
TraceR Documentation

Nikhil Jain, Bilge Acun, Abhinav Bhatele

Apr 15, 2021

CONTENTS:

1	Download and Install	3
1.1	Dependencies	3
1.2	Build	3
2	User Guide	5
2.1	Quickstart	5
2.2	Creating a TraceR configuration file	5
2.3	Creating the network (CODES) configuration file	6
2.4	Creating the job placement file	7
2.5	Generating Traces	7
3	Tutorial	11
4	Source Code Documentation	13
4.1	Class Hierarchy	13
4.2	File Hierarchy	13
4.3	Full API	13
5	Indices and tables	51
	Index	53

TraceR is a trace replay tool built upon the ROSS-based CODES simulation framework. TraceR can be used for predicting network performance and understanding network behavior by simulating messaging in High Performance Computing applications on interconnection networks.

DOWNLOAD AND INSTALL

TraceR can be downloaded from [GitHub](#).

1.1 Dependencies

TraceR depends on [CODES](#) and [ROSS](#).

1.2 Build

There are several ways to build TraceR.

1. Use [spack](#) to build TraceR and its dependencies:

```
spack install tracer
```

2. Build TraceR and its dependencies manually:

- Download and install ROSS and CODES. Set the appropriate paths: ROSS_DIR, and CODES_DIR in tracer/Makefile.common.
- Pick between the two trace formats supported by TraceR: OTF2 or BigSim, and accordingly build the OTF2 or Charm++ library. If using OTF2 traces (default), set SELECT_TRACE = -DTRACER_OTF_TRACES=1, and ensure that otf2-config is in your PATH. If using BigSim traces, set SELECT_TRACE = -DTRACER_BIGSIM_TRACES=1, and set CHARMPATH to the Charm++ installation in tracer/Makefile.common.
- Set the ARCH variable in tracer/Makefile.common or alternatively set the CXX and ARCH_FLAGS variables. Then type:

```
cd tracer  
make
```

1.2.1 Trace Formats

TraceR supports two different trace formats as input. For each format, you need to build additional software as explained below.

1. Score-P's OTF2 format (default): To use OTF2 traces, you need to download and build the [OTF2](#) library.
2. AMPI-based BigSim format: To use BigSim traces as input to TraceR, you need to download and build [Charm++](#).

The instructions to build Charm++ are in the [Charm++ manual](#). You should use the “charm++” target and pass “bigemulator” as a build option.

USER GUIDE

Below, we provide detailed instructions for how to start doing network simulations using TraceR.

2.1 Quickstart

This is a basic `mpirun` command to launch a TraceR simulation in the optimistic mode:

```
mpirun -np <p> ../traceR --sync=3 -- <network_config> <tracer_config>
```

Some useful options to use with TraceR:

- | | |
|----------------------------|---|
| --sync | ROSS's PDES type. 1 - sequential, 2 - conservative, 3 - optimistic |
| --nkp | number of groups used for clustering LPs; recommended value for lower roll-backs: (total #LPs)/(#MPI processes) |
| --extramem | number of messages in ROSS's extra message buffer (each message is ~500 bytes, 100K should work for most cases) |
| --max-opt-lookahead | leash on optimistic execution in nanoseconds (1 microsecond is a good value) |
| --timer-frequency | frequency with which PEO should print current virtual time |

2.2 Creating a TraceR configuration file

This is the format for the TraceR config file:

```
<global map file>
<num jobs>
<Trace path for job0> <map file for job0> <number of ranks in job0> <iterations (use_
↪ 1 if running in normal mode)>
<Trace path for job1> <map file for job1> <number of ranks in job1> <iterations (use_
↪ 1 if running in normal mode)>
...
<Trace path for jobN> <map file for jobN> <number of ranks in jobN> <iterations (use_
↪ 1 if running in normal mode)>
```

If you do not intend to create global or per-job map files, you can use NA instead of them.

Sample TraceR config files can be found in `examples/jacobi2d-bigsim/tracer_config` (BigSim) or `examples/stencil4d-otf/tracer_config` (OTF)

See [Creating the job placement file](#) below for how to generate global or per-job map files.

2.3 Creating the network (CODES) configuration file

Sample network configuration files can be found in `examples/conf`

Additional documentation on the format of the CODES config file can be found in the CODES wiki at <https://xgitlab.cels.anl.gov/codes/codes/wikis/home>

A brief summary of the format follows.

LPGROUPS, MODELNET_GRP, PARAMS are keywords and should be used as is.

MODELNET_GRP:

```
repetition = number of routers that have nodes connecting to them.

server = number of MPI processes/cores per router

modelnet_* = number of NICs. For torus, this value has to be 1; for dragonfly,
it should be router radix divided by 4; for the fat-tree, it should be router
radix divided by 2. For the dragonfly network, modelnet_dragonfly_router should
also be specified (as 1). For express mesh, modelnet_express_mesh_router should
also be specified as 1.

Similarly, the fat-tree config file requires specifying fattree_switch which
can be 2 or 3, depending on the number of levels in the fat-tree. Note that the
total number of cores specified in the CODES config file can be greater than
the number of MPI processes being simulated (specified in the tracer config
file).
```

Other common parameters:

```
packet_size/chunk_size (both should have the same value) = size of the packets
created by NIC for transmission on the network. Smaller the packet size, longer
the time for which simulation will run (in real time). Larger the packet size,
the less accurate the predictions are expected to be (in virtual time). Packet
sizes of 512 bytes to 4096 bytes are commonly used.

modelnet_order = torus/dragonfly/fattree/slimfly/express_mesh

modelnet_scheduler =
    fcfs: packetize messages one by one.
    round-robin: packetize message in a round robin manner.

message_size = PDES parameter (keep constant at 512)

router_delay = delay at each router for packet transmission (in nanoseconds)

soft_delay = delay caused by software stack such as that of MPI (in nanoseconds)

link_bandwidth = bandwidth of each link in the system (in GB/s)

cn_bandwidth = bandwidth of connection between NIC and router (in GB/s)

buffer_size/vc_size = size of channels used to store transient packets at routers (in
bytes). Typical value is 64*packet_size.

routing = how are packets being routed. Options depend on the network.
    torus: static/adaptive
```

(continues on next page)

(continued from previous page)

```
dragonfly: minimal/nonminimal/adaptive
fat-tree: adaptive/static
```

Network specific parameters:

```
Torus:
  n_dims = number of dimensions in the torus
  dim_length = length of each dimension

Dragonfly:
  num_routers = number of routers within a group.
  global_bandwidth = bandwidth of the links that connect groups.

Fat-tree:
  ft_type = always choose 1
  num_levels = number of levels in the fat-tree (2 or 3)
  switch_radix = radix of the switch being used
  switch_count = number of switches at leaf level.
```

2.4 Creating the job placement file

See the README in utils for instructions on using the tools to generate the global and job mapping files.

2.5 Generating Traces

2.5.1 Score-P

Installation of Score-P

1. Download from <http://www.vi-hps.org/projects/score-p/>
2. `tar -xvzf scorep-3.0.tar.gz`
3. `cd scorep-3.0`
4. `CC=mpicc CFLAGS="-O2" CXX=mpicxx CXXFLAGS="-O2" FC=mpif77 ./configure --without-gui --prefix=<SCOREP_INSTALL>`
5. `make`
6. `make install`

Generating OTF2 traces with an MPI program using Score-P

Detailed instructions are available at <https://silc.zih.tu-dresden.de/scorep-current/pdf/scorep.pdf>.

1. Add \$SCOREP_INSTALL/bin to your PATH for convenience. Example:

```
export SCOREP_INSTALL=$HOME/workspace/scoreP/scorep-3.0/install
export PATH=$SCOREP_INSTALL/bin:$PATH
```

2. Add the following compile time flags to the application:

```
-I$SCOREP_INSTALL/include -I$SCOREP_INSTALL/include/scorep -DSCOREP_USER_ENABLE
```

3. Add `#include <scorep/SCOREP_User.h>` to all files where you plan to add any of the following Score-P calls (optional step):

```
SCOREP_RECORDING_OFF(); - stop recording
SCOREP_RECORDING_ON(); - start recording
```

Marking special regions: `SCOREP_USER_REGION_BY_NAME_BEGIN(regionname, SCOREP_USER_REGION_TYPE_COMMON)` and `SCOREP_USER_REGION_BY_NAME_END(regionname)`.

Region names beginning with `TRACER_WallTime_` are special: using `TRACER_WallTime_<any_name>` prints current time during simulation with tag `<any_name>`.

An example using these features is given below:

```
#include <scorep/SCOREP_User.h>
...
int main(int argc, char **argv, char **envp)
{
    MPI_Init(&argc,&argv);
    SCOREP_RECORDING_OFF(); //turn recording off for initialization/regions_
    ↪not of interest
    ...
    SCOREP_RECORDING_ON();
    //use verbatim to facilitate looping over the traces in simulation when_
    ↪simulating multiple jobs
    SCOREP_USER_REGION_BY_NAME_BEGIN("TRACER_Loop", SCOREP_USER_REGION_TYPE_
    ↪COMMON);
    // at least add this BEGIN timer call - called from only one rank
    // you can add more calls later with region names TRACER_WallTime_<any_
    ↪string of your choice>
    if(myRank == 0)
        SCOREP_USER_REGION_BY_NAME_BEGIN("TRACER_WallTime_MainLoop", SCOREP_USER_
    ↪REGION_TYPE_COMMON);
    // Application main work LOOP
    for ( int itscf = 0; itscf < nitscf; itscf++ )
    {
        ...
    }
    // time call to mark END of work - called from only one rank
    if(myRank == 0)
        SCOREP_USER_REGION_BY_NAME_END("TRACER_WallTime_MainLoop");
    // use verbatim - mark end of trace loop
    SCOREP_USER_REGION_BY_NAME_END("TRACER_Loop");
    SCOREP_RECORDING_OFF(); //turn off recording again
    ...
}
```

4. For the link step, prefix the linker line with the following:

```
LD = scorep --user --nocompiler --noopenmp --nopomp --nocuda --noopenacc --
    ↪noopencl --nomemory <your_linker>
```

5. For running, set:

```
export SCOREP_ENABLE_TRACING=1
export SCOREP_ENABLE_PROFILING=0
```

(continues on next page)

(continued from previous page)

```
export SCOREP_REDUCE_PROBE_TEST=1
export SCOREP_MPI_ENABLE_GROUPS=ENV, P2P, COLL, XNONBLOCK
```

If Score-P prints a warning about flushing traces during the run, you may avoid them using:

```
export SCOREP_TOTAL_MEMORY=256M
export SCOREP_EXPERIMENT_DIRECTORY=/p/lscratchd/<username>/...
```

6. Run the binary and traces should be generated in a folder named scorep-.*.

2.5.2 BigSim

Installation of BigSim

Compile BigSim/Charm++ for emulation (see <http://charm.cs.illinois.edu/manuals/html/bigsim/manual-1p.html> for more detail). Use any one of the following commands:

- To use UDP as BigSim/Charm++'s communication layer:

```
./build bgampi net-linux-x86_64 bigemulator --with-production --enable-tracing
./build bgampi net-darwin-x86_64 bigemulator --with-production --enable-tracing
```

Or explicitly provide the compiler optimization level:

```
./build bgampi net-linux-x86_64 bigemulator -O2
```

- To use MPI as BigSim/Charm++'s communication layer:

```
./build bgampi mpi-linux-x86_64 bigemulator --with-production --enable-tracing
```

Note: This build is used to compile MPI applications so that traces can be generated. Hence, the communication layer used by BigSim/Charm++ is not important. During simulation, the communication will be replayed using the network simulator from CODES. However, the computation time captured here can be important if it is not being explicitly replaced at simulation time using configuration options. So using appropriate compiler flags is important.

Generating AMPI traces with an MPI program using BigSim

1. Compile your MPI application using BigSim/Charm++.

Example commands:

```
$CHARM_DIR/bin/ampicc -O2 simplePrg.c -o simplePrg_c
$CHARM_DIR/bin/ampiCC -O2 simplePrg.cc -o simplePrg_cxx
```

2. Emulation to generate traces. When the binary generated is run, BigSim/Charm++ runs the program on the allocated cores as if it were running as usual. Users should provide a few additional arguments to specify the number of MPI processes in the prototype systems.

If using UDP as the BigSim/Charm++'s communication layer:

```
./charmrun +p<number of real processes> ++nodelist <machine file> ./pgm
↪<program arguments> +vp<number of processes expected on the future system>
↪+x<x dim> +y<y dim> +z<z dim> +bglog
```

If using MPI as the BigSim/Charm++'s communication layer:

```
mpirun -n<number of real processes> ./pgm <program arguments> +vp<number of  
↪processes expected on the future system> +x<x dim> +y<y dim> +z<z dim> ↪  
↪+bglog
```

Number of real processes is typically equal to the number cores the emulation is being run on.

machine file is the list of systems the emulation should be run on (similar to machine file for MPI; refer to Charm++ website for more details).

vp is the number of MPI ranks that are to be emulated. For simple tests, it can be the same as the number of real processes, in which case one MPI rank is run on each real process (as it happens when a regular program is run). When the number of *vp* (virtual processes) is higher, BigSim launches user level threads to execute multiple MPI ranks within a process.

+x +y +z defines a 3D grid of the virtual processes. The product of these three dimensions must match the number of *vp*'s. These arguments do not have any effect on the emulation, but exist due to historical reasons.

+bglog instructs bigsim to write the logs to files.

3. When this run is finished, you should see many files named *bgTrace** in the directory. The total number of such files equals the number of real processes plus one. Their names are *bgTrace*, *bgTrace0*, *bgTrace1*, and so on. Create a new folder and move all *bgTrace* files to that folder.

CHAPTER
THREE

TUTORIAL

SOURCE CODE DOCUMENTATION

4.1 Class Hierarchy

4.2 File Hierarchy

4.3 Full API

4.3.1 Classes and Structs

Struct Coll_lookup

- Defined in file_tracer_tracer-driver.h

Struct Documentation

`struct Coll_lookup`

Public Members

proc_event **remote_event**

proc_event **local_event**

Struct CoreInf

- Defined in file_tracer_tracer-driver.h

Struct Documentation

struct CoreInf

Public Members

int **mapsTo**
int **jobID**

Struct JobInf

- Defined in file_tracer_reader_datatypes.h

Struct Documentation

struct JobInf

Public Members

int **numRanks**
char **traceDir**[256]
char **map_file**[256]
int ***rankMap**
int ***offsets**
int **skipMsgId**
int **numIters**

Struct MsgEntry

- Defined in file_tracer_elements_MsgEntry.h

Struct Documentation

struct MsgEntry

Public Members

int **node**
int **thread**
MsgID **msgId**

Struct MsgID

- Defined in file_tracer_elements_MsgEntry.h

Struct Documentation

struct **MsgID**

Public Members

int **pe**
int **id**
uint64_t **size**

Struct proc_msg

- Defined in file_tracer_tracer-driver.h

Struct Documentation

struct **proc_msg**

Public Members

enum *proc_event* **proc_event_type**
tw_lpid **src**
int **iteration**
TaskPair **executed**
int **fwd_dep_count**
int **saved_task**
MsgID **msgId**
bool **incremented_flag**
int **model_net_calls**
unsigned int **coll_info**
unsigned int **coll_info_2**

Struct `proc_state`

- Defined in `file_tracer_tracer-driver.h`

Struct Documentation

struct `proc_state`

Public Members

`tw_stime` **start_ts**

`tw_stime` **end_ts**

PE ***my_pe**

`clock_t` **sim_start**

`int` **my_pe_num**

`int` **my_job**

Struct `TaskPair`

- Defined in `file_tracer_reader_datatypes.h`

Struct Documentation

struct `TaskPair`

Public Members

`int` **iter**

`int` **taskid**

Class `CollMsgKey`

- Defined in `file_tracer_elements_PE.h`

Class Documentation

class `CollMsgKey`

Public Functions

```
inline CollMsgKey (uint32_t _rank, uint32_t _comm, int64_t _seq)
inline bool operator< (const CollMsgKey &rhs) const
inline ~CollMsgKey ()
```

Public Members

```
uint32_t rank
uint32_t comm
int64_t seq
```

Class MsgKey

- Defined in file_tracer_elements_PE.h

Class Documentation

class MsgKey

Public Functions

```
inline MsgKey (uint32_t _rank, uint32_t _tag, uint32_t _comm, int64_t _seq)
inline bool operator< (const MsgKey &rhs) const
inline ~MsgKey ()
```

Public Members

```
uint32_t rank
uint32_t comm
uint32_t tag
int64_t seq
```

Class PE

- Defined in file_tracer_elements_PE.h

Class Documentation

class PE

Public Functions

```
PE ()  
~PE ()  
void goToNextIter (int iter)  
bool noUnsatDep (int iter, int tInd)  
void mark_all_done (int iter, int tInd)  
double taskExecTime (int tInd)  
void printStats (int iter)  
void invertMsgPe (int iter, int tInd)  
double getTaskExecTime (int tInd)  
void addTaskExecTime (int tInd, double time)  
int findTaskFromMsg (MsgID *msg)
```

Public Members

```
std::list<TaskPair> msgBuffer  
Task *myTasks  
bool **taskStatus  
bool **taskExecuted  
bool **msgStatus  
bool *allMarked  
double currTime  
bool busy  
int beforeTask  
int totalTasksCount  
int myNum  
int myEmPE  
int jobNum  
int tasksCount  
int currentTask  
int firstTask  
int currIter  
int loop_start_task
```

```

std::map<int, int> *msgDestLogs
int numWth
int numEmpes
KeyType pendingMsgs
KeyType pendingRMsgs
int64_t *sendSeq
int64_t *recvSeq
std::map<int, int> pendingReqs
std::map<int, int64_t> pendingRReqs
std::vector<int64_t> collectiveSeq
std::map<int64_t, std::map<int64_t, std::map<int, int>>> pendingCollMsgs
CollKeyType pendingRCollMsgs
int64_t currentCollComm
int64_t currentCollSeq
int64_t currentCollTask
int64_t currentCollMsgSize
int currentCollRank
int currentCollPartner
int currentCollSize
int currentCollSendCount
int currentCollRecvCount

```

Class Task

- Defined in file_tracer_elements_Task.h

Class Documentation

class Task

Public Functions

Task ()

~Task ()

Public Members

bool **endEvent**
bool **loopEvent**
bool **loopStartEvent**
double **execTime**

Class TraceReader

- Defined in file_tracer_reader_TraceReader.h

Class Documentation

class TraceReader

Public Functions

TraceReader (char*)
~TraceReader ()

Public Members

int **numEmPes**
int **totalWorkerProcs**
int **totalNodes**
int **numWth**
int ***allNodeOffsets**
char **tracePath**[256]
int **fileLoc**
int **firstLog**
int **totalTlineLength**

4.3.2 Enums

Enum proc_event

- Defined in file_tracer_tracer-driver.h

Enum Documentation

`enum proc_event`

Values:

```
enumerator KICKOFF
enumerator LOCAL
enumerator RECV_MSG
enumerator BCAST
enumerator EXEC_COMPLETE
enumerator SEND_COMP
enumerator RECV_POST
enumerator COLL_BCAST
enumerator COLL_REDUCTION
enumerator COLL_A2A
enumerator COLL_A2A_SEND_DONE
enumerator COLL_ALLGATHER
enumerator COLL_ALLGATHER_SEND_DONE
enumerator COLL_BRUCK
enumerator COLL_BRUCK_SEND_DONE
enumerator COLL_A2A_BLOCKED
enumerator COLL_A2A_BLOCKED_SEND_DONE
enumerator COLL_SCATTER_SMALL
enumerator COLL_SCATTER
enumerator COLL_SCATTER_SEND_DONE
enumerator RECV_COLL_POST
enumerator COLL_COMPLETE
```

Enum tracer_coll_type

- Defined in file_tracer_tracer-driver.h

Enum Documentation

`enum tracer_coll_type`

Values:

```
enumerator TRACER_COLLECTIVE_BCAST
enumerator TRACER_COLLECTIVE_REDUCE
enumerator TRACER_COLLECTIVE_BARRIER
enumerator TRACER_COLLECTIVE_ALLTOALL_LARGE
```

```
enumerator TRACER_COLLECTIVE_ALLTOALL_BLOCKED
enumerator TRACER_COLLECTIVE_ALL_BRUCK
enumerator TRACER_COLLECTIVE_ALLGATHER_LARGE
enumerator TRACER_COLLECTIVE_SCATTER_SMALL
enumerator TRACER_COLLECTIVE_SCATTER
```

4.3.3 Functions

Function addEventSub

- Defined in file_tracer_reader_CWrapper.h

Function Documentation

void **addEventSub** (int *job*, char **key*, double *val*, int *numjobs*)

Function addMsgSizeSub

- Defined in file_tracer_reader_CWrapper.h

Function Documentation

void **addMsgSizeSub** (int *job*, int64_t *key*, int64_t *val*, int *numjobs*)

Function bcast_msg

- Defined in file_tracer_tracer-driver.h

Function Documentation

int **bcast_msg** (*proc_state* **ns*, int *size*, int *iter*, *MsgID* **msgId*, tw_stime *timeOffset*, tw_stime *copyTime*,
tw_lp **lp*, *proc_msg* **m*)

Function delegate_send_msg

- Defined in file_tracer_tracer-driver.h

Function Documentation

void **delegate_send_msg** (*proc_state* *ns, tw_lp *lp, *proc_msg* *m, tw_bf *b, *Task* *t, int taskid, tw_stime delay)

Function enqueue_msg

- Defined in file_tracer_tracer-driver.h

Function Documentation

void **enqueue_msg** (*proc_state* *ns, int size, int iter, *MsgID* *msgId, int64_t seq, int dest_id, tw_stime sendOffset, *enum proc_event* evt_type, *proc_msg* *m_local, tw_lp *lp)

Function exec_comp

- Defined in file_tracer_tracer-driver.h

Function Documentation

int **exec_comp** (*proc_state* *ns, int iter, int task_id, int comm_id, tw_stime sendOffset, int recv, tw_lp *lp)

Function exec_task

- Defined in file_tracer_tracer-driver.h

Function Documentation

tw_stime **exec_task** (*proc_state* *ns, *TaskPair* task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b)

Function exec_task_rev

- Defined in file_tracer_tracer-driver.h

Function Documentation

void **exec_task_rev** (*proc_state* *ns, *TaskPair* task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b)

Function `handle_a2a_blocked_send_comp_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_a2a_blocked_send_comp_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_a2a_blocked_send_comp_rev_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_a2a_blocked_send_comp_rev_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_a2a_send_comp_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_a2a_send_comp_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_a2a_send_comp_rev_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_a2a_send_comp_rev_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_allgather_send_comp_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_allgather_send_comp_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_allgather_send_comp_rev_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_allgather_send_comp_rev_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_bcast_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_bcast_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_bcast_rev_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_bcast_rev_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_bruck_send_comp_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_bruck_send_comp_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_bruck_send_comp_rev_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_bruck_send_comp_rev_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_coll_complete_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_coll_complete_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_coll_complete_rev_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_coll_complete_rev_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_coll_recv_post_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_coll_recv_post_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_coll_recv_post_rev_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_coll_recv_post_rev_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_exec_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_exec_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_exec_rev_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_exec_rev_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_kickoff_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_kickoff_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_kickoff_rev_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_kickoff_rev_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_local_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_local_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_local_rev_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_local_rev_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_recv_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **handle_recv_event** (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_recv_post_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **handle_recv_post_event** (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_recv_post_rev_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **handle_recv_post_rev_event** (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_recv_rev_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **handle_recv_rev_event** (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_scatter_send_comp_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **handle_scatter_send_comp_event** (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_scatter_send_comp_rev_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_scatter_send_comp_rev_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_send_comp_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_send_comp_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `handle_send_comp_rev_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void `handle_send_comp_rev_event` (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `isPEonThisRank`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

bool `isPEonThisRank` (int *jobID*, int *i*)

Function `lpid_to_job`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

int `lpid_to_job` (int *lp_gid*)

Function `lpid_to_pe`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

int **lpid_to_pe** (int *lp_gid*)

Function `MsgEntry_getID`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int **MsgEntry_getID** (*MsgEntry* **m*)

Function `MsgEntry_getNode`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int **MsgEntry_getNode** (*MsgEntry* **m*)

Function `MsgEntry_getPE`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int **MsgEntry_getPE** (*MsgEntry* **m*)

Function `MsgEntry_getSize`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int **MsgEntry_getSize** (*MsgEntry* **m*)

Function `MsgEntry_getThread`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int **MsgEntry_getThread** (*MsgEntry* **m*)

Function `MsgID_getID`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int **MsgID_getID** (*MsgID* **m*)

Function `MsgID_getPE`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int **MsgID_getPE** (*MsgID* **m*)

Function `MsgID_getSize`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int **MsgID_getSize** (*MsgID* **m*)

Function `newMsgEntry`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

MsgEntry ***newMsgEntry** ()

Function newMsgID

- Defined in file_tracer_reader_CWrapper.h

Function Documentation

MsgID ***newMsgID** (int *size*, int *pe*, int *id*)

Function ns_to_s

- Defined in file_tracer_tracer-driver.h

Function Documentation

tw_stime **ns_to_s** (tw_stime *ns*)

Function PE_addTaskExecTime

- Defined in file_tracer_reader_CWrapper.h

Function Documentation

void **PE_addTaskExecTime** (*PE* **p*, int *tInd*, double *time*)

Function PE_addToBuffer

- Defined in file_tracer_reader_CWrapper.h

Function Documentation

void **PE_addToBuffer** (*PE* **p*, *TaskPair* **task_id*)

Function PE_addToFrontBuffer

- Defined in file_tracer_reader_CWrapper.h

Function Documentation

void **PE_addToFrontBuffer** (*PE* **p*, *TaskPair* **task_id*)

Function `PE_clearMsgBuffer`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

void **PE_clearMsgBuffer** (*PE* **p*)

Function `PE_dec_iter`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

void **PE_dec_iter** (*PE* **p*)

Function `PE_findTaskFromMsg`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int **PE_findTaskFromMsg** (*PE* **p*, *MsgID* **msgId*)

Function `PE_get_currentTask`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int **PE_get_currentTask** (*PE* **p*)

Function `PE_get_iter`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int **PE_get_iter** (*PE* **p*)

Function `PE_get_myEmPE`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int `PE_get_myEmPE` (*PE* **p*)

Function `PE_get_myNum`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int `PE_get_myNum` (*PE* **p*)

Function `PE_get_numWorkThreads`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int `PE_get_numWorkThreads` (*PE* **p*)

Function `PE_get_taskDone`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

bool `PE_get_taskDone` (*PE* **p*, int, int *tInd*)

Function `PE_get_tasksCount`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int `PE_get_tasksCount` (*PE* **p*)

Function `PE_get_totalTasksCount`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int **PE_get_totalTasksCount** (*PE* **p*)

Function `PE_getBufferSize`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int **PE_getBufferSize** (*PE* **p*)

Function `PE_getFirstTask`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

int **PE_getFirstTask** (*PE* **p*)

Function `PE_getNextBuffedMsg`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

TaskPair **PE_getNextBuffedMsg** (*PE* **p*)

Function `PE_getTaskExecTime`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

double **PE_getTaskExecTime** (*PE* **p*, int *tInd*)

Function `PE_inc_iter`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

void **PE_inc_iter** (*PE* **p*)

Function `PE_invertMsgPe`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

void **PE_invertMsgPe** (*PE* **p*, int, int *tInd*)

Function `PE_is_busy`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

bool **PE_is_busy** (*PE* **p*)

Function `PE_isEndEvent`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

bool **PE_isEndEvent** (*PE* **p*, int *task_id*)

Function `PE_isLoopEvent`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

bool **PE_isLoopEvent** (*PE* **p*, int *task_id*)

Function PE_mark_all_done

- Defined in file_tracer_reader_CWrapper.h

Function Documentation

void **PE_mark_all_done** (*PE* **p*, int *iter*, int *task_id*)

Function PE_noMsgDep

- Defined in file_tracer_reader_CWrapper.h

Function Documentation

bool **PE_noMsgDep** (*PE* **p*, int, int *tInd*)

Function PE_noUnsatDep

- Defined in file_tracer_reader_CWrapper.h

Function Documentation

bool **PE_noUnsatDep** (*PE* **p*, int, int *tInd*)

Function PE_printStat

- Defined in file_tracer_reader_CWrapper.h

Function Documentation

void **PE_printStat** (*PE* **p*, int *iter*)

Function PE_removeFromBuffer

- Defined in file_tracer_reader_CWrapper.h

Function Documentation

void **PE_removeFromBuffer** (*PE* **p*, *TaskPair* **task_id*)

Function `PE_resizeBuffer`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

void **PE_resizeBuffer** (*PE* **p*, int *num_elems_to_remove*)

Function `PE_set_busy`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

void **PE_set_busy** (*PE* **p*, bool *b*)

Function `PE_set_currentTask`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

void **PE_set_currentTask** (*PE* **p*, int *tInd*)

Function `PE_set_taskDone`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

void **PE_set_taskDone** (*PE* **p*, int, int *tInd*, bool *b*)

Function `pe_to_job`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

int **pe_to_job** (int *pe*)

Function `pe_to_lpid`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

int **pe_to_lpid** (int *pe*, int *job*)

Function `perform_a2a`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **perform_a2a** (*proc_state* **ns*, int *task_id*, tw_lp **lp*, *proc_msg* **m*, tw_bf **b*, int *isEvent*)

Function `perform_a2a_blocked`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **perform_a2a_blocked** (*proc_state* **ns*, int *task_id*, tw_lp **lp*, *proc_msg* **m*, tw_bf **b*, int *isEvent*)

Function `perform_a2a_blocked_rev`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **perform_a2a_blocked_rev** (*proc_state* **ns*, int *task_id*, tw_lp **lp*, *proc_msg* **m*, tw_bf **b*, int *isEvent*)

Function `perform_a2a_rev`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **perform_a2a_rev** (*proc_state* **ns*, int *task_id*, tw_lp **lp*, *proc_msg* **m*, tw_bf **b*, int *isEvent*)

Function `perform_allgather`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **perform_allgather** (*proc_state* *ns, int task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b, int isEvent)

Function `perform_allgather_rev`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **perform_allgather_rev** (*proc_state* *ns, int task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b, int isEvent)

Function `perform_allreduce`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **perform_allreduce** (*proc_state* *ns, int task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b, int isEvent)

Function `perform_allreduce_rev`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **perform_allreduce_rev** (*proc_state* *ns, int task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b, int isEvent)

Function `perform_bcast`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **perform_bcast** (*proc_state* *ns, int task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b, int isEvent)

Function perform_bcast_rev

- Defined in file_tracer_tracer-driver.h

Function Documentation

void **perform_bcast_rev** (*proc_state* *ns, int task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b, int isEvent)

Function perform_bruck

- Defined in file_tracer_tracer-driver.h

Function Documentation

void **perform_bruck** (*proc_state* *ns, int task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b, int isEvent)

Function perform_bruck_rev

- Defined in file_tracer_tracer-driver.h

Function Documentation

void **perform_bruck_rev** (*proc_state* *ns, int task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b, int isEvent)

Function perform_collective

- Defined in file_tracer_tracer-driver.h

Function Documentation

void **perform_collective** (*proc_state* *ns, int task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b)

Function perform_collective_rev

- Defined in file_tracer_tracer-driver.h

Function Documentation

void **perform_collective_rev** (*proc_state* *ns, int task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b)

Function perform_reduction

- Defined in file_tracer_tracer-driver.h

Function Documentation

void **perform_reduction** (*proc_state* *ns, int task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b, int isEvent)

Function perform_reduction_rev

- Defined in file_tracer_tracer-driver.h

Function Documentation

void **perform_reduction_rev** (*proc_state* *ns, int task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b, int isEvent)

Function perform_scatter

- Defined in file_tracer_tracer-driver.h

Function Documentation

void **perform_scatter** (*proc_state* *ns, int task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b, int isEvent)

Function perform_scatter_rev

- Defined in file_tracer_tracer-driver.h

Function Documentation

void **perform_scatter_rev** (*proc_state* *ns, int task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b, int isEvent)

Function `perform_scatter_small`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **perform_scatter_small** (*proc_state* *ns, int task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b, int isEvent)

Function `perform_scatter_small_rev`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **perform_scatter_small_rev** (*proc_state* *ns, int task_id, tw_lp *lp, *proc_msg* *m, tw_bf *b, int isEvent)

Function `proc_commit_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **proc_commit_event** (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `proc_event`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **proc_event** (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function `proc_finalize`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

void **proc_finalize** (*proc_state* *ns, tw_lp *lp)

Function proc_init

- Defined in file_tracer_tracer-driver.h

Function Documentation

void **proc_init** (*proc_state* *ns, tw_lp *lp)

Function proc_rev_event

- Defined in file_tracer_tracer-driver.h

Function Documentation

void **proc_rev_event** (*proc_state* *ns, tw_bf *b, *proc_msg* *m, tw_lp *lp)

Function s_to_ns

- Defined in file_tracer_tracer-driver.h

Function Documentation

tw_stime **s_to_ns** (tw_stime ns)

Function send_coll_comp

- Defined in file_tracer_tracer-driver.h

Function Documentation

int **send_coll_comp** (*proc_state* *ns, tw_stime sendOffset, int collType, tw_lp *lp, int isEvent, *proc_msg* *m)

Function `send_coll_comp_rev`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

int **send_coll_comp_rev** (*proc_state* *ns, tw_stime *sendOffset*, int *collType*, tw_lp *lp, int *isEvent*,
proc_msg *m)

Function `send_msg`

- Defined in `file_tracer_tracer-driver.h`

Function Documentation

int **send_msg** (*proc_state* *ns, int *size*, int *iter*, *MsgID* *msgId, int64_t *seq*, int *dest_id*, tw_stime *timeOffset*,
enum proc_event *evt_type*, tw_lp *lp, bool *fillSz* = false, int64_t *size2* = 0)

Function `TraceReader_readOTF2Trace`

- Defined in `file_tracer_reader_CWrapper.h`

Function Documentation

void **TraceReader_readOTF2Trace** (*PE* *pe, int *my_pe_num*, int *my_job*, double **startTime*)

4.3.4 Variables

Variable `copy_per_byte`

- Defined in `file_tracer_tracer-driver.h`

Variable Documentation

double **copy_per_byte**

Variable `eager_limit`

- Defined in `file_tracer_tracer-driver.h`

Variable Documentation

double **eager_limit**

Variable jobs

- Defined in file_tracer_tracer-driver.h

Variable Documentation

JobInf ***jobs**

Variable net_id

- Defined in file_tracer_tracer-driver.h

Variable Documentation

int **net_id**

Variable nic_delay

- Defined in file_tracer_tracer-driver.h

Variable Documentation

tw_stime **nic_delay**

Variable print_frequency

- Defined in file_tracer_tracer-driver.h

Variable Documentation

unsigned int **print_frequency**

Variable rdma_delay

- Defined in file_tracer_tracer-driver.h

Variable Documentation

tw_stime **rdma_delay**

Variable **soft_delay_mpi**

- Defined in file_tracer_tracer-driver.h

Variable Documentation

tw_stime **soft_delay_mpi**

4.3.5 Defines

Define **BCAST_DEGREE**

- Defined in file_tracer_tracer-driver.h

Define Documentation

BCAST_DEGREE

Define **MPI_INTERNAL_DELAY**

- Defined in file_tracer_tracer-driver.h

Define Documentation

MPI_INTERNAL_DELAY

Define **REDUCE_DEGREE**

- Defined in file_tracer_tracer-driver.h

Define Documentation

REDUCE_DEGREE

Define `TIME_MULT`

- Defined in file_tracer_elements_Task.h

Define Documentation

`TIME_MULT`

Define `TRACER_A2A_ALG_CUTOFF`

- Defined in file_tracer_tracer-driver.h

Define Documentation

`TRACER_A2A_ALG_CUTOFF`

Define `TRACER_ALLGATHER_ALG_CUTOFF`

- Defined in file_tracer_tracer-driver.h

Define Documentation

`TRACER_ALLGATHER_ALG_CUTOFF`

Define `TRACER_BLOCK_SIZE`

- Defined in file_tracer_tracer-driver.h

Define Documentation

`TRACER_BLOCK_SIZE`

Define `TRACER_SCATTER_ALG_CUTOFF`

- Defined in file_tracer_tracer-driver.h

Define Documentation

`TRACER_SCATTER_ALG_CUTOFF`

4.3.6 Typedefs

Typedef CollKeyType

- Defined in file_tracer_elements_PE.h

Typedef Documentation

```
typedef std::map<CollMsgKey, std::list<int>> CollKeyType
```

Typedef CoreInf

- Defined in file_tracer_tracer-driver.h

Typedef Documentation

```
typedef struct CoreInf CoreInf
```

Typedef JobInf

- Defined in file_tracer_reader_datatypes.h

Typedef Documentation

```
typedef struct JobInf JobInf
```

Typedef KeyType

- Defined in file_tracer_elements_PE.h

Typedef Documentation

```
typedef std::map<MsgKey, std::list<int>> KeyType
```

Typedef MsgEntry

- Defined in file_tracer_reader_CWrapper.h

Typedef Documentation

`typedef struct MsgEntry MsgEntry`

Typedef **MsgID**

- Defined in file_tracer_reader_CWrapper.h

Typedef Documentation

`typedef struct MsgID MsgID`

Typedef **PE**

- Defined in file_tracer_reader_CWrapper.h

Typedef Documentation

`typedef struct PE PE`

Typedef **TaskPair**

- Defined in file_tracer_reader_datatypes.h

Typedef Documentation

`typedef struct TaskPair TaskPair`

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

A

addEventSub (C++ function), 22
addMsgSizeSub (C++ function), 22

B

BCAST_DEGREE (C macro), 47
bcast_msg (C++ function), 22

C

Coll_lookup (C++ struct), 13
Coll_lookup::local_event (C++ member), 13
Coll_lookup::remote_event (C++ member), 13
CollKeyType (C++ type), 49
CollMsgKey (C++ class), 16
CollMsgKey::~CollMsgKey (C++ function), 17
CollMsgKey::CollMsgKey (C++ function), 17
CollMsgKey::comm (C++ member), 17
CollMsgKey::operator< (C++ function), 17
CollMsgKey::rank (C++ member), 17
CollMsgKey::seq (C++ member), 17
copy_per_byte (C++ member), 45
CoreInf (C++ struct), 14
CoreInf (C++ type), 49
CoreInf::jobID (C++ member), 14
CoreInf::mapsTo (C++ member), 14

D

delegate_send_msg (C++ function), 23

E

eager_limit (C++ member), 46
enqueue_msg (C++ function), 23
exec_comp (C++ function), 23
exec_task (C++ function), 23
exec_task_rev (C++ function), 23

H

handle_a2a_blocked_send_comp_event (C++ function), 24
handle_a2a_blocked_send_comp_rev_event (C++ function), 24

handle_a2a_send_comp_event (C++ function), 24
handle_a2a_send_comp_rev_event (C++ function), 24
handle_allgather_send_comp_event (C++ function), 24
handle_allgather_send_comp_rev_event (C++ function), 25
handle_bcast_event (C++ function), 25
handle_bcast_rev_event (C++ function), 25
handle_bruck_send_comp_event (C++ function), 25
handle_bruck_send_comp_rev_event (C++ function), 25
handle_coll_complete_event (C++ function), 26
handle_coll_complete_rev_event (C++ function), 26
handle_coll_recv_post_event (C++ function), 26
handle_coll_recv_post_rev_event (C++ function), 26
handle_exec_event (C++ function), 26
handle_exec_rev_event (C++ function), 27
handle_kickoff_event (C++ function), 27
handle_kickoff_rev_event (C++ function), 27
handle_local_event (C++ function), 27
handle_local_rev_event (C++ function), 27
handle_recv_event (C++ function), 28
handle_recv_post_event (C++ function), 28
handle_recv_post_rev_event (C++ function), 28
handle_recv_rev_event (C++ function), 28
handle_scatter_send_comp_event (C++ function), 28
handle_scatter_send_comp_rev_event (C++ function), 29
handle_send_comp_event (C++ function), 29
handle_send_comp_rev_event (C++ function), 29

I

isPEonThisRank (C++ function), 29

J

JobInf (C++ struct), 14

JobInf (C++ type), 49

JobInf::map_file (C++ member), 14

JobInf::numIters (C++ member), 14

JobInf::numRanks (C++ member), 14

JobInf::offsets (C++ member), 14

JobInf::rankMap (C++ member), 14

JobInf::skipMsgId (C++ member), 14

JobInf::traceDir (C++ member), 14

jobs (C++ member), 46

K

KeyType (C++ type), 49

L

lpid_to_job (C++ function), 29

lpid_to_pe (C++ function), 30

M

MPI_INTERNAL_DELAY (C macro), 47

MsgEntry (C++ struct), 14

MsgEntry (C++ type), 50

MsgEntry::msgId (C++ member), 14

MsgEntry::node (C++ member), 14

MsgEntry::thread (C++ member), 14

MsgEntry_getID (C++ function), 30

MsgEntry_getNode (C++ function), 30

MsgEntry_getPE (C++ function), 30

MsgEntry_getSize (C++ function), 30

MsgEntry_getThread (C++ function), 31

MsgID (C++ struct), 15

MsgID (C++ type), 50

MsgID::id (C++ member), 15

MsgID::pe (C++ member), 15

MsgID::size (C++ member), 15

MsgID_getID (C++ function), 31

MsgID_getPE (C++ function), 31

MsgID_getSize (C++ function), 31

MsgKey (C++ class), 17

MsgKey::~MsgKey (C++ function), 17

MsgKey::comm (C++ member), 17

MsgKey::MsgKey (C++ function), 17

MsgKey::operator< (C++ function), 17

MsgKey::rank (C++ member), 17

MsgKey::seq (C++ member), 17

MsgKey::tag (C++ member), 17

N

net_id (C++ member), 46

newMsgEntry (C++ function), 31

newMsgID (C++ function), 32

nic_delay (C++ member), 46

ns_to_s (C++ function), 32

P

PE (C++ class), 18

PE (C++ type), 50

PE::~~PE (C++ function), 18

PE::addTaskExecTime (C++ function), 18

PE::allMarked (C++ member), 18

PE::beforeTask (C++ member), 18

PE::busy (C++ member), 18

PE::collectiveSeq (C++ member), 19

PE::currentCollComm (C++ member), 19

PE::currentCollMsgSize (C++ member), 19

PE::currentCollPartner (C++ member), 19

PE::currentCollRank (C++ member), 19

PE::currentCollRecvCount (C++ member), 19

PE::currentCollSendCount (C++ member), 19

PE::currentCollSeq (C++ member), 19

PE::currentCollSize (C++ member), 19

PE::currentCollTask (C++ member), 19

PE::currentTask (C++ member), 18

PE::currIter (C++ member), 18

PE::currTime (C++ member), 18

PE::findTaskFromMsg (C++ function), 18

PE::firstTask (C++ member), 18

PE::getTaskExecTime (C++ function), 18

PE::goToNextIter (C++ function), 18

PE::invertMsgPe (C++ function), 18

PE::jobNum (C++ member), 18

PE::loop_start_task (C++ member), 18

PE::mark_all_done (C++ function), 18

PE::msgBuffer (C++ member), 18

PE::msgDestLogs (C++ member), 18

PE::msgStatus (C++ member), 18

PE::myEmPE (C++ member), 18

PE::myNum (C++ member), 18

PE::myTasks (C++ member), 18

PE::noUnsatDep (C++ function), 18

PE::numEmPes (C++ member), 19

PE::numWth (C++ member), 19

PE::PE (C++ function), 18

PE::pendingCollMsgs (C++ member), 19

PE::pendingMsgs (C++ member), 19

PE::pendingRCollMsgs (C++ member), 19

PE::pendingReqs (C++ member), 19

PE::pendingRMsgs (C++ member), 19

PE::pendingRReqs (C++ member), 19

PE::printStat (C++ function), 18

PE::recvSeq (C++ member), 19

PE::sendSeq (C++ member), 19

PE::taskExecTime (C++ function), 18

PE::taskExecuted (C++ member), 18
 PE::tasksCount (C++ member), 18
 PE::taskStatus (C++ member), 18
 PE::totalTasksCount (C++ member), 18
 PE_addTaskExecTime (C++ function), 32
 PE_addToBuffer (C++ function), 32
 PE_addToFrontBuffer (C++ function), 32
 PE_clearMsgBuffer (C++ function), 33
 PE_dec_iter (C++ function), 33
 PE_findTaskFromMsg (C++ function), 33
 PE_get_currentTask (C++ function), 33
 PE_get_iter (C++ function), 33
 PE_get_myEmPE (C++ function), 34
 PE_get_myNum (C++ function), 34
 PE_get_numWorkThreads (C++ function), 34
 PE_get_taskDone (C++ function), 34
 PE_get_tasksCount (C++ function), 34
 PE_get_totalTasksCount (C++ function), 35
 PE_getBufferSize (C++ function), 35
 PE_getFirstTask (C++ function), 35
 PE_getNextBuffedMsg (C++ function), 35
 PE_getTaskExecTime (C++ function), 35
 PE_inc_iter (C++ function), 36
 PE_invertMsgPe (C++ function), 36
 PE_is_busy (C++ function), 36
 PE_isEndEvent (C++ function), 36
 PE_isLoopEvent (C++ function), 36
 PE_mark_all_done (C++ function), 37
 PE_noMsgDep (C++ function), 37
 PE_noUnsatDep (C++ function), 37
 PE_printStat (C++ function), 37
 PE_removeFromBuffer (C++ function), 37
 PE_resizeBuffer (C++ function), 38
 PE_set_busy (C++ function), 38
 PE_set_currentTask (C++ function), 38
 PE_set_taskDone (C++ function), 38
 pe_to_job (C++ function), 38
 pe_to_lpid (C++ function), 39
 perform_a2a (C++ function), 39
 perform_a2a_blocked (C++ function), 39
 perform_a2a_blocked_rev (C++ function), 39
 perform_a2a_rev (C++ function), 39
 perform_allgather (C++ function), 40
 perform_allgather_rev (C++ function), 40
 perform_allreduce (C++ function), 40
 perform_allreduce_rev (C++ function), 40
 perform_bcast (C++ function), 41
 perform_bcast_rev (C++ function), 41
 perform_bruck (C++ function), 41
 perform_bruck_rev (C++ function), 41
 perform_collective (C++ function), 41
 perform_collective_rev (C++ function), 42
 perform_reduction (C++ function), 42
 perform_reduction_rev (C++ function), 42
 perform_scatter (C++ function), 42
 perform_scatter_rev (C++ function), 42
 perform_scatter_small (C++ function), 43
 perform_scatter_small_rev (C++ function), 43
 print_frequency (C++ member), 46
 proc_commit_event (C++ function), 43
 proc_event (C++ enum), 21
 proc_event (C++ function), 43
 proc_event::BCAST (C++ enumerator), 21
 proc_event::COLL_A2A (C++ enumerator), 21
 proc_event::COLL_A2A_BLOCKED (C++ enumerator), 21
 proc_event::COLL_A2A_BLOCKED_SEND_DONE (C++ enumerator), 21
 proc_event::COLL_A2A_SEND_DONE (C++ enumerator), 21
 proc_event::COLL_ALLGATHER (C++ enumerator), 21
 proc_event::COLL_ALLGATHER_SEND_DONE (C++ enumerator), 21
 proc_event::COLL_BCAST (C++ enumerator), 21
 proc_event::COLL_BRUCK (C++ enumerator), 21
 proc_event::COLL_BRUCK_SEND_DONE (C++ enumerator), 21
 proc_event::COLL_COMPLETE (C++ enumerator), 21
 proc_event::COLL_REDUCTION (C++ enumerator), 21
 proc_event::COLL_SCATTER (C++ enumerator), 21
 proc_event::COLL_SCATTER_SEND_DONE (C++ enumerator), 21
 proc_event::COLL_SCATTER_SMALL (C++ enumerator), 21
 proc_event::EXEC_COMPLETE (C++ enumerator), 21
 proc_event::KICKOFF (C++ enumerator), 21
 proc_event::LOCAL (C++ enumerator), 21
 proc_event::RECV_COLL_POST (C++ enumerator), 21
 proc_event::RECV_MSG (C++ enumerator), 21
 proc_event::RECV_POST (C++ enumerator), 21
 proc_event::SEND_COMP (C++ enumerator), 21
 proc_finalize (C++ function), 44
 proc_init (C++ function), 44
 proc_msg (C++ struct), 15
 proc_msg::coll_info (C++ member), 15
 proc_msg::coll_info_2 (C++ member), 15
 proc_msg::executed (C++ member), 15
 proc_msg::fwd_dep_count (C++ member), 15
 proc_msg::incremented_flag (C++ member), 15
 proc_msg::iteration (C++ member), 15
 proc_msg::model_net_calls (C++ member), 15

`proc_msg::msgId (C++ member)`, 15
`proc_msg::proc_event_type (C++ member)`, 15
`proc_msg::saved_task (C++ member)`, 15
`proc_msg::src (C++ member)`, 15
`proc_rev_event (C++ function)`, 44
`proc_state (C++ struct)`, 16
`proc_state::end_ts (C++ member)`, 16
`proc_state::my_job (C++ member)`, 16
`proc_state::my_pe (C++ member)`, 16
`proc_state::my_pe_num (C++ member)`, 16
`proc_state::sim_start (C++ member)`, 16
`proc_state::start_ts (C++ member)`, 16

R

`rdma_delay (C++ member)`, 47
`REDUCE_DEGREE (C macro)`, 47

S

`s_to_ns (C++ function)`, 44
`send_coll_comp (C++ function)`, 44
`send_coll_comp_rev (C++ function)`, 45
`send_msg (C++ function)`, 45
`soft_delay_mpi (C++ member)`, 47

T

`Task (C++ class)`, 19
`Task::~~Task (C++ function)`, 19
`Task::endEvent (C++ member)`, 20
`Task::execTime (C++ member)`, 20
`Task::loopEvent (C++ member)`, 20
`Task::loopStartEvent (C++ member)`, 20
`Task::Task (C++ function)`, 19
`TaskPair (C++ struct)`, 16
`TaskPair (C++ type)`, 50
`TaskPair::iter (C++ member)`, 16
`TaskPair::taskid (C++ member)`, 16
`TIME_MULT (C macro)`, 48
`TRACER_A2A_ALG_CUTOFF (C macro)`, 48
`TRACER_ALLGATHER_ALG_CUTOFF (C macro)`, 48
`TRACER_BLOCK_SIZE (C macro)`, 48
`tracer_coll_type (C++ enum)`, 21
`tracer_coll_type::TRACER_COLLECTIVE_ALL_BRUCK (C++ enumerator)`, 22
`tracer_coll_type::TRACER_COLLECTIVE_ALLGATHER_LARGE (C++ enumerator)`, 22
`tracer_coll_type::TRACER_COLLECTIVE_ALLTOALL_BLOCKED (C++ enumerator)`, 21
`tracer_coll_type::TRACER_COLLECTIVE_ALLTOALL_LARGE (C++ enumerator)`, 21
`tracer_coll_type::TRACER_COLLECTIVE_BARRIER (C++ enumerator)`, 21
`tracer_coll_type::TRACER_COLLECTIVE_BCAST (C++ enumerator)`, 21

`tracer_coll_type::TRACER_COLLECTIVE_REDUCE (C++ enumerator)`, 21
`tracer_coll_type::TRACER_COLLECTIVE_SCATTER (C++ enumerator)`, 22
`tracer_coll_type::TRACER_COLLECTIVE_SCATTER_SMALL (C++ enumerator)`, 22
`TRACER_SCATTER_ALG_CUTOFF (C macro)`, 48
`TraceReader (C++ class)`, 20
`TraceReader::~~TraceReader (C++ function)`, 20
`TraceReader::allNodeOffsets (C++ member)`, 20
`TraceReader::fileLoc (C++ member)`, 20
`TraceReader::firstLog (C++ member)`, 20
`TraceReader::numEmPes (C++ member)`, 20
`TraceReader::numWth (C++ member)`, 20
`TraceReader::totalNodes (C++ member)`, 20
`TraceReader::totalTlineLength (C++ member)`, 20
`TraceReader::totalWorkerProcs (C++ member)`, 20
`TraceReader::tracePath (C++ member)`, 20
`TraceReader::TraceReader (C++ function)`, 20
`TraceReader_readOTF2Trace (C++ function)`, 45