



Message Passing Interface

Quick Reference in C

```
#include <mpi.h>
```

Blocking Point-to-Point

Send a message to one process. (§3.2.1)

```
int MPI_Send (void *buf, int count,  
MPI_Datatype datatype, int dest, int  
tag, MPI_Comm comm)
```

Receive a message from one process. (§3.2.4)

```
int MPI_Recv (void *buf, int count,  
MPI_Datatype datatype, int source, int  
tag, MPI_Comm comm, MPI_Status *status)
```

Count received data elements. (§3.2.5)

```
int MPI_Get_count (MPI_Status *status,  
MPI_Datatype datatype, int *count)
```

Wait for message arrival. (§3.8)

```
int MPI_Probe (int source, int tag,  
MPI_Comm comm, MPI_Status *status)
```

Related Functions: MPI_Bsend, MPI_Ssend, MPI_Rsend,

MPI_Buffer_attach, MPI_Buffer_detach, MPI_Sendrecv,
MPI_Sendrecv_replace, MPI_Get_elements

Non-blocking Point-to-Point

Begin to receive a message. (§3.7.2)

```
int MPI_Irecv (void *buf, int count,  
MPI_Datatype, int source, int tag,  
MPI_Comm comm, MPI_Request *request)
```

Complete a non-blocking operation. (§3.7.3)

```
int MPI_Wait (MPI_Request *request,  
MPI_Status *status)
```

Check or complete a non-blocking operation. (§3.7.3)

```
int MPI_Test (MPI_Request *request, int  
*flag, MPI_Status *status)
```

Check message arrival. (§3.8)

```
int MPI_Iprobe (int source, int tag,  
MPI_Comm comm, int *flag, MPI_Status  
*status)
```

Related Functions: MPI_Isend, MPI_Ibsend, MPI_Issend,
MPI_Irsend, MPI_Request_free, MPI_Waitany,
MPI_Testany, MPI_Waitall, MPI_Testall, MPI_Waitosome,
MPI_Testsome, MPI_Cancel, MPI_Test_cancelled

Persistent Requests

Related Functions: MPI_Send_init, MPI_Bsend_init,
MPI_Ssend_init, MPI_Rsend_init, MPI_Recv_init,
MPI_Start, MPI_Startall

Derived Datatypes

Create a strided homogeneous vector. (§3.12.1)

```
int MPI_Type_vector (int count, int  
blocklength, int stride, MPI_Datatype  
oldtype, MPI_Datatype *newtype)
```

Save a derived datatype (§3.12.4)

```
int MPI_Type_commit (MPI_Datatype  
*datatype)
```

Pack data into a message buffer. (§3.13)

```
int MPI_Pack (void *inbuf, int incount,  
MPI_Datatype datatype, void *outbuf,  
int outsize, int *position, MPI_Comm  
comm)
```

Unpack data from a message buffer. (§3.13)

```
int MPI_Unpack (void *inbuf, int insize,  
int *position, void *outbuf, int  
outcount, MPI_Datatype datatype,  
MPI_Comm comm)
```

Determine buffer size for packed data. (§3.13)

```
int MPI_Pack_size (int incount,  
MPI_Datatype datatype, MPI_Comm comm,  
int *size)
```

Related Functions: MPI_Type_contiguous,

MPI_Type_hvector, MPI_Type_indexed,
MPI_Type_hindexed, MPI_Type_struct, MPI_Address,
MPI_Type_extent, MPI_Type_size, MPI_Type_lb,
MPI_Type_ub, MPI_Type_free

Collective

Send one message to all group members. (§4.4)

```
int MPI_Bcast (void *buf, int count,  
MPI_Datatype datatype, int root,  
MPI_Comm comm)
```

Receive from all group members. (§4.5)

```
int MPI_Gather (void *sendbuf, int  
sendcount, MPI_Datatype sendtype, void  
*recvbuf, int recvcount, MPI_Datatype  
recvtype, int root, MPI_Comm comm)
```

Send separate messages to all group members. (§4.6)

```
int MPI_scatter (void *sendbuf, int  
sendcount, MPI_Datatype sendtype, void  
*recvbuf, int recvcount, MPI_Datatype  
recvtype, int root, MPI_Comm comm)
```

Combine messages from all group members. (§4.9.1)

```
int MPI_Reduce (void *sendbuf, void  
*recvbuf, int count, MPI_Datatype  
datatype, MPI_Op op, int root, MPI_Comm  
comm)
```

Related Functions: MPI_Barrier, MPI_Gatherv,

MPI_Scatter, MPI_Allgather, MPI_Allgatherv,
MPI_Alltoall, MPI_Alltoallv, MPI_Op_create,
MPI_Op_free, MPI_Allreduce, MPI_Reduce_scatter,
MPI_Scan

Groups

Related Functions: MPI_Group_size, MPI_Group_rank,

MPI_Group_translate_ranks, MPI_Group_compare,
MPI_Comm_group, MPI_Group_union,
MPI_Group_intersection, MPI_Group_difference,
MPI_Group_incl, MPI_Group_excl,
MPI_Group_range_incl, MPI_Group_range_excl,
MPI_Group_free

Basic Communicators

Count group members in communicator. (§5.4.1)

```
int MPI_Comm_size (MPI_Comm comm, int  
*size)
```

Determine group rank of self. (§5.4.1)

```
int MPI_Comm_rank (MPI_Comm comm, int  
*rank)
```

Duplicate with new context. (§5.4.2)

```
int MPI_Comm_dup (MPI_Comm comm, MPI_Comm  
*newcomm)
```

Split into categorized sub-groups. (§5.4.2)

```
int MPI_Comm_split (MPI_Comm comm, int  
color, int key, MPI_Comm *newcomm)
```

Related Functions: MPI_Comm_compare,

MPI_Comm_create, MPI_Comm_free,

MPI_Comm_test_inter, MPI_Comm_remote_size,
MPI_Comm_remote_group, MPI_Intercomm_create,
MPI_Intercomm_merge

Communicators with Topology

Create with cartesian topology. (§6.5.1)
int **MPI_Cart_create** (MPI_Comm comm_old,
int ndims, int *dims, int *periods, int
reorder, MPI_Comm *comm_cart)

Suggest balanced dimension ranges. (§6.5.2)
int **MPI_Dims_create** (int nnodes, int
ndims, int *dims)

Determine rank from cartesian coordinates. (§6.5.4)
int **MPI_Cart_rank** (MPI_Comm comm, int
*coords, int *rank)

Determine cartesian coordinates from rank. (§6.5.4)
int **MPI_Cart_coords** (MPI_Comm comm, int
rank, int maxdims, int *coords)

Determine ranks for cartesian shift. (§6.5.5)
int **MPI_Cart_shift** (MPI_Comm comm, int
direction, int disp, int *rank_source,
int *rank_dest)

Split into lower dimensional sub-grids. (§6.5.6)
int **MPI_Cart_sub** (MPI_Comm comm, int
*remain_dims, MPI_Comm *newcomm)

Related Functions: MPI_Graph_create, MPI_Topo_test,
MPI_Graphdims_get, MPI_Graph_get,
MPI_Cartdim_get, MPI_Cart_get,
MPI_Graph_neighbors_count, MPI_Graph_neighbors,
MPI_Cart_map, MPI_Graph_map

Communicator Caches

Related Functions: MPI_Keyval_create, MPI_Keyval_free,
MPI_Atr_put, MPI_Atr_get, MPI_Atr_delete

Error Handling

Related Functions: MPI_Errhandler_create,
MPI_Errhandler_set, MPI_Errhandler_get,
MPI_Errhandler_free, MPI_Error_string, MPI_Error_class

Dynamic Processes

Spawn a process. (MPI-2)
int **MPI_Spawn** (char prog[], char *argv[],
int maxprocs, MPI_Info info, int root,
MPI_Comm parents, MPI_Comm *children,
int errs[]);

Related Functions: MPI_Spawn_multiple, MPI_Ispawn,
MPI_Ispawn_multiple, MPI_Port_open, MPI_Port_Close,
MPI_Accept, MPI_Connect, MPI_Name_publish,
MPI_Name_unpublish, MPI_Name_get, MPI_Accept,
MPI_Iconnect, MPI_Info_create, MPI_Info_set,
MPI_Info_get, MPI_Info_get_valuelen,
MPI_Info_get_nkeys, MPI_Info_get_nthkey,
MPI_Info_dup, MPI_Info_free, MPI_Info_delete

Environmental

Determine wall clock time. (§7.4)
double **MPI_wtime** (void)

Initialize MPI. (§7.5)
int **MPI_Init** (int *argc, char **argv)

Cleanup MPI. (§7.5)
int **MPI_Finalize** (void)

Related Functions: MPI_Get_processor_name, MPI_Wtick,
MPI_Initialized, MPI_Abort, MPI_Pcontrol, MPI_Get_version

Constants

Wildcards (§3.2.4)
MPI_ANY_TAG, MPI_ANY_SOURCE

Elementary Datatypes (§3.2.2)
MPI_CHAR, MPI_SHORT, MPI_INT, MPI_LONG,
MPI_UNSIGNED_CHAR, MPI_UNSIGNED_SHORT,
MPI_UNSIGNED, MPI_UNSIGNED_LONG, MPI_FLOAT,
MPI_DOUBLE, MPI_LONG_DOUBLE, MPI_BYTE,
MPI_PACKED

Reserved Communicators (§5.2.4)
MPI_COMM_WORLD, MPI_COMM_SELF, MPI_COMM_PARENT

Reduction Operations (§4.9.2)
MPI_MAX, MPI_MIN, MPI_SUM, MPI_PROD,
MPI_BAND, MPI_BOR, MPI_BXOR, MPI_LAND,
MPI_LOR, MPI_LXOR



LAM Quick Reference

Session Management

Confirm a group of hosts.
recon -v <hostfile>

Start LAM on a group of hosts.
lamboot -v <hostfile>

Terminate LAM.
wipe -v <hostfile>

Hostfile Syntax
comment
<hostname> <userid>
<hostname> <userid>
...etc....

Compilation

Compile a program for LAM / MPI.
hcc -o <binary> <source> -I<incdir>
-L<libdir> -l<lib> -lmpi

Processes and Messages

Start an SPMD application.
mpirun -v -s <src_node> -c <copies>
<nodes> <program> -- <args>

Start a MIMD application.
mpirun -v <appfile>

Appfile Syntax
comment
<program> -s <src_node> <nodes> -- <args>
<program> -s <src_node> <nodes> -- <args>
...etc....

Examine the state of processes.
mpitask
Examine the state of messages.
mpimsg

Cleanup all processes and messages.
lamclean -v

LAM & MPI Information

1224 Kinnear Rd.
Columbus, Ohio 43212
614-292-8492

lam@tbag.osc.edu

<http://www.osc.edu/lam.html>
<ftp://ftp.osc.edu/pub/lam>

