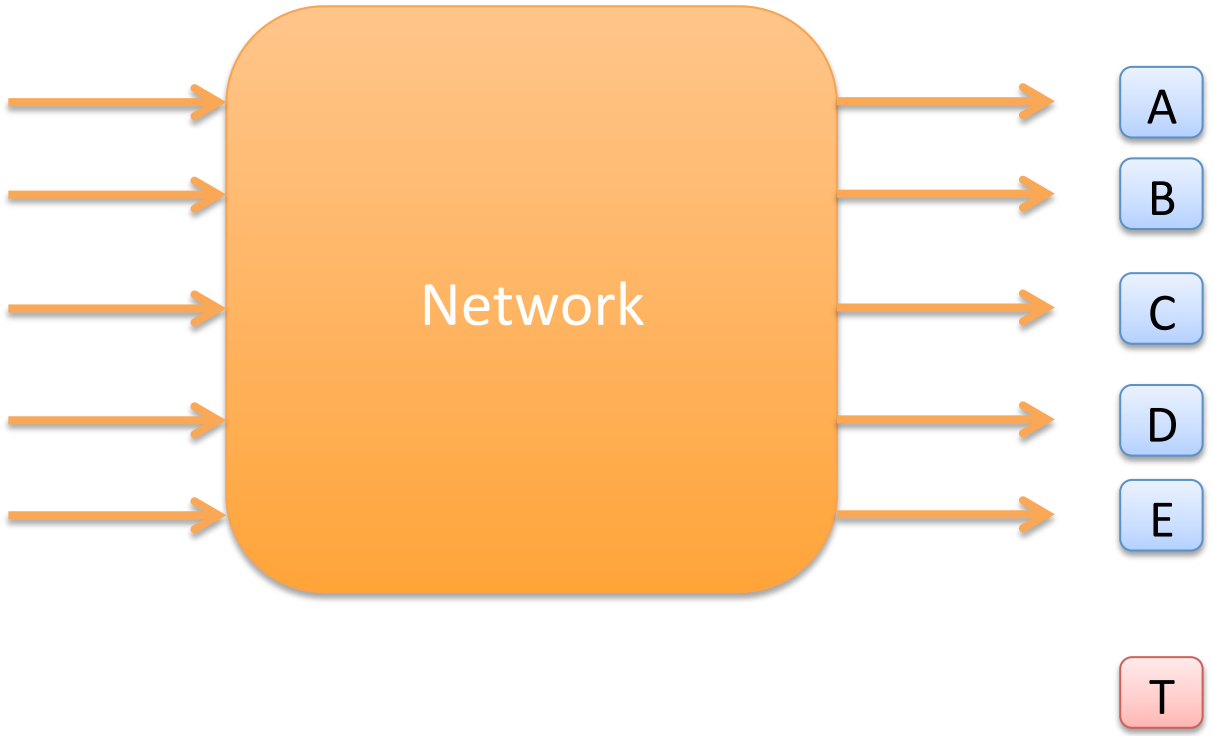
BG/Q Network

5D Torus

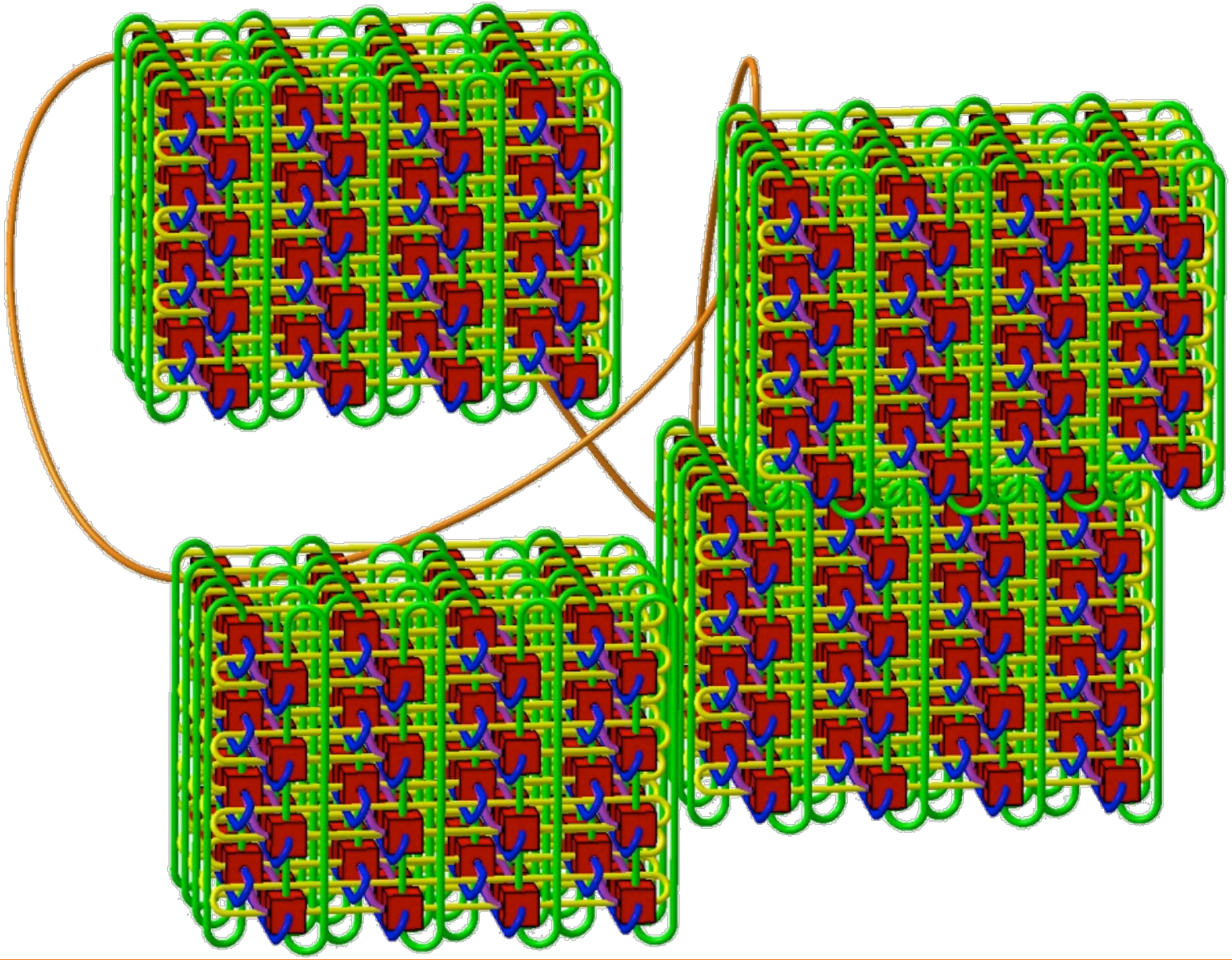
BG/Q Network



BG/Q Network

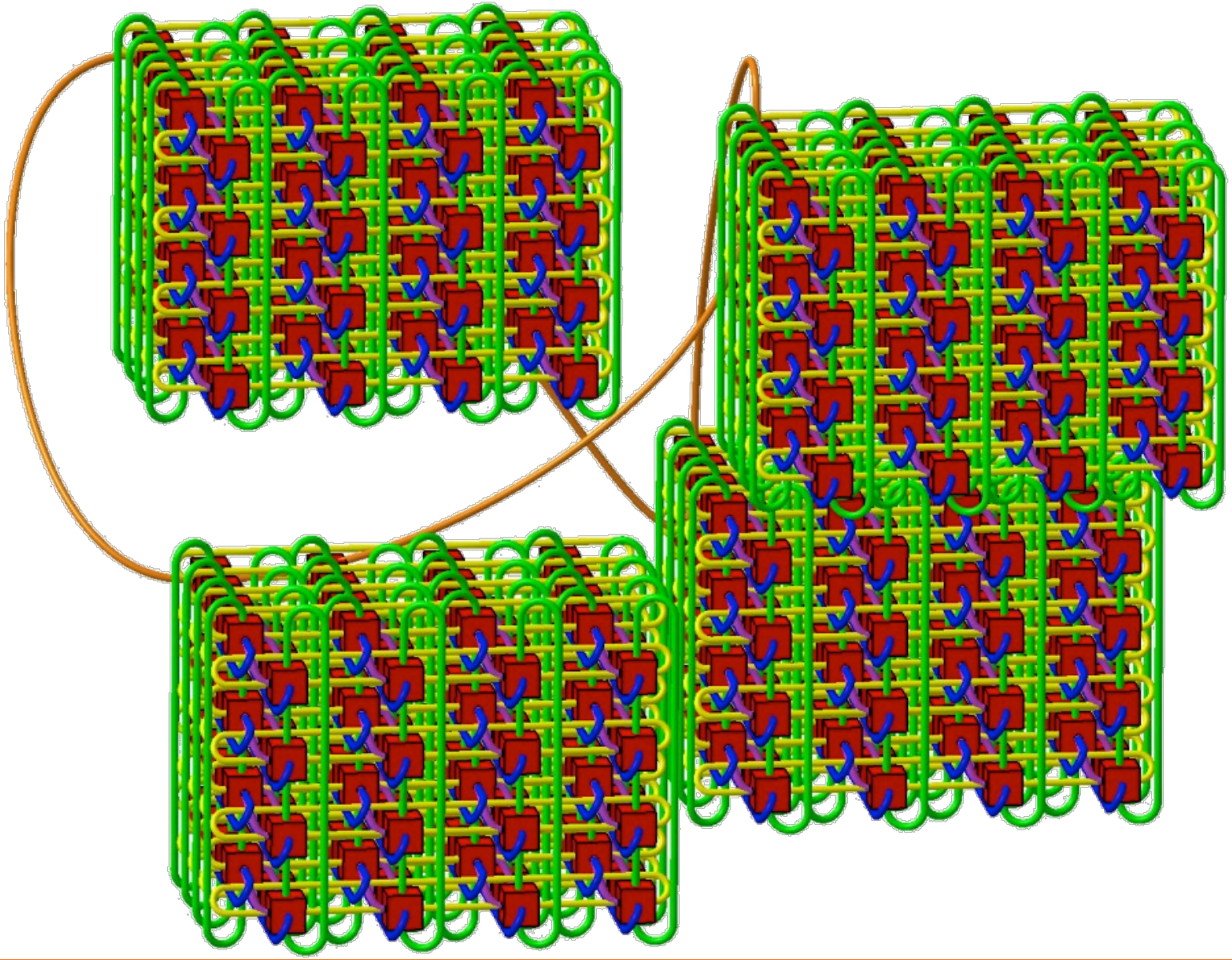


BG/Q 512 Node Torus Partition



BG/Q 512 Node Torus Partition

4 × 4 × 4 × 4 × 2



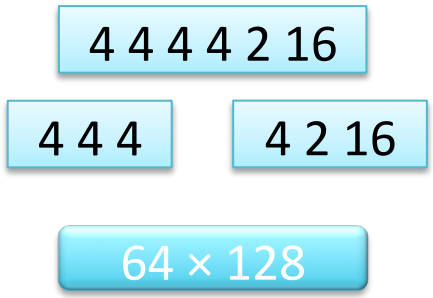
Mapping Ranks/Processes to Nodes

- Permutation of ABCDET
  - ABCDET on midplane --mode c1 <4,4,4,4,2,1>

    Rank 0 coordinates <0,0,0,0,0,0>
    Rank 1 coordinates <0,0,0,0,1,0>
    Rank 2 coordinates <0,0,0,1,0,0>
    Rank 3 coordinates <0,0,0,1,1,0>
    Rank 4 coordinates <0,0,0,2,0,0>
    Rank 5 coordinates <0,0,0,2,1,0>
    Rank 6 coordinates <0,0,0,3,0,0>
    Rank 7 coordinates <0,0,0,3,1,0>
    Rank 8 coordinates <0,0,1,0,0,0>
    …
    Rank 511 coordinates <3,3,3,3,1,0>
  - runjob --mapping TEDCBA
- Mapping file
  - 0 0 0 0 0 0 # rank 0
    0 0 0 0 1 0 # rank 1
    0 0 0 1 0 0 # rank 2
    …
  - runjob --mapping *mapfilename*

Mapping Ranks/Processes to Nodes (cont'd)

- **Goal:** in cartesian topology
  - Preserve locality for nearest-neighbor
  - Minimize extra hops in partition
- Example: 2D logical topology
  - Midplane c16 <4,4,4,4,2,16>



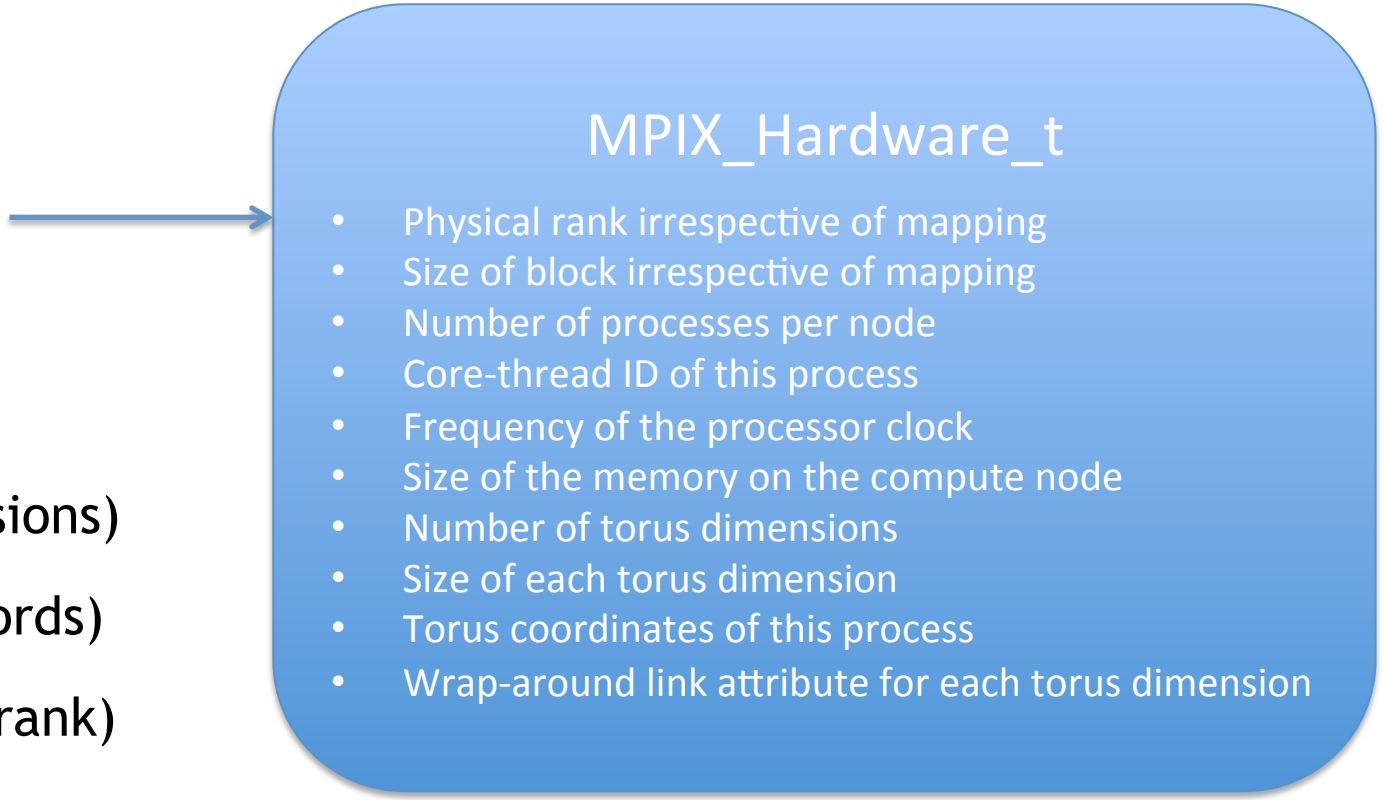
- Two ways to implement
  1. Generate map file
  2. Order the ranks in a new MPI communicator

```
MPI_Comm_split(MPI_COMM_WORLD, color, key, new2DComm);
```

Order in 64 × 128

Topology Access: MPIX

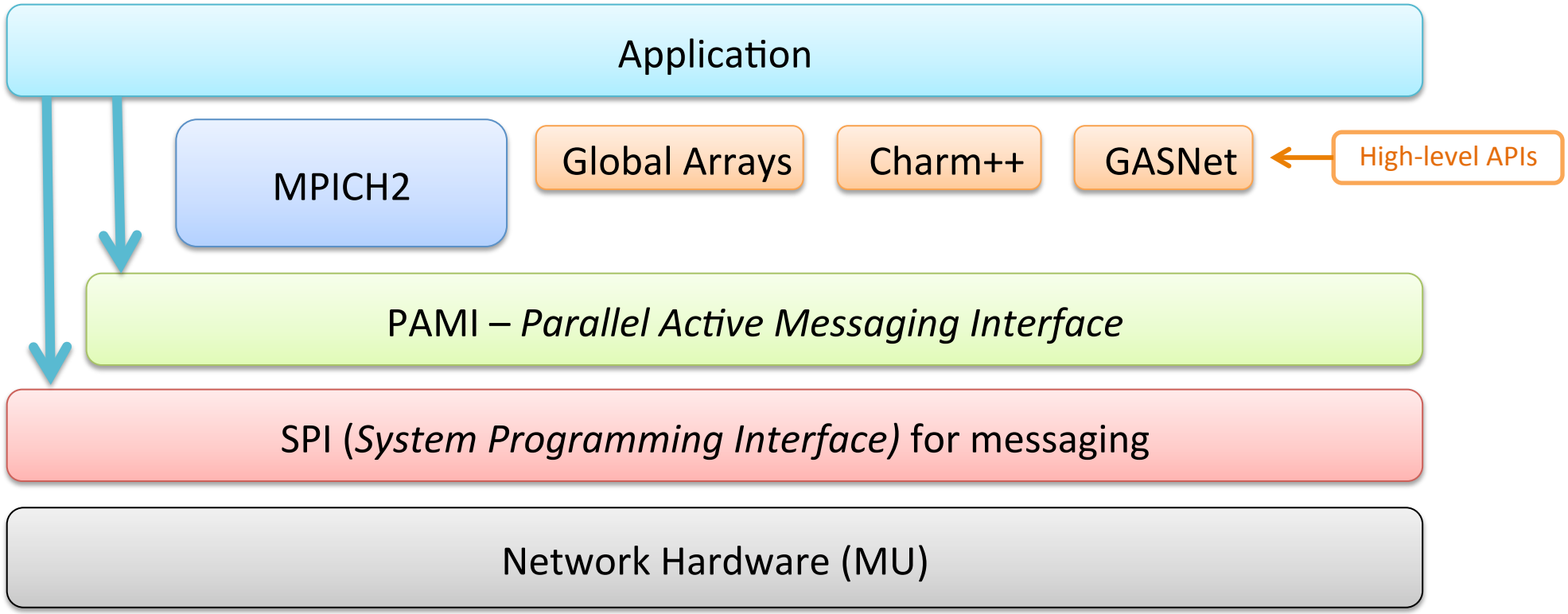
```
#include <mpix.h>

MPIX_Init_hw(MPIX_Hardware_t *hw)

int MPIX_Torus_ndims(int *numdimensions)

int MPIX_Rank2torus(int rank, int *coords)

int MPIX_Torus2rank(int *coords, int *rank)
```



Network Speed is a Major Strength of BG/Q

- Each A/B/C/D/E link bandwidth: 4 GB/s
- Bisection bandwidth (32 racks): 13.1 TB/s
- HW latency
  - Best: 80 ns (nearest neighbor)
  - Worst: 3 µs (96-rack 20 PF system, 31 hops)
- MPI latency (zero-length, nearest-neighbor): 2.2 µs

Blue Gene/Q Communication Programming



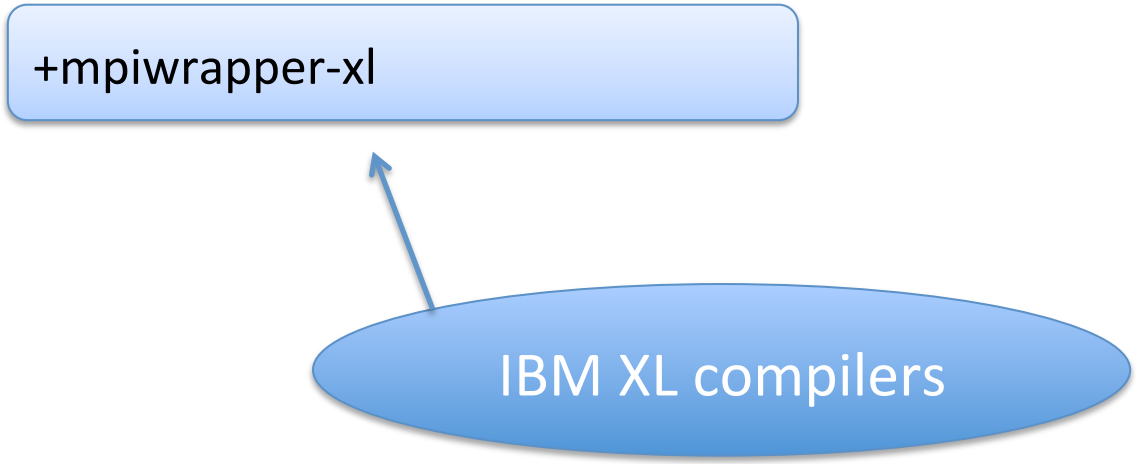
MPI on BG/Q

- Based on MPICH
- MPI-2.2
  - *Except* incompatible features (needing fork, e.g.)

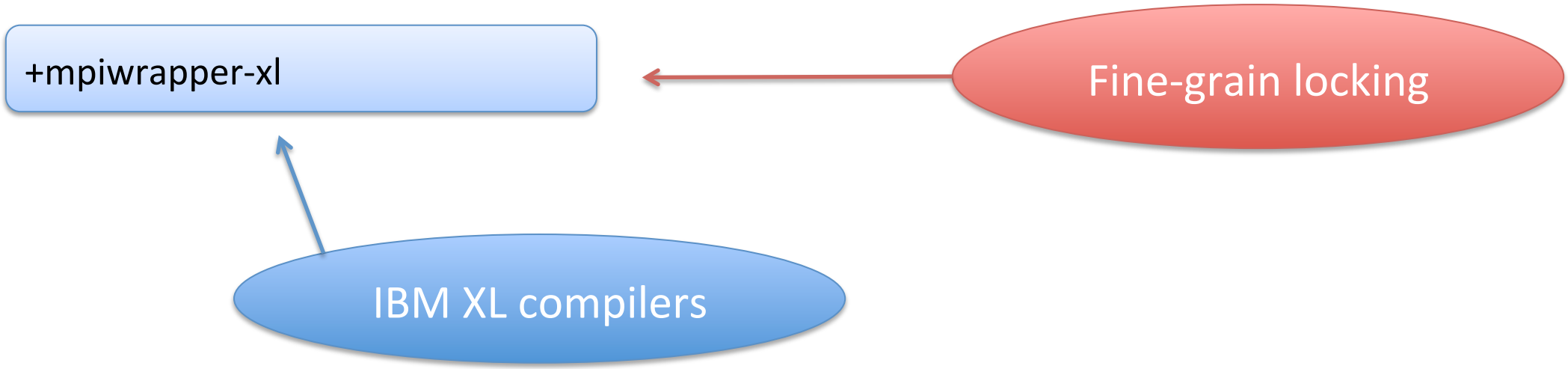
MPI on BG/Q

+mpiwrapper-xl

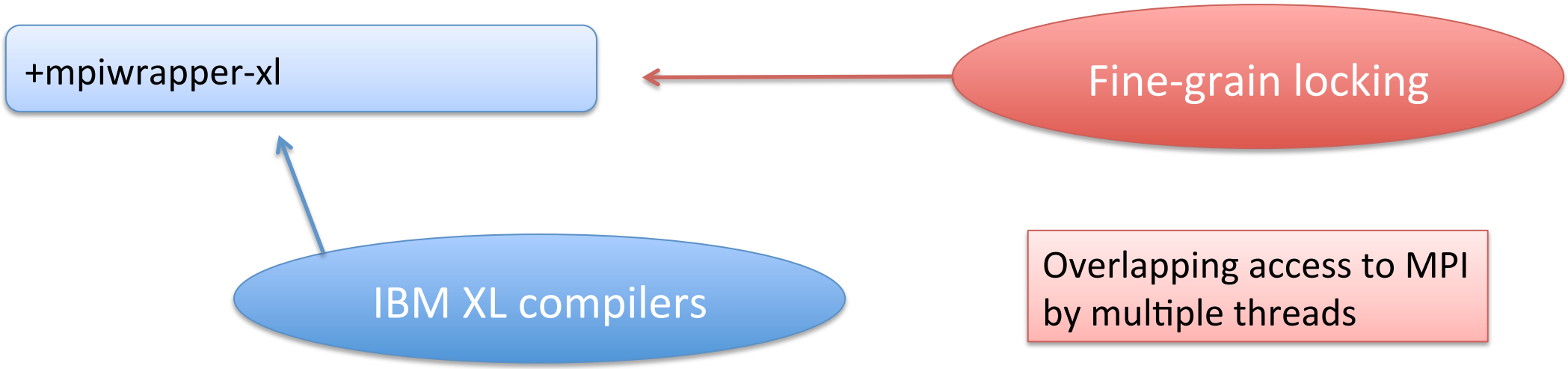
MPI on BG/Q



MPI on BG/Q



MPI on BG/Q

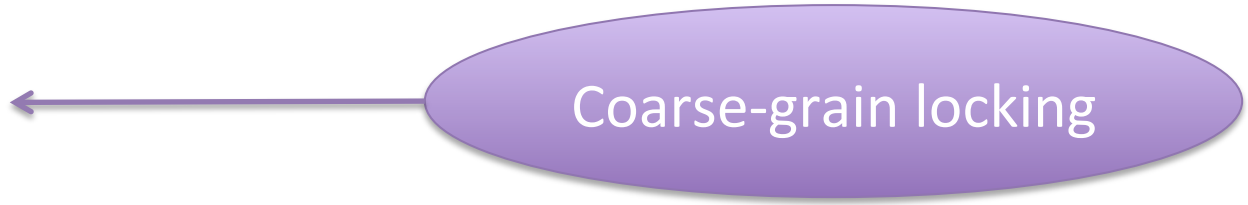

+mpiwrapper-xl
Fine-grain locking
IBM XL compilers
Overlapping access to MPI by multiple threads

MPI on BG/Q

+mpiwrapper-xl.legacy

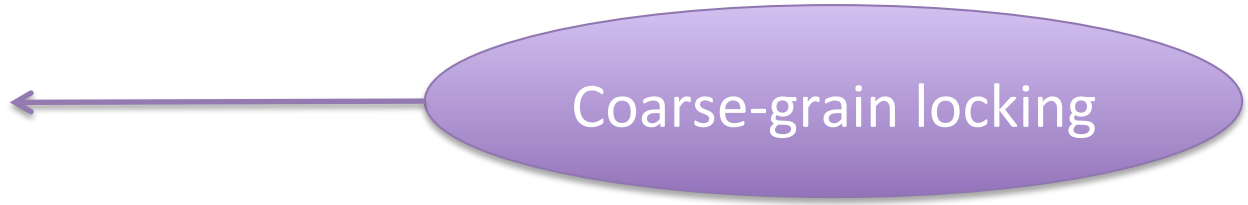
MPI on BG/Q

+mpiwrapper-xl.legacy



MPI on BG/Q

+mpiwrapper-xl.legacy



Mutual exclusion between threads at MPI function level

MPI on BG/Q

+mpiwrapper-xl.ndebug

+mpiwrapper-xl.legacy.ndebug

MPI on BG/Q

+mpiwrapper-gcc

+mpiwrapper-gcc.legacy

+mpiwrapper-gcc.ndebug

+mpiwrapper-gcc.legacy.ndebug

MPI-3

- No official plan for support on BG/Q
- Nonblocking collectives: *use PAMI*
- Remote Memory Access (RMA): *use PAMI*
- Other MPI-3 features:
  - MPI + MPIX + PAMI + SPI

PAMI

- Lots of PAMI goodness inside BG/Q MPI
- Why might you use PAMI directly?
  - Point-to-point communication faster than MPI
  - One-sided
  - RMA
  - Concurrent comm threads

PAMI Abstractions

- *Client*: exclusive communication channel
  - MPI client
  - PAMI-based library client
- *Context*: communication resources making independent progress
  - *Communication threads*: Multiple threads communicate concurrently on different contexts
  - *Endpoint*: communication address in a context
  - *Task*: process identifier within a client
- *Active messages*: Registered handler function called on remote node
- *Geometry*: group of tasks/endpoints used by collectives

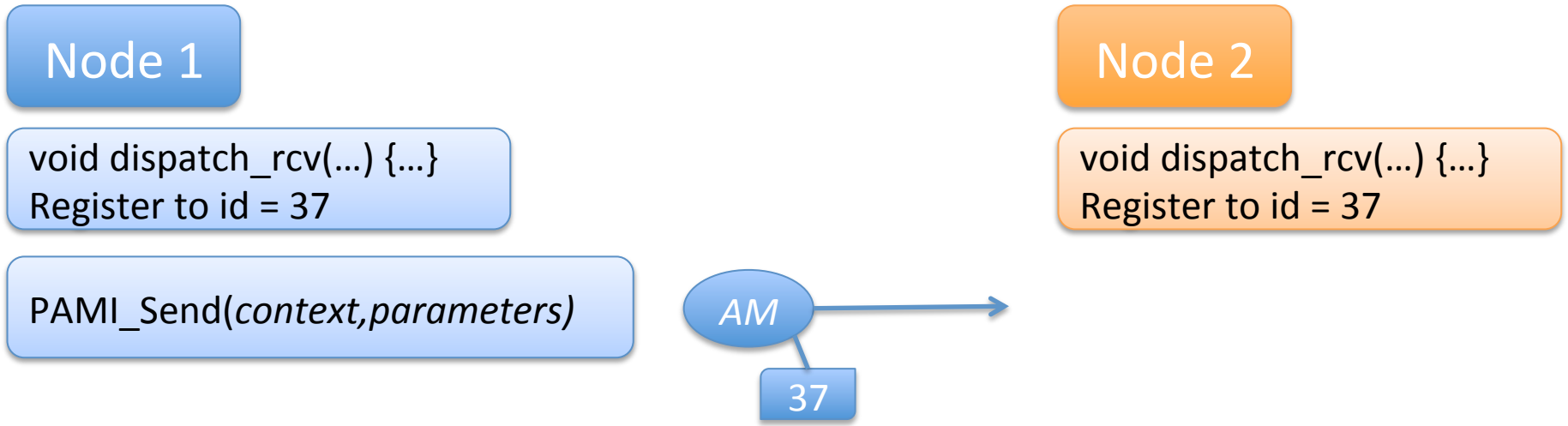
PAMI Abstractions

- *Client*: exclusive communication channel
  - MPI client
  - PAMI-based library client
- *Context*: communication resources making independent progress
  - *Communication threads*: Multiple threads communicate concurrently on different contexts
  - *Endpoint*: communication address in a context
  - *Task*: process identifier within a client
- *Active messages*: Registered handler function called on remote node
- *Communication threads*:
- *Geometry*: group of tasks/endpoints used by collectives

PAMI_Context_advance()

Returns after processed event or max poll iterates

Active Message



Active Message

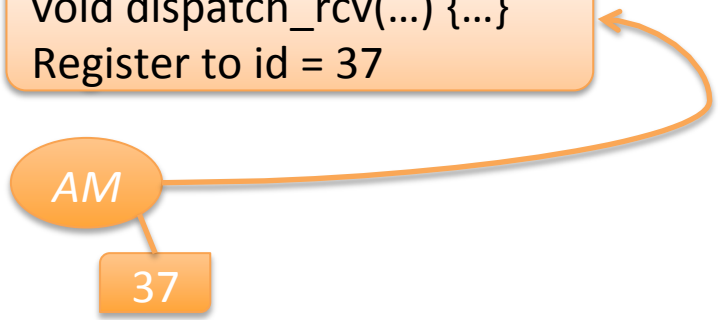
Node 1

void dispatch_rcv(…) {…}
Register to id = 37

PAMI_Send(*context,parameters)*

Node 2

void dispatch_rcv(…) {…}
Register to id = 37



PAMI Collectives

- Nonblocking collectives

Barrier
Broadcast
Reduce
AllReduce
Gather
Scatter
Allgather(v)
Alltoall(v)
....

- Active message collectives
  - Special dispatch functions
    - AMBroadcast
    - AMScatter
    - AMGather
    - AMReduce

PAMI Remote Memory Access

- PAMI_Rput(), PAMI_Rget()
- PAMI_Rput_typed(), PAMI_Rget_typed()
  - Non-contiguous
- PAMI_Rmw()
  - Atomic read-modify-write operations
    - Fetch & add
    - Compare & swap
    - ….

Use RDMA on BG/Q

PAMI Active Message Sends

- PAMI_Send_immediate()
- PAMI_Send()
- PAMI_Send_typed()
  - Non-contiguous

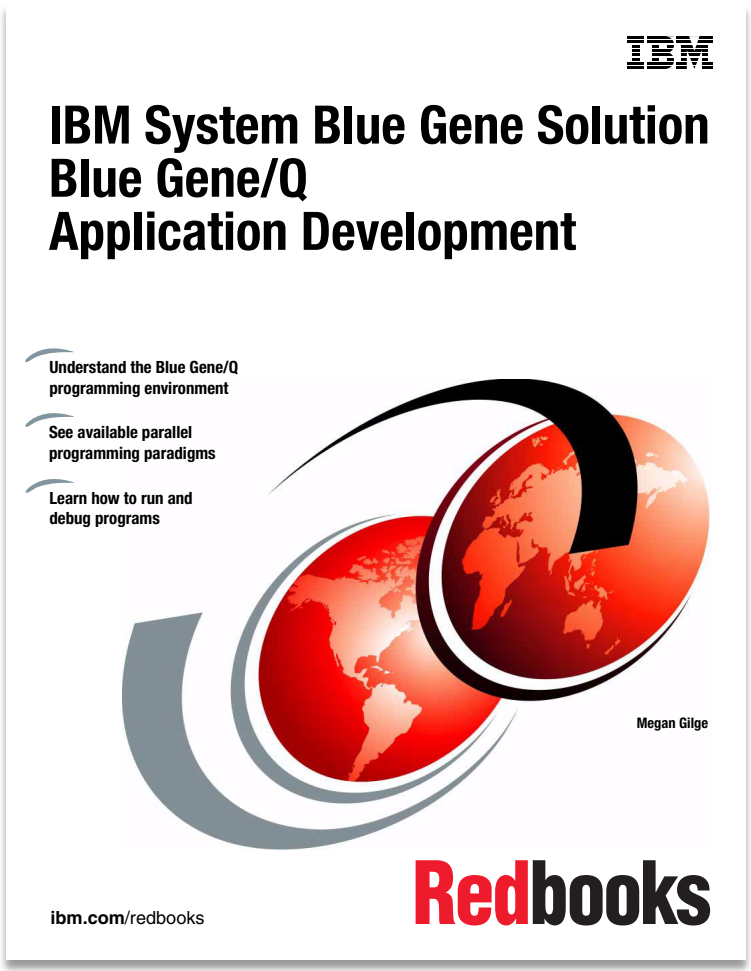
Other PAMI Features

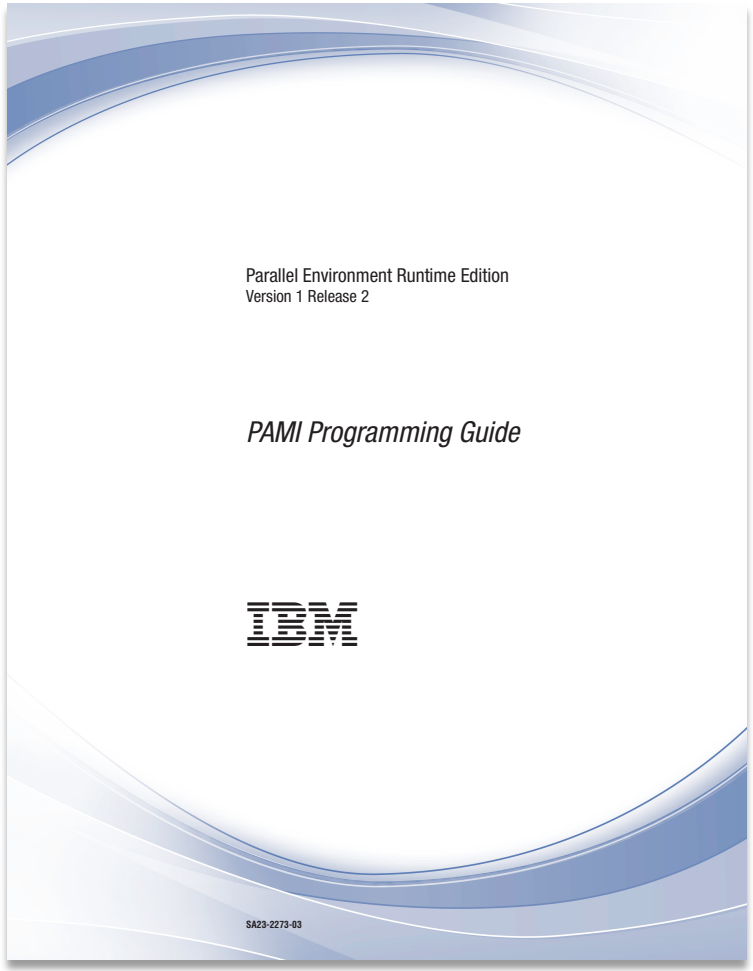
- Type system
  - PAMI_Type_create(), PAMI_Type_add_simple(), ….
- Fence
  - PAMI_Fence_begin(), PAMI_Fence_end(), ….

Environment Variables

- PAMID_VERBOSE = 1
  - PAMID_VERBOSE = 2 or 3: debug performance of collectives
- PAMID_THREAD_MULTIPLE = 1
  - Asynchronous progress on send-recv
  - One-sided communication
- Trouble with collectives (errors)? Try
  - PAMID_COLLECTIVES_SELECTION = 0
  - Then try PAMID_COLLECTIVES = 0
- Trouble with MPI-PAMI using too much memory?
  - PAMI_MEMORY_OPTIMIZED = 1
  - PAMID_DISABLE_INTERNAL_EAGER_TASK_LIMIT = 1 (esp. in c32 & c64 modes)

References





- PAMI Doxygen documentation
- /bgsys/drivers/ppcfloor/comm/sys/include/pami.h
- IPDS 2012 Talk (Sameer Kumar)
- OpenSHMEM 2013 talk (Alan Benner)
- Mysteries of the Deep (J. Hammond)
- pami-examples on Google Code

END