

MPI: A Message-Passing Interface Standard

Version 4.0

(Draft)

Unofficial, for comment only

Message Passing Interface Forum

July 30, 2019

A.2 C Bindings

A.2.1 Point-to-Point Communication C Bindings

```
1 int MPI_Bsend(const void *buf, MPI_Count count, MPI_Datatype datatype,
2               int dest, int tag, MPI_Comm comm)
3
4 int MPI_Bsend(const void *buf, int count, MPI_Datatype datatype, int dest,
5               int tag, MPI_Comm comm)
6
7 int MPI_Bsend_init(const void *buf, MPI_Count count, MPI_Datatype datatype,
8                   int dest, int tag, MPI_Comm comm, MPI_Request *request)
9
10 int MPI_Bsend_init(const void *buf, int count, MPI_Datatype datatype,
11                  int dest, int tag, MPI_Comm comm, MPI_Request *request)
12
13 int MPI_Bsend_init_x(const void *buf, MPI_Count count,
14                    MPI_Datatype datatype, int dest, int tag, MPI_Comm comm,
15                    MPI_Request *request)
16
17 int MPI_Bsend_x(const void *buf, MPI_Count count, MPI_Datatype datatype,
18                int dest, int tag, MPI_Comm comm)
19
20 int MPI_Buffer_attach(void *buffer, MPI_Count size)
21
22 int MPI_Buffer_attach(void *buffer, int size)
23
24 int MPI_Buffer_attach_x(void *buffer, MPI_Count size)
25
26 int MPI_Buffer_detach(void *buffer_addr, MPI_Count *size)
27
28 int MPI_Buffer_detach(void *buffer_addr, int *size)
29
30 int MPI_Buffer_detach_x(void *buffer_addr, MPI_Count *size)
31
32 int MPI_Cancel(MPI_Request *request)
33
34 int MPI_Get_count(const MPI_Status *status, MPI_Datatype datatype,
35                 MPI_Count *count)
36
37 int MPI_Get_count(const MPI_Status *status, MPI_Datatype datatype,
38                 int *count)
39
40 int MPI_Get_count_x(const MPI_Status *status, MPI_Datatype datatype,
41                   MPI_Count *count)
42
43 int MPI_Ibsend(const void *buf, MPI_Count count, MPI_Datatype datatype,
44               int dest, int tag, MPI_Comm comm, MPI_Request *request)
45
46 int MPI_Ibsend(const void *buf, int count, MPI_Datatype datatype, int dest,
47               int tag, MPI_Comm comm, MPI_Request *request)
48
49 int MPI_Ibsend_x(const void *buf, MPI_Count count, MPI_Datatype datatype,
50                 int dest, int tag, MPI_Comm comm, MPI_Request *request)
51
52 int MPI_Improbe(int source, int tag, MPI_Comm comm, int *flag,
53                MPI_Message *message, MPI_Status *status)
```

```
int MPI_Imrecv(void *buf, MPI_Count count, MPI_Datatype datatype,      1
               MPI_Message *message, MPI_Request *request)           2
                                                                    3
int MPI_Imrecv(void *buf, int count, MPI_Datatype datatype,          4
               MPI_Message *message, MPI_Request *request)          5
                                                                    6
int MPI_Imrecv_x(void *buf, MPI_Count count, MPI_Datatype datatype,  7
                 MPI_Message *message, MPI_Request *request)       8
                                                                    9
int MPI_Iprobe(int source, int tag, MPI_Comm comm, int *flag,       9
               MPI_Status *status)                                  10
                                                                    11
int MPI_Irecv(void *buf, MPI_Count count, MPI_Datatype datatype,    11
               int source, int tag, MPI_Comm comm, MPI_Request *request) 12
                                                                    13
int MPI_Irecv(void *buf, int count, MPI_Datatype datatype, int source, 14
               int tag, MPI_Comm comm, MPI_Request *request)       15
                                                                    16
int MPI_Irecv_x(void *buf, MPI_Count count, MPI_Datatype datatype,  16
                 int source, int tag, MPI_Comm comm, MPI_Request *request) 17
                                                                    18
int MPI_Irsend(const void *buf, MPI_Count count, MPI_Datatype datatype, 19
               int dest, int tag, MPI_Comm comm, MPI_Request *request) 20
                                                                    21
int MPI_Irsend(const void *buf, int count, MPI_Datatype datatype, int dest, 22
               int tag, MPI_Comm comm, MPI_Request *request)       23
                                                                    24
int MPI_Irsend_x(const void *buf, MPI_Count count, MPI_Datatype datatype, 24
                 int dest, int tag, MPI_Comm comm, MPI_Request *request) 25
                                                                    26
int MPI_Isend(const void *buf, MPI_Count count, MPI_Datatype datatype, 26
               int dest, int tag, MPI_Comm comm, MPI_Request *request) 27
                                                                    28
int MPI_Isend(const void *buf, int count, MPI_Datatype datatype, int dest, 29
               int tag, MPI_Comm comm, MPI_Request *request)       30
                                                                    31
int MPI_Isend_x(const void *buf, MPI_Count count, MPI_Datatype datatype, 31
                 int dest, int tag, MPI_Comm comm, MPI_Request *request) 32
                                                                    33
int MPI_Issend(const void *buf, MPI_Count count, MPI_Datatype datatype, 34
               int dest, int tag, MPI_Comm comm, MPI_Request *request) 35
                                                                    36
int MPI_Issend(const void *buf, int count, MPI_Datatype datatype, int dest, 37
               int tag, MPI_Comm comm, MPI_Request *request)       38
                                                                    39
int MPI_Issend_x(const void *buf, MPI_Count count, MPI_Datatype datatype, 39
                 int dest, int tag, MPI_Comm comm, MPI_Request *request) 40
                                                                    41
int MPI_Mprobe(int source, int tag, MPI_Comm comm, MPI_Message *message, 42
               MPI_Status *status)                                  43
                                                                    44
int MPI_Mrecv(void *buf, MPI_Count count, MPI_Datatype datatype,    44
               MPI_Message *message, MPI_Status *status)           45
                                                                    46
int MPI_Mrecv(void *buf, int count, MPI_Datatype datatype,          46
               MPI_Message *message, MPI_Status *status)           47
                                                                    48
```

```
1 int MPI_Mrecv_x(void *buf, MPI_Count count, MPI_Datatype datatype,
2               MPI_Message *message, MPI_Status *status)
3
4 int MPI_Probe(int source, int tag, MPI_Comm comm, MPI_Status *status)
5
6 int MPI_Recv(void *buf, MPI_Count count, MPI_Datatype datatype, int source,
7             int tag, MPI_Comm comm, MPI_Status *status)
8
9 int MPI_Recv(void *buf, int count, MPI_Datatype datatype, int source,
10            int tag, MPI_Comm comm, MPI_Status *status)
11
12 int MPI_Recv_init(void *buf, MPI_Count count, MPI_Datatype datatype,
13                 int dest, int tag, MPI_Comm comm, MPI_Request *request)
14
15 int MPI_Recv_init(void *buf, int count, MPI_Datatype datatype, int dest,
16                 int tag, MPI_Comm comm, MPI_Request *request)
17
18 int MPI_Recv_init_x(void *buf, MPI_Count count, MPI_Datatype datatype,
19                   int dest, int tag, MPI_Comm comm, MPI_Request *request)
20
21 int MPI_Recv_x(void *buf, MPI_Count count, MPI_Datatype datatype,
22              int source, int tag, MPI_Comm comm, MPI_Status *status)
23
24 int MPI_Request_free(MPI_Request *request)
25
26 int MPI_Request_get_status(MPI_Request request, int *flag,
27                           MPI_Status *status)
28
29 int MPI_Rsend(const void *buf, MPI_Count count, MPI_Datatype datatype,
30             int dest, int tag, MPI_Comm comm)
31
32 int MPI_Rsend(const void *buf, int count, MPI_Datatype datatype, int dest,
33             int tag, MPI_Comm comm)
34
35 int MPI_Rsend_init(const void *buf, MPI_Count count, MPI_Datatype datatype,
36                  int dest, int tag, MPI_Comm comm, MPI_Request *request)
37
38 int MPI_Rsend_init(const void *buf, int count, MPI_Datatype datatype,
39                  int dest, int tag, MPI_Comm comm, MPI_Request *request)
40
41 int MPI_Rsend_init_x(const void *buf, MPI_Count count,
42                    MPI_Datatype datatype, int dest, int tag, MPI_Comm comm,
43                    MPI_Request *request)
44
45 int MPI_Rsend_x(const void *buf, MPI_Count count, MPI_Datatype datatype,
46               int dest, int tag, MPI_Comm comm)
47
48 int MPI_Send(const void *buf, MPI_Count count, MPI_Datatype datatype,
49            int dest, int tag, MPI_Comm comm)
50
51 int MPI_Send(const void *buf, int count, MPI_Datatype datatype, int dest,
52            int tag, MPI_Comm comm)
53
54 int MPI_Send_init(const void *buf, MPI_Count count, MPI_Datatype datatype,
55                 int dest, int tag, MPI_Comm comm, MPI_Request *request)
```

```
int MPI_Send_init(const void *buf, int count, MPI_Datatype datatype,      1
                  int dest, int tag, MPI_Comm comm, MPI_Request *request)  2
                                                                              3
int MPI_Send_init_x(const void *buf, MPI_Count count,                    4
                    MPI_Datatype datatype, int dest, int tag, MPI_Comm comm,  5
                    MPI_Request *request)                                  6
                                                                              7
int MPI_Send_x(const void *buf, MPI_Count count, MPI_Datatype datatype,  8
               int dest, int tag, MPI_Comm comm)                        9
                                                                              10
int MPI_Sendrecv(const void *sendbuf, MPI_Count sendcount,              11
                 MPI_Datatype sendtype, int dest, int sendtag, void *recvbuf,  12
                 MPI_Count recvcount, MPI_Datatype recvtype, int source,     13
                 int recvtag, MPI_Comm comm, MPI_Status *status)           14
                                                                              15
int MPI_Sendrecv(const void *sendbuf, int sendcount, MPI_Datatype sendtype,  16
                 int dest, int sendtag, void *recvbuf, int recvcount,        17
                 MPI_Datatype recvtype, int source, int recvtag, MPI_Comm comm,  18
                 MPI_Status *status)                                         19
                                                                              20
int MPI_Sendrecv_replace(void *buf, MPI_Count count, MPI_Datatype datatype,  21
                         int dest, int sendtag, int source, int recvtag, MPI_Comm comm,  22
                         MPI_Status *status)                                  23
                                                                              24
int MPI_Sendrecv_replace(void *buf, int count, MPI_Datatype datatype,       25
                         int dest, int sendtag, int source, int recvtag, MPI_Comm comm,  26
                         MPI_Status *status)                                  27
                                                                              28
int MPI_Sendrecv_replace_x(void *buf, MPI_Count count,                    29
                           MPI_Datatype datatype, int dest, int sendtag, int source,     30
                           int recvtag, MPI_Comm comm, MPI_Status *status)       31
                                                                              32
int MPI_Sendrecv_x(const void *sendbuf, MPI_Count sendcount,              33
                  MPI_Datatype sendtype, int dest, int sendtag, void *recvbuf,  34
                  MPI_Count recvcount, MPI_Datatype recvtype, int source,     35
                  int recvtag, MPI_Comm comm, MPI_Status *status)           36
                                                                              37
int MPI_Ssend(const void *buf, MPI_Count count, MPI_Datatype datatype,     38
              int dest, int tag, MPI_Comm comm)                             39
                                                                              40
int MPI_Ssend(const void *buf, int count, MPI_Datatype datatype, int dest,   41
              int tag, MPI_Comm comm)                                       42
                                                                              43
int MPI_Ssend_init(const void *buf, MPI_Count count, MPI_Datatype datatype,  44
                  int dest, int tag, MPI_Comm comm, MPI_Request *request)     45
                                                                              46
int MPI_Ssend_init(const void *buf, int count, MPI_Datatype datatype,        47
                  int dest, int tag, MPI_Comm comm, MPI_Request *request)     48
                                                                              49
int MPI_Ssend_init_x(const void *buf, MPI_Count count,                    50
                    MPI_Datatype datatype, int dest, int tag, MPI_Comm comm,  51
                    MPI_Request *request)                                  52
                                                                              53
int MPI_Ssend_x(const void *buf, MPI_Count count, MPI_Datatype datatype,     54
                int dest, int tag, MPI_Comm comm)                          55
```

```
1         int dest, int tag, MPI_Comm comm)
2
3     int MPI_Start(MPI_Request *request)
4
5     int MPI_Startall(MPI_Count count, MPI_Request array_of_requests[])
6
7     int MPI_Startall(int count, MPI_Request array_of_requests[])
8
9     int MPI_Startall_x(MPI_Count count, MPI_Request array_of_requests[])
10
11    int MPI_Test(MPI_Request *request, int *flag, MPI_Status *status)
12
13    int MPI_Test_cancelled(const MPI_Status *status, int *flag)
14
15    int MPI_Testall(MPI_Count count, MPI_Request array_of_requests[],
16                   int *flag, MPI_Status array_of_statuses[])
17
18    int MPI_Testall(int count, MPI_Request array_of_requests[], int *flag,
19                   MPI_Status array_of_statuses[])
20
21    int MPI_Testall_x(MPI_Count count, MPI_Request array_of_requests[],
22                     int *flag, MPI_Status array_of_statuses[])
23
24    int MPI_Testany(MPI_Count count, MPI_Request array_of_requests[],
25                   int *index, int *flag, MPI_Status *status)
26
27    int MPI_Testany(int count, MPI_Request array_of_requests[], int *index,
28                   int *flag, MPI_Status *status)
29
30    int MPI_Testany_x(MPI_Count count, MPI_Request array_of_requests[],
31                     int *index, int *flag, MPI_Status *status)
32
33    int MPI_Testsome(MPI_Count incount, MPI_Request array_of_requests[],
34                    MPI_Count *outcount, int array_of_indices[],
35                    MPI_Status array_of_statuses[])
36
37    int MPI_Testsome(int incount, MPI_Request array_of_requests[],
38                    int *outcount, int array_of_indices[],
39                    MPI_Status array_of_statuses[])
40
41    int MPI_Testsome_x(MPI_Count incount, MPI_Request array_of_requests[],
42                      MPI_Count *outcount, int array_of_indices[],
43                      MPI_Status array_of_statuses[])
44
45    int MPI_Wait(MPI_Request *request, MPI_Status *status)
46
47    int MPI_Waitall(MPI_Count count, MPI_Request array_of_requests[],
48                   MPI_Status array_of_statuses[])
49
50    int MPI_Waitall(int count, MPI_Request array_of_requests[],
51                   MPI_Status array_of_statuses[])
52
53    int MPI_Waitall_x(MPI_Count count, MPI_Request array_of_requests[],
54                     MPI_Status array_of_statuses[])
55
56    int MPI_Waitany(MPI_Count count, MPI_Request array_of_requests[],
57                   int *index, MPI_Status *status)
```

```

int MPI_Waitany(int count, MPI_Request array_of_requests[], int *index,
               MPI_Status *status)
int MPI_Waitany_x(MPI_Count count, MPI_Request array_of_requests[],
                 int *index, MPI_Status *status)
int MPI_Waitsome(MPI_Count incount, MPI_Request array_of_requests[],
                MPI_Count *outcount, int array_of_indices[],
                MPI_Status array_of_statuses[])
int MPI_Waitsome(int incount, MPI_Request array_of_requests[],
                 int *outcount, int array_of_indices[],
                 MPI_Status array_of_statuses[])
int MPI_Waitsome_x(MPI_Count incount, MPI_Request array_of_requests[],
                  MPI_Count *outcount, int array_of_indices[],
                  MPI_Status array_of_statuses[])

```

A.2.2 Datatypes C Bindings

```

MPI_Aint MPI_Aint_add(MPI_Aint base, MPI_Aint disp)
MPI_Aint MPI_Aint_diff(MPI_Aint addr1, MPI_Aint addr2)
int MPI_Get_address(const void *location, MPI_Aint *address)
int MPI_Get_elements(const MPI_Status *status, MPI_Datatype datatype,
                    int *count)
int MPI_Get_elements_x(const MPI_Status *status, MPI_Datatype datatype,
                      MPI_Count *count)
int MPI_Pack(const void* inbuf, int incount, MPI_Datatype datatype,
            void *outbuf, int outsize, int *position, MPI_Comm comm)
int MPI_Pack_external(const char datarep[], const void *inbuf, int incount,
                    MPI_Datatype datatype, void *outbuf, MPI_Aint outsize,
                    MPI_Aint *position)
int MPI_Pack_external_size(const char datarep[], int incount,
                          MPI_Datatype datatype, MPI_Aint *size)
int MPI_Pack_size(int incount, MPI_Datatype datatype, MPI_Comm comm,
                 int *size)
int MPI_Type_commit(MPI_Datatype *datatype)
int MPI_Type_contiguous(MPI_Count count, MPI_Datatype oldtype,
                       MPI_Datatype *newtype)
int MPI_Type_contiguous(int count, MPI_Datatype oldtype,
                       MPI_Datatype *newtype)
int MPI_Type_contiguous(int count, MPI_Datatype oldtype,
                       MPI_Datatype *newtype)

```

```
1 int MPI_Type_contiguous_x(MPI_Count count, MPI_Datatype oldtype,
2     MPI_Datatype *newtype)
3
4 int MPI_Type_create_darray(int size, int rank, int ndims,
5     const int array_of_gsizes[], const int array_of_distrib[],
6     const int array_of_dargs[], const int array_of_psize[],
7     int order, MPI_Datatype oldtype, MPI_Datatype *newtype)
8
9 int MPI_Type_create_hindexed(int count, const int array_of_blocklengths[],
10     const MPI_Aint array_of_displacements[], MPI_Datatype oldtype,
11     MPI_Datatype *newtype)
12
13 int MPI_Type_create_hindexed_block(int count, int blocklength,
14     const MPI_Aint array_of_displacements[], MPI_Datatype oldtype,
15     MPI_Datatype *newtype)
16
17 int MPI_Type_create_hvector(int count, int blocklength, MPI_Aint stride,
18     MPI_Datatype oldtype, MPI_Datatype *newtype)
19
20 int MPI_Type_create_indexed_block(int count, int blocklength,
21     const int array_of_displacements[], MPI_Datatype oldtype,
22     MPI_Datatype *newtype)
23
24 int MPI_Type_create_resized(MPI_Datatype oldtype, MPI_Aint lb,
25     MPI_Aint extent, MPI_Datatype *newtype)
26
27 int MPI_Type_create_struct(int count, const int array_of_blocklengths[],
28     const MPI_Aint array_of_displacements[],
29     const MPI_Datatype array_of_types[], MPI_Datatype *newtype)
30
31 int MPI_Type_create_subarray(int ndims, const int array_of_sizes[],
32     const int array_of_subsizes[], const int array_of_starts[],
33     int order, MPI_Datatype oldtype, MPI_Datatype *newtype)
34
35 int MPI_Type_dup(MPI_Datatype oldtype, MPI_Datatype *newtype)
36
37 int MPI_Type_free(MPI_Datatype *datatype)
38
39 int MPI_Type_get_contents(MPI_Datatype datatype, int max_integers,
40     int max_addresses, int max_datatypes, int array_of_integers[],
41     MPI_Aint array_of_addresses[],
42     MPI_Datatype array_of_datatypes[])
43
44 int MPI_Type_get_envelope(MPI_Datatype datatype, int *num_integers,
45     int *num_addresses, int *num_datatypes, int *combiner)
46
47 int MPI_Type_get_extent(MPI_Datatype datatype, MPI_Aint *lb,
48     MPI_Aint *extent)
49
50 int MPI_Type_get_extent_x(MPI_Datatype datatype, MPI_Count *lb,
51     MPI_Count *extent)
52
53 int MPI_Type_get_true_extent(MPI_Datatype datatype, MPI_Aint *true_lb,
54     MPI_Aint *true_extent)
```



```

int MPI_Type_get_true_extent_x(MPI_Datatype datatype, MPI_Count *true_lb, 1
                               MPI_Count *true_extent) 2
3
int MPI_Type_indexed(int count, const int array_of_blocklengths[], 4
                     const int array_of_displacements[], MPI_Datatype oldtype, 5
                     MPI_Datatype *newtype) 6
7
int MPI_Type_size(MPI_Datatype datatype, int *size) 7
8
int MPI_Type_size_x(MPI_Datatype datatype, MPI_Count *size) 9
10
int MPI_Type_vector(int count, int blocklength, int stride, 10
                    MPI_Datatype oldtype, MPI_Datatype *newtype) 11
12
int MPI_Unpack(const void* inbuf, int insize, int *position, void *outbuf, 13
              int outcount, MPI_Datatype datatype, MPI_Comm comm) 14
15
int MPI_Unpack_external(const char datarep[], const void *inbuf, 16
                        MPI_Aint insize, MPI_Aint *position, void *outbuf, 17
                        int outcount, MPI_Datatype datatype) 18
19

```

A.2.3 Collective Communication C Bindings

```

int MPI_Allgather(const void* sendbuf, int sendcount, 21
                  MPI_Datatype sendtype, void* recvbuf, int recvcount, 22
                  MPI_Datatype recvtype, MPI_Comm comm) 23
24
int MPI_Allgather_init(const void* sendbuf, int sendcount, 25
                       MPI_Datatype sendtype, void* recvbuf, int recvcount, 26
                       MPI_Datatype recvtype, MPI_Comm comm, MPI_Info info, 27
                       MPI_Request *request) 28
29
int MPI_Allgatherv(const void* sendbuf, int sendcount, 30
                   MPI_Datatype sendtype, void* recvbuf, const int recvcounts[], 31
                   const int displs[], MPI_Datatype recvtype, MPI_Comm comm) 32
33
int MPI_Allgatherv_init(const void* sendbuf, int sendcount, 33
                        MPI_Datatype sendtype, void* recvbuf, const int recvcounts[], 34
                        const int displs[], MPI_Datatype recvtype, MPI_Comm comm, 35
                        MPI_Info info, MPI_Request* request) 36
37
int MPI_Allreduce(const void* sendbuf, void* recvbuf, int count, 38
                  MPI_Datatype datatype, MPI_Op op, MPI_Comm comm) 39
40
int MPI_Allreduce_init(const void* sendbuf, void* recvbuf, int count, 40
                       MPI_Datatype datatype, MPI_Op op, MPI_Comm comm, 41
                       MPI_Info info, MPI_Request *request) 42
43
int MPI_Alltoall(const void* sendbuf, int sendcount, MPI_Datatype sendtype, 44
                 void* recvbuf, int recvcount, MPI_Datatype recvtype, 45
                 MPI_Comm comm) 46
47
int MPI_Alltoall_init(const void* sendbuf, int sendcount, 47

```

```
1         MPI_Datatype sendtype, void* recvbuf, int recvcnt,
2         MPI_Datatype recvttype, MPI_Comm comm, MPI_Info info,
3         MPI_Request *request)
4
5     int MPI_Alltoallv(const void* sendbuf, const int sendcounts[],
6         const int sdispls[], MPI_Datatype sendtype, void* recvbuf,
7         const int recvcnt[], const int rdispls[],
8         MPI_Datatype recvttype, MPI_Comm comm)
9
10    int MPI_Alltoallv_init(const void* sendbuf, const int sendcounts[],
11        const int sdispls[], MPI_Datatype sendtype, void* recvbuf,
12        const int recvcnt[], const int rdispls[],
13        MPI_Datatype recvttype, MPI_Comm comm, MPI_Info info,
14        MPI_Request *request)
15
16    int MPI_Alltoallw(const void* sendbuf, const int sendcounts[],
17        const int sdispls[], const MPI_Datatype sendtypes[],
18        void* recvbuf, const int recvcnt[], const int rdispls[],
19        const MPI_Datatype recvtypes[], MPI_Comm comm)
20
21    int MPI_Alltoallw_init(const void* sendbuf, const int sendcounts[],
22        const int sdispls[], const MPI_Datatype sendtypes[],
23        void* recvbuf, const int recvcnt[], const int rdispls[],
24        const MPI_Datatype recvtypes[], MPI_Comm comm, MPI_Info info,
25        MPI_Request *request)
26
27    int MPI_Barrier(MPI_Comm comm)
28
29    int MPI_Barrier_init(MPI_Comm comm, MPI_Info info, MPI_Request *request)
30
31    int MPI_Bcast(void* buffer, int count, MPI_Datatype datatype, int root,
32        MPI_Comm comm)
33
34    int MPI_Bcast_init(void* buffer, int count, MPI_Datatype datatype,
35        int root, MPI_Comm comm, MPI_Info info, MPI_Request *request)
36
37    int MPI_Exscan(const void* sendbuf, void* recvbuf, int count,
38        MPI_Datatype datatype, MPI_Op op, MPI_Comm comm)
39
40    int MPI_Exscan_init(const void* sendbuf, void* recvbuf, int count,
41        MPI_Datatype datatype, MPI_Op op, MPI_Comm comm,
42        MPI_Info info, MPI_Request *request)
43
44    int MPI_Gather(const void* sendbuf, int sendcount, MPI_Datatype sendtype,
45        void* recvbuf, int recvcnt, MPI_Datatype recvttype, int root,
46        MPI_Comm comm)
47
48    int MPI_Gather_init(const void* sendbuf, int sendcount,
49        MPI_Datatype sendtype, void* recvbuf, int recvcnt,
50        MPI_Datatype recvttype, int root, MPI_Comm comm, MPI_Info info,
51        MPI_Request *request)
52
53    int MPI_Gatherv(const void* sendbuf, int sendcount, MPI_Datatype sendtype,
54        void* recvbuf, const int recvcnt[], const int displs[],
```

```
    MPI_Datatype recvtype, int root, MPI_Comm comm) 1
int MPI_Gatherv_init(const void* sendbuf, int sendcount, 2
    MPI_Datatype sendtype, void* recvbuf, const int recvcnts[], 3
    const int displs[], MPI_Datatype recvtype, int root, 4
    MPI_Comm comm, MPI_Info info, MPI_Request *request) 5
int MPI_Iallgather(const void* sendbuf, int sendcount, 6
    MPI_Datatype sendtype, void* recvbuf, int recvcount, 7
    MPI_Datatype recvtype, MPI_Comm comm, MPI_Request *request) 8
int MPI_Iallgatherv(const void* sendbuf, int sendcount, 9
    MPI_Datatype sendtype, void* recvbuf, const int recvcnts[], 10
    const int displs[], MPI_Datatype recvtype, MPI_Comm comm, 11
    MPI_Request* request) 12
int MPI_Iallreduce(const void* sendbuf, void* recvbuf, int count, 13
    MPI_Datatype datatype, MPI_Op op, MPI_Comm comm, 14
    MPI_Request *request) 15
int MPI_Ialltoall(const void* sendbuf, int sendcount, 16
    MPI_Datatype sendtype, void* recvbuf, int recvcount, 17
    MPI_Datatype recvtype, MPI_Comm comm, MPI_Request *request) 18
int MPI_Ialltoallv(const void* sendbuf, const int sendcounts[], 19
    const int sdispls[], MPI_Datatype sendtype, void* recvbuf, 20
    const int recvcnts[], const int rdispls[], 21
    MPI_Datatype recvtype, MPI_Comm comm, MPI_Request *request) 22
int MPI_Ialltoallw(const void* sendbuf, const int sendcounts[], 23
    const int sdispls[], const MPI_Datatype sendtypes[], 24
    void* recvbuf, const int recvcnts[], const int rdispls[], 25
    const MPI_Datatype recvtypes[], MPI_Comm comm, 26
    MPI_Request *request) 27
int MPI_Ibarrier(MPI_Comm comm, MPI_Request *request) 28
int MPI_Ibcast(void* buffer, int count, MPI_Datatype datatype, int root, 29
    MPI_Comm comm, MPI_Request *request) 30
int MPI_Iexscan(const void* sendbuf, void* recvbuf, int count, 31
    MPI_Datatype datatype, MPI_Op op, MPI_Comm comm, 32
    MPI_Request *request) 33
int MPI_Igather(const void* sendbuf, int sendcount, MPI_Datatype sendtype, 34
    void* recvbuf, int recvcount, MPI_Datatype recvtype, int root, 35
    MPI_Comm comm, MPI_Request *request) 36
int MPI_Igatherv(const void* sendbuf, int sendcount, MPI_Datatype sendtype, 37
    void* recvbuf, const int recvcnts[], const int displs[], 38
    MPI_Datatype recvtype, int root, MPI_Comm comm, 39
    MPI_Request *request) 40
int MPI_Ireduce(const void* sendbuf, void* recvbuf, int count, 41
    MPI_Datatype datatype, MPI_Op op, MPI_Comm comm, 42
    MPI_Request *request) 43
int MPI_Ireduce_sc(const void* sendbuf, void* recvbuf, int count, 44
    MPI_Datatype datatype, MPI_Op op, MPI_Comm comm, 45
    MPI_Request *request) 46
int MPI_Ireduce_w(const void* sendbuf, void* recvbuf, int count, 47
    MPI_Datatype datatype, MPI_Op op, MPI_Comm comm, 48
    MPI_Datatype recvtype, MPI_Request *request) 49
```

```
1         MPI_Datatype datatype, MPI_Op op, int root, MPI_Comm comm,
2         MPI_Request *request)
3
4     int MPI_Ireduce_scatter(const void* sendbuf, void* recvbuf,
5         const int recvcnts[], MPI_Datatype datatype, MPI_Op op,
6         MPI_Comm comm, MPI_Request *request)
7
8     int MPI_Ireduce_scatter_block(const void* sendbuf, void* recvbuf,
9         int recvcnt, MPI_Datatype datatype, MPI_Op op,
10        MPI_Comm comm, MPI_Request *request)
11
12    int MPI_Iscan(const void* sendbuf, void* recvbuf, int count,
13        MPI_Datatype datatype, MPI_Op op, MPI_Comm comm,
14        MPI_Request *request)
15
16    int MPI_Iscatter(const void* sendbuf, int sendcount, MPI_Datatype sendtype,
17        void* recvbuf, int recvcnt, MPI_Datatype recvtype, int root,
18        MPI_Comm comm, MPI_Request *request)
19
20    int MPI_Iscatterv(const void* sendbuf, const int sendcounts[],
21        const int displs[], MPI_Datatype sendtype, void* recvbuf,
22        int recvcnt, MPI_Datatype recvtype, int root, MPI_Comm comm,
23        MPI_Request *request)
24
25    int MPI_Op_commutative(MPI_Op op, int *commute)
26
27    int MPI_Op_create(MPI_User_function* user_fn, int commute, MPI_Op* op)
28
29    int MPI_Op_free(MPI_Op *op)
30
31    int MPI_Reduce(const void* sendbuf, void* recvbuf, int count,
32        MPI_Datatype datatype, MPI_Op op, int root, MPI_Comm comm)
33
34    int MPI_Reduce_init(const void* sendbuf, void* recvbuf, int count,
35        MPI_Datatype datatype, MPI_Op op, int root, MPI_Comm comm,
36        MPI_Info info, MPI_Request *request)
37
38    int MPI_Reduce_local(const void* inbuf, void* inoutbuf, int count,
39        MPI_Datatype datatype, MPI_Op op)
40
41    int MPI_Reduce_scatter(const void* sendbuf, void* recvbuf,
42        const int recvcnts[], MPI_Datatype datatype, MPI_Op op,
43        MPI_Comm comm)
44
45    int MPI_Reduce_scatter_block(const void* sendbuf, void* recvbuf,
46        int recvcnt, MPI_Datatype datatype, MPI_Op op,
47        MPI_Comm comm)
48
49    int MPI_Reduce_scatter_block_init(const void* sendbuf, void* recvbuf,
50        int recvcnt, MPI_Datatype datatype, MPI_Op op,
51        MPI_Comm comm, MPI_Info info, MPI_Request *request)
52
53    int MPI_Reduce_scatter_init(const void* sendbuf, void* recvbuf,
54        const int recvcnts[], MPI_Datatype datatype, MPI_Op op,
55        MPI_Comm comm, MPI_Info info, MPI_Request *request)
```

```

int MPI_Scan(const void* sendbuf, void* recvbuf, int count,
             MPI_Datatype datatype, MPI_Op op, MPI_Comm comm)
int MPI_Scan_init(const void* sendbuf, void* recvbuf, int count,
                 MPI_Datatype datatype, MPI_Op op, MPI_Comm comm,
                 MPI_Info info, MPI_Request *request)
int MPI_Scatter(const void* sendbuf, int sendcount, MPI_Datatype sendtype,
               void* recvbuf, int recvcount, MPI_Datatype recvtype, int root,
               MPI_Comm comm)
int MPI_Scatter_init(const void* sendbuf, int sendcount,
                    MPI_Datatype sendtype, void* recvbuf, int recvcount,
                    MPI_Datatype recvtype, int root, MPI_Comm comm, MPI_Info info,
                    MPI_Request *request)
int MPI_Scatterv(const void* sendbuf, const int sendcounts[],
                const int displs[], MPI_Datatype sendtype, void* recvbuf,
                int recvcount, MPI_Datatype recvtype, int root, MPI_Comm comm)
int MPI_Scatterv_init(const void* sendbuf, const int sendcounts[],
                     const int displs[], MPI_Datatype sendtype, void* recvbuf,
                     int recvcount, MPI_Datatype recvtype, int root, MPI_Comm comm,
                     MPI_Info info, MPI_Request *request)

```

A.2.4 Groups, Contexts, Communicators, and Caching C Bindings

```

int MPI_COMM_DUP_FN(MPI_Comm oldcomm, int comm_keyval, void *extra_state,
                   void *attribute_val_in, void *attribute_val_out, int *flag)
int MPI_COMM_NULL_COPY_FN(MPI_Comm oldcomm, int comm_keyval,
                           void *extra_state, void *attribute_val_in,
                           void *attribute_val_out, int *flag)
int MPI_COMM_NULL_DELETE_FN(MPI_Comm comm, int comm_keyval,
                             void *attribute_val, void *extra_state)
int MPI_Comm_compare(MPI_Comm comm1, MPI_Comm comm2, int *result)
int MPI_Comm_create(MPI_Comm comm, MPI_Group group, MPI_Comm *newcomm)
int MPI_Comm_create_group(MPI_Comm comm, MPI_Group group, int tag,
                           MPI_Comm *newcomm)
int MPI_Comm_create_keyval(MPI_Comm_copy_attr_function *comm_copy_attr_fn,
                           MPI_Comm_delete_attr_function *comm_delete_attr_fn,
                           int *comm_keyval, void *extra_state)
int MPI_Comm_delete_attr(MPI_Comm comm, int comm_keyval)
int MPI_Comm_dup(MPI_Comm comm, MPI_Comm *newcomm)
int MPI_Comm_dup_with_info(MPI_Comm comm, MPI_Info info, MPI_Comm *newcomm)

```

```
1 int MPI_Comm_free(MPI_Comm *comm)
2
3 int MPI_Comm_free_keyval(int *comm_keyval)
4
5 int MPI_Comm_get_attr(MPI_Comm comm, int comm_keyval, void *attribute_val,
6                       int *flag)
7
8 int MPI_Comm_get_info(MPI_Comm comm, MPI_Info *info_used)
9
10 int MPI_Comm_get_name(MPI_Comm comm, char *comm_name, int *resultlen)
11
12 int MPI_Comm_group(MPI_Comm comm, MPI_Group *group)
13
14 int MPI_Comm_idup(MPI_Comm comm, MPI_Comm *newcomm, MPI_Request *request)
15
16 int MPI_Comm_idup_with_info(MPI_Comm comm, MPI_Info info,
17                             MPI_Comm *newcomm, MPI_Request *request)
18
19 int MPI_Comm_rank(MPI_Comm comm, int *rank)
20
21 int MPI_Comm_remote_group(MPI_Comm comm, MPI_Group *group)
22
23 int MPI_Comm_remote_size(MPI_Comm comm, int *size)
24
25 int MPI_Comm_set_attr(MPI_Comm comm, int comm_keyval, void *attribute_val)
26
27 int MPI_Comm_set_info(MPI_Comm comm, MPI_Info info)
28
29 int MPI_Comm_set_name(MPI_Comm comm, const char *comm_name)
30
31 int MPI_Comm_size(MPI_Comm comm, int *size)
32
33 int MPI_Comm_split(MPI_Comm comm, int color, int key, MPI_Comm *newcomm)
34
35 int MPI_Comm_split_type(MPI_Comm comm, int split_type, int key,
36                         MPI_Info info, MPI_Comm *newcomm)
37
38 int MPI_Comm_test_inter(MPI_Comm comm, int *flag)
39
40 int MPI_Group_compare(MPI_Group group1, MPI_Group group2, int *result)
41
42 int MPI_Group_difference(MPI_Group group1, MPI_Group group2,
43                         MPI_Group *newgroup)
44
45 int MPI_Group_excl(MPI_Group group, int n, const int ranks[],
46                   MPI_Group *newgroup)
47
48 int MPI_Group_free(MPI_Group *group)
49
50 int MPI_Group_incl(MPI_Group group, int n, const int ranks[],
51                   MPI_Group *newgroup)
52
53 int MPI_Group_intersection(MPI_Group group1, MPI_Group group2,
54                            MPI_Group *newgroup)
55
56 int MPI_Group_range_excl(MPI_Group group, int n, int ranges[][3],
57                          MPI_Group *newgroup)
```

```
int MPI_Group_range_incl(MPI_Group group, int n, int ranges[][3],      1
                        MPI_Group *newgroup)                          2
int MPI_Group_rank(MPI_Group group, int *rank)                        3
int MPI_Group_size(MPI_Group group, int *size)                       4
int MPI_Group_translate_ranks(MPI_Group group1, int n, const int ranks1[], 5
                              MPI_Group group2, int ranks2[])        6
int MPI_Group_union(MPI_Group group1, MPI_Group group2,              7
                   MPI_Group *newgroup)                             8
int MPI_Intercomm_create(MPI_Comm local_comm, int local_leader,      9
                        MPI_Comm peer_comm, int remote_leader, int tag, 10
                        MPI_Comm *newintercomm)                     11
int MPI_Intercomm_merge(MPI_Comm intercomm, int high,               12
                        MPI_Comm *newintracomm)                     13
int MPI_Type_create_keyval(MPI_Datatype oldtype, int type_keyval,   14
                          void *extra_state, void *attribute_val_in, 15
                          void *attribute_val_out, int *flag)       16
int MPI_Type_delete_attr(MPI_Datatype datatype, int type_keyval,   17
                        void *attribute_val, void *extra_state)     18
int MPI_Type_get_attr(MPI_Datatype datatype, int type_keyval,      19
                     void *attribute_val, int *flag)               20
int MPI_Type_get_name(MPI_Datatype datatype, char *type_name,     21
                     int *resultlen)                               22
int MPI_Type_set_attr(MPI_Datatype datatype, int type_keyval,      23
                     void *attribute_val)                          24
int MPI_Type_set_name(MPI_Datatype datatype, const char *type_name) 25
int MPI_Win_dup(MPI_Win oldwin, int win_keyval, void *extra_state, 26
               void *attribute_val_in, void *attribute_val_out, int *flag) 27
int MPI_Win_null_copy(MPI_Win oldwin, int win_keyval, void *extra_state, 28
                    void *attribute_val_in, void *attribute_val_out, int *flag) 29
int MPI_Group_translate_ranks(MPI_Group group1, int n, const int ranks1[], 30
                              MPI_Group group2, int ranks2[])        31
int MPI_Group_rank(MPI_Group group, int *rank)                      32
int MPI_Group_size(MPI_Group group, int *size)                     33
int MPI_Group_union(MPI_Group group1, MPI_Group group2,            34
                   MPI_Group *newgroup)                           35
int MPI_Intercomm_create(MPI_Comm local_comm, int local_leader,    36
                        MPI_Comm peer_comm, int remote_leader, int tag, 37
                        MPI_Comm *newintercomm)                     38
int MPI_Intercomm_merge(MPI_Comm intercomm, int high,               39
                        MPI_Comm *newintracomm)                     40
int MPI_Type_create_keyval(MPI_Datatype oldtype, int type_keyval,   41
                          void *extra_state, void *attribute_val_in, 42
                          void *attribute_val_out, int *flag)       43
int MPI_Type_delete_attr(MPI_Datatype datatype, int type_keyval,   44
                        void *attribute_val, void *extra_state)     45
int MPI_Type_get_attr(MPI_Datatype datatype, int type_keyval,      46
                     void *attribute_val, int *flag)               47
int MPI_Type_get_name(MPI_Datatype datatype, char *type_name,     48
                     int *resultlen)                               49
int MPI_Type_set_attr(MPI_Datatype datatype, int type_keyval,      50
                     void *attribute_val)                          51
int MPI_Type_set_name(MPI_Datatype datatype, const char *type_name) 52
int MPI_Win_dup(MPI_Win oldwin, int win_keyval, void *extra_state, 53
               void *attribute_val_in, void *attribute_val_out, int *flag) 54
int MPI_Win_null_copy(MPI_Win oldwin, int win_keyval, void *extra_state, 55
                    void *attribute_val_in, void *attribute_val_out, int *flag) 56
```

```

1  int MPI_WIN_NULL_DELETE_FN(MPI_Win win, int win_keyval,
2      void *attribute_val, void *extra_state)
3
4  int MPI_Win_create_keyval(MPI_Win_copy_attr_function *win_copy_attr_fn,
5      MPI_Win_delete_attr_function *win_delete_attr_fn,
6      int *win_keyval, void *extra_state)
7
8  int MPI_Win_delete_attr(MPI_Win win, int win_keyval)
9
10 int MPI_Win_free_keyval(int *win_keyval)
11
12 int MPI_Win_get_attr(MPI_Win win, int win_keyval, void *attribute_val,
13     int *flag)
14
15 int MPI_Win_get_name(MPI_Win win, char *win_name, int *resultlen)
16
17 int MPI_Win_set_attr(MPI_Win win, int win_keyval, void *attribute_val)
18
19 int MPI_Win_set_name(MPI_Win win, const char *win_name)
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48

```

A.2.5 Process Topologies C Bindings

```

20 int MPI_Cart_coords(MPI_Comm comm, int rank, int maxdims, int coords[])
21
22 int MPI_Cart_create(MPI_Comm comm_old, int ndims, const int dims[],
23     const int periods[], int reorder, MPI_Comm *comm_cart)
24
25 int MPI_Cart_get(MPI_Comm comm, int maxdims, int dims[], int periods[],
26     int coords[])
27
28 int MPI_Cart_map(MPI_Comm comm, int ndims, const int dims[],
29     const int periods[], int *newrank)
30
31 int MPI_Cart_rank(MPI_Comm comm, const int coords[], int *rank)
32
33 int MPI_Cart_shift(MPI_Comm comm, int direction, int disp,
34     int *rank_source, int *rank_dest)
35
36 int MPI_Cart_sub(MPI_Comm comm, const int remain_dims[], MPI_Comm *newcomm)
37
38 int MPI_Cartdim_get(MPI_Comm comm, int *ndims)
39
40 int MPI_Dims_create(int nnodes, int ndims, int dims[])
41
42 int MPI_Dist_graph_create(MPI_Comm comm_old, int n, const int sources[],
43     const int degrees[], const int destinations[],
44     const int weights[], MPI_Info info, int reorder,
45     MPI_Comm *comm_dist_graph)
46
47 int MPI_Dist_graph_create_adjacent(MPI_Comm comm_old, int indegree,
48     const int sources[], const int sourceweights[], int outdegree,
49     const int destinations[], const int destweights[],
50     MPI_Info info, int reorder, MPI_Comm *comm_dist_graph)
51
52 int MPI_Dist_graph_neighbors(MPI_Comm comm, int maxindegree, int sources[],
53     int sourceweights[], int maxoutdegree, int destinations[],
54     int destweights[])

```



```
    int destweights[]) 1
int MPI_Dist_graph_neighbors_count(MPI_Comm comm, int *indegree, 2
    int *outdegree, int *weighted) 3
int MPI_Graph_create(MPI_Comm comm_old, int nnodes, const int index[], 4
    const int edges[], int reorder, MPI_Comm *comm_graph) 5
int MPI_Graph_get(MPI_Comm comm, int maxindex, int maxedges, int index[], 6
    int edges[]) 7
int MPI_Graph_map(MPI_Comm comm, int nnodes, const int index[], 8
    const int edges[], int *newrank) 9
int MPI_Graph_neighbors(MPI_Comm comm, int rank, int maxneighbors, 10
    int neighbors[]) 11
int MPI_Graph_neighbors_count(MPI_Comm comm, int rank, int *nneighbors) 12
int MPI_Graphdims_get(MPI_Comm comm, int *nnodes, int *nedges) 13
int MPI_Ineighbor_allgather(const void* sendbuf, int sendcount, 14
    MPI_Datatype sendtype, void* recvbuf, int recvcount, 15
    MPI_Datatype recvtype, MPI_Comm comm, MPI_Request *request) 16
int MPI_Ineighbor_allgatherv(const void* sendbuf, int sendcount, 17
    MPI_Datatype sendtype, void* recvbuf, const int recvcounts[], 18
    const int displs[], MPI_Datatype recvtype, MPI_Comm comm, 19
    MPI_Request *request) 20
int MPI_Ineighbor_alltoall(const void* sendbuf, int sendcount, 21
    MPI_Datatype sendtype, void* recvbuf, int recvcount, 22
    MPI_Datatype recvtype, MPI_Comm comm, MPI_Request *request) 23
int MPI_Ineighbor_alltoallv(const void* sendbuf, const int sendcounts[], 24
    const int sdispls[], MPI_Datatype sendtype, void* recvbuf, 25
    const int recvcounts[], const int rdispls[], 26
    MPI_Datatype recvtype, MPI_Comm comm, MPI_Request *request) 27
int MPI_Ineighbor_alltoallw(const void* sendbuf, const int sendcounts[], 28
    const MPI_Aint sdispls[], const MPI_Datatype sendtypes[], 29
    void* recvbuf, const int recvcounts[], 30
    const MPI_Aint rdispls[], const MPI_Datatype recvtypes[], 31
    MPI_Comm comm, MPI_Request *request) 32
int MPI_Neighbor_allgather(const void* sendbuf, int sendcount, 33
    MPI_Datatype sendtype, void* recvbuf, int recvcount, 34
    MPI_Datatype recvtype, MPI_Comm comm) 35
int MPI_Neighbor_allgather_init(const void* sendbuf, int sendcount, 36
    MPI_Datatype sendtype, void* recvbuf, int recvcount, 37
    MPI_Datatype recvtype, MPI_Comm comm, MPI_Info info, 38
    MPI_Request *request) 39
int MPI_Neighbor_allgatherv(const void* sendbuf, int sendcount, 40
```

```

1         MPI_Datatype sendtype, void* recvbuf, const int recvcnts[],
2         const int displs[], MPI_Datatype recvtype, MPI_Comm comm)
3
4     int MPI_Neighbor_allgatherv_init(const void* sendbuf, int sendcount,
5         MPI_Datatype sendtype, void* recvbuf, const int recvcnts[],
6         const int displs[], MPI_Datatype recvtype, MPI_Comm comm,
7         MPI_Info info, MPI_Request *request)
8
9     int MPI_Neighbor_alltoall(const void* sendbuf, int sendcount,
10        MPI_Datatype sendtype, void* recvbuf, int recvcount,
11        MPI_Datatype recvtype, MPI_Comm comm)
12
13    int MPI_Neighbor_alltoall_init(const void* sendbuf, int sendcount,
14        MPI_Datatype sendtype, void* recvbuf, int recvcount,
15        MPI_Datatype recvtype, MPI_Comm comm, MPI_Info info,
16        MPI_Request *request)
17
18    int MPI_Neighbor_alltoallv(const void* sendbuf, const int sendcounts[],
19        const int sdispls[], MPI_Datatype sendtype, void* recvbuf,
20        const int recvcnts[], const int rdispls[],
21        MPI_Datatype recvtype, MPI_Comm comm)
22
23    int MPI_Neighbor_alltoallv_init(const void* sendbuf,
24        const int sendcounts[], const int sdispls[],
25        MPI_Datatype sendtype, void* recvbuf, const int recvcnts[],
26        const int rdispls[], MPI_Datatype recvtype, MPI_Comm comm,
27        MPI_Info info, MPI_Request *request)
28
29    int MPI_Neighbor_alltoallw(const void* sendbuf, const int sendcounts[],
30        const MPI_Aint sdispls[], const MPI_Datatype sendtypes[],
31        void* recvbuf, const int recvcnts[],
32        const MPI_Aint rdispls[], const MPI_Datatype recvtypes[],
33        MPI_Comm comm)
34
35    int MPI_Neighbor_alltoallw_init(const void* sendbuf,
36        const int sendcounts[], const MPI_Aint sdispls[],
37        const MPI_Datatype sendtypes[], void* recvbuf,
38        const int recvcnts[], const MPI_Aint rdispls[],
39        const MPI_Datatype recvtypes[], MPI_Comm comm, MPI_Info info,
40        MPI_Request *request)
41
42    int MPI_Topo_test(MPI_Comm comm, int *status)

```

A.2.6 MPI Environmental Management C Bindings

```

42    double MPI_Wtick(void)
43
44    double MPI_Wtime(void)
45
46    int MPI_Abort(MPI_Comm comm, int errorcode)
47
48    int MPI_Add_error_class(int *errorclass)

```

```

int MPI_Add_error_code(int errorclass, int *errorcode) 1
int MPI_Add_error_string(int errorcode, const char *string) 2
int MPI_Alloc_mem(MPI_Aint size, MPI_Info info, void *baseptr) 3
int MPI_Comm_call_errhandler(MPI_Comm comm, int errorcode) 4
int MPI_Comm_create_errhandler(MPI_Comm_errhandler_function *comm_errhandler_fn, 5
MPI_Errhandler *errhandler) 6
int MPI_Comm_get_errhandler(MPI_Comm comm, MPI_Errhandler *errhandler) 7
int MPI_Comm_set_errhandler(MPI_Comm comm, MPI_Errhandler errhandler) 8
int MPI_Errhandler_free(MPI_Errhandler *errhandler) 9
int MPI_Error_class(int errorcode, int *errorclass) 10
int MPI_Error_string(int errorcode, char *string, int *resultlen) 11
int MPI_File_call_errhandler(MPI_File fh, int errorcode) 12
int MPI_File_create_errhandler(MPI_File_errhandler_function *file_errhandler_fn, 13
MPI_Errhandler *errhandler) 14
int MPI_File_get_errhandler(MPI_File file, MPI_Errhandler *errhandler) 15
int MPI_File_set_errhandler(MPI_File file, MPI_Errhandler errhandler) 16
int MPI_Finalize(void) 17
int MPI_Finalized(int *flag) 18
int MPI_Free_mem(void *base) 19
int MPI_Get_library_version(char *version, int *resultlen) 20
int MPI_Get_processor_name(char *name, int *resultlen) 21
int MPI_Get_version(int *version, int *subversion) 22
int MPI_Init(int *argc, char ***argv) 23
int MPI_Initialized(int *flag) 24
int MPI_Win_call_errhandler(MPI_Win win, int errorcode) 25
int MPI_Win_create_errhandler(MPI_Win_errhandler_function *win_errhandler_fn, 26
MPI_Errhandler *errhandler) 27
int MPI_Win_get_errhandler(MPI_Win win, MPI_Errhandler *errhandler) 28
int MPI_Win_set_errhandler(MPI_Win win, MPI_Errhandler errhandler) 29

```

A.2.7 The Info Object C Bindings

```
1 int MPI_Info_create(MPI_Info *info)
2
3 int MPI_Info_delete(MPI_Info info, const char *key)
4
5 int MPI_Info_dup(MPI_Info info, MPI_Info *newinfo)
6
7 int MPI_Info_free(MPI_Info *info)
8
9 int MPI_Info_get(MPI_Info info, const char *key, int valuelen, char *value,
10                 int *flag)
11
12 int MPI_Info_get_nkeys(MPI_Info info, int *nkeys)
13
14 int MPI_Info_get_nthkey(MPI_Info info, int n, char *key)
15
16 int MPI_Info_get_valuelen(MPI_Info info, const char *key, int *valuelen,
17                           int *flag)
18
19 int MPI_Info_set(MPI_Info info, const char *key, const char *value)
```

A.2.8 Process Creation and Management C Bindings

```
21 int MPI_Close_port(const char *port_name)
22
23 int MPI_Comm_accept(const char *port_name, MPI_Info info, int root,
24                   MPI_Comm comm, MPI_Comm *newcomm)
25
26 int MPI_Comm_connect(const char *port_name, MPI_Info info, int root,
27                    MPI_Comm comm, MPI_Comm *newcomm)
28
29 int MPI_Comm_disconnect(MPI_Comm *comm)
30
31 int MPI_Comm_get_parent(MPI_Comm *parent)
32
33 int MPI_Comm_join(int fd, MPI_Comm *intercomm)
34
35 int MPI_Comm_spawn(const char *command, char *argv[], int maxprocs,
36                  MPI_Info info, int root, MPI_Comm comm, MPI_Comm *intercomm,
37                  int array_of_errcodes[])
38
39 int MPI_Comm_spawn_multiple(int count, char *array_of_commands[],
40                            char **array_of_argv[], const int array_of_maxprocs[],
41                            const MPI_Info array_of_info[], int root, MPI_Comm comm,
42                            MPI_Comm *intercomm, int array_of_errcodes[])
43
44 int MPI_Lookup_name(const char *service_name, MPI_Info info,
45                  char *port_name)
46
47 int MPI_Open_port(MPI_Info info, char *port_name)
48
49 int MPI_Publish_name(const char *service_name, MPI_Info info,
50                   const char *port_name)
51
52 int MPI_Unpublish_name(const char *service_name, MPI_Info info,
53                      const char *port_name)
```

A.2.9 One-Sided Communications C Bindings

```
int MPI_Accumulate(const void *origin_addr, int origin_count,
                  MPI_Datatype origin_datatype, int target_rank,
                  MPI_Aint target_disp, int target_count,
                  MPI_Datatype target_datatype, MPI_Op op, MPI_Win win)

int MPI_Compare_and_swap(const void *origin_addr, const void *compare_addr,
                        void *result_addr, MPI_Datatype datatype, int target_rank,
                        MPI_Aint target_disp, MPI_Win win)

int MPI_Fetch_and_op(const void *origin_addr, void *result_addr,
                    MPI_Datatype datatype, int target_rank, MPI_Aint target_disp,
                    MPI_Op op, MPI_Win win)

int MPI_Get(void *origin_addr, int origin_count,
            MPI_Datatype origin_datatype, int target_rank,
            MPI_Aint target_disp, int target_count,
            MPI_Datatype target_datatype, MPI_Win win)

int MPI_Get_accumulate(const void *origin_addr, int origin_count,
                      MPI_Datatype origin_datatype, void *result_addr,
                      int result_count, MPI_Datatype result_datatype,
                      int target_rank, MPI_Aint target_disp, int target_count,
                      MPI_Datatype target_datatype, MPI_Op op, MPI_Win win)

int MPI_Put(const void *origin_addr, int origin_count,
            MPI_Datatype origin_datatype, int target_rank,
            MPI_Aint target_disp, int target_count,
            MPI_Datatype target_datatype, MPI_Win win)

int MPI_Raccumulate(const void *origin_addr, int origin_count,
                   MPI_Datatype origin_datatype, int target_rank,
                   MPI_Aint target_disp, int target_count,
                   MPI_Datatype target_datatype, MPI_Op op, MPI_Win win,
                   MPI_Request *request)

int MPI_Rget(void *origin_addr, int origin_count,
             MPI_Datatype origin_datatype, int target_rank,
             MPI_Aint target_disp, int target_count,
             MPI_Datatype target_datatype, MPI_Win win,
             MPI_Request *request)

int MPI_Rget_accumulate(const void *origin_addr, int origin_count,
                       MPI_Datatype origin_datatype, void *result_addr,
                       int result_count, MPI_Datatype result_datatype,
                       int target_rank, MPI_Aint target_disp, int target_count,
                       MPI_Datatype target_datatype, MPI_Op op, MPI_Win win,
                       MPI_Request *request)

int MPI_Rput(const void *origin_addr, int origin_count,
             MPI_Datatype origin_datatype, int target_rank,
```

```
1         MPI_Aint target_disp, int target_count,
2         MPI_Datatype target_datatype, MPI_Win win,
3         MPI_Request *request)
4
5     int MPI_Win_allocate(MPI_Aint size, int disp_unit, MPI_Info info,
6         MPI_Comm comm, void *baseptr, MPI_Win *win)
7
8     int MPI_Win_allocate_shared(MPI_Aint size, int disp_unit, MPI_Info info,
9         MPI_Comm comm, void *baseptr, MPI_Win *win)
10
11    int MPI_Win_attach(MPI_Win win, void *base, MPI_Aint size)
12
13    int MPI_Win_complete(MPI_Win win)
14
15    int MPI_Win_create(void *base, MPI_Aint size, int disp_unit, MPI_Info info,
16        MPI_Comm comm, MPI_Win *win)
17
18    int MPI_Win_create_dynamic(MPI_Info info, MPI_Comm comm, MPI_Win *win)
19
20    int MPI_Win_detach(MPI_Win win, const void *base)
21
22    int MPI_Win_fence(int assert, MPI_Win win)
23
24    int MPI_Win_flush(int rank, MPI_Win win)
25
26    int MPI_Win_flush_all(MPI_Win win)
27
28    int MPI_Win_flush_local(int rank, MPI_Win win)
29
30    int MPI_Win_flush_local_all(MPI_Win win)
31
32    int MPI_Win_free(MPI_Win *win)
33
34    int MPI_Win_get_group(MPI_Win win, MPI_Group *group)
35
36    int MPI_Win_get_info(MPI_Win win, MPI_Info *info_used)
37
38    int MPI_Win_lock(int lock_type, int rank, int assert, MPI_Win win)
39
40    int MPI_Win_lock_all(int assert, MPI_Win win)
41
42    int MPI_Win_post(MPI_Group group, int assert, MPI_Win win)
43
44    int MPI_Win_set_info(MPI_Win win, MPI_Info info)
45
46    int MPI_Win_shared_query(MPI_Win win, int rank, MPI_Aint *size,
47        int *disp_unit, void *baseptr)
48
49    int MPI_Win_start(MPI_Group group, int assert, MPI_Win win)
50
51    int MPI_Win_sync(MPI_Win win)
52
53    int MPI_Win_test(MPI_Win win, int *flag)
54
55    int MPI_Win_unlock(int rank, MPI_Win win)
56
57    int MPI_Win_unlock_all(MPI_Win win)
58
59    int MPI_Win_wait(MPI_Win win)
```

A.2.10 External Interfaces C Bindings

```
int MPI_Grequest_complete(MPI_Request request)
int MPI_Grequest_start(MPI_Grequest_query_function *query_fn,
    MPI_Grequest_free_function *free_fn,
    MPI_Grequest_cancel_function *cancel_fn, void *extra_state,
    MPI_Request *request)
int MPI_Init_thread(int *argc, char ***argv, int required, int *provided)
int MPI_Is_thread_main(int *flag)
int MPI_Query_thread(int *provided)
int MPI_Status_set_cancelled(MPI_Status *status, int flag)
int MPI_Status_set_elements(MPI_Status *status, MPI_Datatype datatype,
    MPI_Count count)
int MPI_Status_set_elements(MPI_Status *status, MPI_Datatype datatype,
    int count)
int MPI_Status_set_elements_x(MPI_Status *status, MPI_Datatype datatype,
    MPI_Count count)
```

A.2.11 I/O C Bindings

```
int MPI_CONVERSION_FN_NULL(void *userbuf, MPI_Datatype datatype, int count,
    void *filebuf, MPI_Offset position, void *extra_state)
int MPI_File_close(MPI_File *fh)
int MPI_File_delete(const char *filename, MPI_Info info)
int MPI_File_get_amode(MPI_File fh, int *amode)
int MPI_File_get_atomicity(MPI_File fh, int *flag)
int MPI_File_get_byte_offset(MPI_File fh, MPI_Offset offset,
    MPI_Offset *disp)
int MPI_File_get_group(MPI_File fh, MPI_Group *group)
int MPI_File_get_info(MPI_File fh, MPI_Info *info_used)
int MPI_File_get_position(MPI_File fh, MPI_Offset *offset)
int MPI_File_get_position_shared(MPI_File fh, MPI_Offset *offset)
int MPI_File_get_size(MPI_File fh, MPI_Offset *size)
int MPI_File_get_type_extent(MPI_File fh, MPI_Datatype datatype,
    MPI_Aint *extent)
int MPI_File_get_view(MPI_File fh, MPI_Offset *disp, MPI_Datatype *etype,
    MPI_Datatype *filetype, char *datarep)
```

```
1 int MPI_File_iread(MPI_File fh, void *buf, int count,
2     MPI_Datatype datatype, MPI_Request *request)
3
4 int MPI_File_iread_all(MPI_File fh, void *buf, int count,
5     MPI_Datatype datatype, MPI_Request *request)
6
7 int MPI_File_iread_at(MPI_File fh, MPI_Offset offset, void *buf, int count,
8     MPI_Datatype datatype, MPI_Request *request)
9
10 int MPI_File_iread_at_all(MPI_File fh, MPI_Offset offset, void *buf,
11     int count, MPI_Datatype datatype, MPI_Request *request)
12
13 int MPI_File_iread_shared(MPI_File fh, void *buf, int count,
14     MPI_Datatype datatype, MPI_Request *request)
15
16 int MPI_File_iread_shared(MPI_File fh, void *buf, int count,
17     MPI_Datatype datatype, MPI_Request *request)
18
19 int MPI_File_iread_shared(MPI_File fh, void *buf, int count,
20     MPI_Datatype datatype, MPI_Request *request)
21
22 int MPI_File_iread_shared(MPI_File fh, void *buf, int count,
23     MPI_Datatype datatype, MPI_Request *request)
24
25 int MPI_File_iread_shared(MPI_File fh, void *buf, int count,
26     MPI_Datatype datatype, MPI_Request *request)
27
28 int MPI_File_iread_shared(MPI_File fh, void *buf, int count,
29     MPI_Datatype datatype, MPI_Request *request)
30
31 int MPI_File_iread_shared(MPI_File fh, void *buf, int count,
32     MPI_Datatype datatype, MPI_Request *request)
33
34 int MPI_File_iread_shared(MPI_File fh, void *buf, int count,
35     MPI_Datatype datatype, MPI_Request *request)
36
37 int MPI_File_iread_shared(MPI_File fh, void *buf, int count,
38     MPI_Datatype datatype, MPI_Request *request)
39
40 int MPI_File_iread_shared(MPI_File fh, void *buf, int count,
41     MPI_Datatype datatype, MPI_Request *request)
42
43 int MPI_File_iread_shared(MPI_File fh, void *buf, int count,
44     MPI_Datatype datatype, MPI_Request *request)
45
46 int MPI_File_iread_shared(MPI_File fh, void *buf, int count,
47     MPI_Datatype datatype, MPI_Request *request)
48
```



```
int MPI_File_read_ordered(MPI_File fh, void *buf, int count,      1
                          MPI_Datatype datatype, MPI_Status *status) 2
                                                                    3
int MPI_File_read_ordered_begin(MPI_File fh, void *buf, int count, 4
                               MPI_Datatype datatype)                5
                                                                    6
int MPI_File_read_ordered_end(MPI_File fh, void *buf, MPI_Status *status) 7
                                                                    8
int MPI_File_read_shared(MPI_File fh, void *buf, int count,      8
                        MPI_Datatype datatype, MPI_Status *status) 9
                                                                    10
int MPI_File_seek(MPI_File fh, MPI_Offset offset, int whence)    11
                                                                    12
int MPI_File_seek_shared(MPI_File fh, MPI_Offset offset, int whence) 12
                                                                    13
int MPI_File_set_atomicity(MPI_File fh, int flag)                14
                                                                    15
int MPI_File_set_info(MPI_File fh, MPI_Info info)                15
                                                                    16
int MPI_File_set_size(MPI_File fh, MPI_Offset size)              17
                                                                    18
int MPI_File_set_view(MPI_File fh, MPI_Offset disp, MPI_Datatype etype, 18
                     MPI_Datatype filetype, const char *datarep, MPI_Info info) 19
                                                                    20
int MPI_File_sync(MPI_File fh)                                    21
                                                                    22
int MPI_File_write(MPI_File fh, const void *buf, int count,      22
                  MPI_Datatype datatype, MPI_Status *status)    23
                                                                    24
int MPI_File_write_all(MPI_File fh, const void *buf, int count,  24
                      MPI_Datatype datatype, MPI_Status *status) 25
                                                                    26
int MPI_File_write_all_begin(MPI_File fh, const void *buf, int count, 27
                             MPI_Datatype datatype)              28
                                                                    29
int MPI_File_write_all_end(MPI_File fh, const void *buf,         30
                          MPI_Status *status)                    31
                                                                    32
int MPI_File_write_at(MPI_File fh, MPI_Offset offset, const void *buf, 32
                    int count, MPI_Datatype datatype, MPI_Status *status) 33
                                                                    34
int MPI_File_write_at_all(MPI_File fh, MPI_Offset offset, const void *buf, 35
                        int count, MPI_Datatype datatype, MPI_Status *status) 36
                                                                    37
int MPI_File_write_at_all_begin(MPI_File fh, MPI_Offset offset,  37
                               const void *buf, int count, MPI_Datatype datatype) 38
                                                                    39
int MPI_File_write_at_all_end(MPI_File fh, const void *buf,     40
                              MPI_Status *status)                41
                                                                    42
int MPI_File_write_ordered(MPI_File fh, const void *buf, int count, 42
                          MPI_Datatype datatype, MPI_Status *status) 43
                                                                    44
int MPI_File_write_ordered_begin(MPI_File fh, const void *buf, int count, 45
                                MPI_Datatype datatype)            46
                                                                    47
int MPI_File_write_ordered_end(MPI_File fh, const void *buf,     47
                              MPI_Status *status)                48
```

```

1         MPI_Status *status)
2
3     int MPI_File_write_shared(MPI_File fh, const void *buf, int count,
4         MPI_Datatype datatype, MPI_Status *status)
5
6     int MPI_Register_datarep(const char *datarep,
7         MPI_Datarep_conversion_function *read_conversion_fn,
8         MPI_Datarep_conversion_function *write_conversion_fn,
9         MPI_Datarep_extent_function *dtype_file_extent_fn,
10        void *extra_state)

```

A.2.12 Language Bindings C Bindings

```

13     int MPI_Status_f082f(MPI_F08_status *f08_status, MPI_Fint *f_status)
14
15     int MPI_Status_f2f08(MPI_Fint *f_status, MPI_F08_status *f08_status)
16
17     int MPI_Type_create_f90_complex(int p, int r, MPI_Datatype *newtype)
18
19     int MPI_Type_create_f90_integer(int r, MPI_Datatype *newtype)
20
21     int MPI_Type_create_f90_real(int p, int r, MPI_Datatype *newtype)
22
23     int MPI_Type_match_size(int typeclass, int size, MPI_Datatype *datatype)
24
25     MPI_Fint MPI_Comm_c2f(MPI_Comm comm)
26
27     MPI_Comm MPI_Comm_f2c(MPI_Fint comm)
28
29     MPI_Fint MPI_Errhandler_c2f(MPI_Errhandler errhandler)
30
31     MPI_Errhandler MPI_Errhandler_f2c(MPI_Fint errhandler)
32
33     MPI_Fint MPI_File_c2f(MPI_File file)
34
35     MPI_File MPI_File_f2c(MPI_Fint file)
36
37     MPI_Fint MPI_Group_c2f(MPI_Group group)
38
39     MPI_Group MPI_Group_f2c(MPI_Fint group)
40
41     MPI_Fint MPI_Info_c2f(MPI_Info info)
42
43     MPI_Info MPI_Info_f2c(MPI_Fint info)
44
45     MPI_Fint MPI_Message_c2f(MPI_Message message)
46
47     MPI_Message MPI_Message_f2c(MPI_Fint message)
48
49     MPI_Fint MPI_Op_c2f(MPI_Op op)
50
51     MPI_Op MPI_Op_f2c(MPI_Fint op)
52
53     MPI_Fint MPI_Request_c2f(MPI_Request request)
54
55     MPI_Request MPI_Request_f2c(MPI_Fint request)
56
57     int MPI_Status_c2f(const MPI_Status *c_status, MPI_Fint *f_status)

```

```

int MPI_Status_c2f08(const MPI_Status *c_status,           1
                    MPI_F08_status *f08_status)           2
int MPI_Status_f082c(const MPI_F08_status *f08_status,    3
                    MPI_Status *c_status)                 4
int MPI_Status_f2c(const MPI_Fint *f_status, MPI_Status *c_status) 6
MPI_Fint MPI_Type_c2f(MPI_Datatype datatype)             7
MPI_Datatype MPI_Type_f2c(MPI_Fint datatype)             8
MPI_Fint MPI_Win_c2f(MPI_Win win)                        9
MPI_Win MPI_Win_f2c(MPI_Fint win)                       10

```

A.2.13 Tools / Profiling Interface C Bindings

```

int MPI_Pcontrol(const int level, ...)                    15

```

A.2.14 Tools / MPI Tool Information Interface C Bindings

```

int MPI_T_category_changed(int *stamp)                   16
int MPI_T_category_get_categories(int cat_index, int len, int indices[]) 17
int MPI_T_category_get_cvars(int cat_index, int len, int indices[]) 18
int MPI_T_category_get_index(const char *name, int *cat_index) 19
int MPI_T_category_get_info(int cat_index, char *name, int *name_len, 20
                            char *desc, int *desc_len, int *num_cvars, int *num_pvars, 21
                            int *num_categories)          22
int MPI_T_category_get_num(int *num_cat)                 23
int MPI_T_category_get_pvars(int cat_index, int len, int indices[]) 24
int MPI_T_cvar_get_index(const char *name, int *cvar_index) 25
int MPI_T_cvar_get_info(int cvar_index, char *name, int *name_len, 26
                        int *verbosity, MPI_Datatype *datatype, MPI_T_enum *enumtype, 27
                        char *desc, int *desc_len, int *bind, int *scope) 28
int MPI_T_cvar_get_num(int *num_cvar)                    29
int MPI_T_cvar_handle_alloc(int cvar_index, void *obj_handle, 30
                            MPI_T_cvar_handle *handle, int *count) 31
int MPI_T_cvar_handle_free(MPI_T_cvar_handle *handle)    32
int MPI_T_cvar_read(MPI_T_cvar_handle handle, void* buf) 33
int MPI_T_cvar_write(MPI_T_cvar_handle handle, const void* buf) 34
int MPI_T_enum_get_info(MPI_T_enum enumtype, int *num, char *name, 35
                        int *name_len)                   36

```

```

1  int MPI_T_enum_get_item(MPI_T_enum enumtype, int index, int *value,
2      char *name, int *name_len)
3
4  int MPI_T_finalize(void)
5
6  int MPI_T_init_thread(int required, int *provided)
7
8  int MPI_T_pvar_get_index(const char *name, int var_class, int *pvar_index)
9
10 int MPI_T_pvar_get_info(int pvar_index, char *name, int *name_len,
11     int *verbosity, int *var_class, MPI_Datatype *datatype,
12     MPI_T_enum *enumtype, char *desc, int *desc_len, int *bind,
13     int *readonly, int *continuous, int *atomic)
14
15 int MPI_T_pvar_get_num(int *num_pvar)
16
17 int MPI_T_pvar_handle_alloc(MPI_T_pvar_session session, int pvar_index,
18     void *obj_handle, MPI_T_pvar_handle *handle, int *count)
19
20 int MPI_T_pvar_handle_free(MPI_T_pvar_session session,
21     MPI_T_pvar_handle *handle)
22
23 int MPI_T_pvar_read(MPI_T_pvar_session session, MPI_T_pvar_handle handle,
24     void* buf)
25
26 int MPI_T_pvar_readreset(MPI_T_pvar_session session,
27     MPI_T_pvar_handle handle, void* buf)
28
29 int MPI_T_pvar_reset(MPI_T_pvar_session session, MPI_T_pvar_handle handle)
30
31 int MPI_T_pvar_session_create(MPI_T_pvar_session *session)
32
33 int MPI_T_pvar_session_free(MPI_T_pvar_session *session)
34
35 int MPI_T_pvar_start(MPI_T_pvar_session session, MPI_T_pvar_handle handle)
36
37 int MPI_T_pvar_stop(MPI_T_pvar_session session, MPI_T_pvar_handle handle)
38
39 int MPI_T_pvar_write(MPI_T_pvar_session session, MPI_T_pvar_handle handle,
40     const void* buf)
41
42
43
44
45
46
47
48

```

A.2.15 Deprecated C Bindings

```

37 int MPI_Attr_delete(MPI_Comm comm, int keyval)
38
39 int MPI_Attr_get(MPI_Comm comm, int keyval, void *attribute_val, int *flag)
40
41 int MPI_Attr_put(MPI_Comm comm, int keyval, void* attribute_val)
42
43 int MPI_DUP_FN(MPI_Comm oldcomm, int keyval, void *extra_state,
44     void *attribute_val_in, void *attribute_val_out, int *flag)
45
46 int MPI_Keyval_create(MPI_Copy_function *copy_fn,
47     MPI_Delete_function *delete_fn, int *keyval,
48     void* extra_state)
49
50 int MPI_Keyval_free(int *keyval)

```

```
int MPI_NULL_COPY_FN(MPI_Comm oldcomm, int keyval, void *extra_state,
                    void *attribute_val_in, void *attribute_val_out, int *flag)
int MPI_NULL_DELETE_FN(MPI_Comm comm, int keyval, void *attribute_val,
                      void *extra_state)
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48

1 A.3 Fortran 2008 Bindings with the mpi_f08 Module

2 A.3.1 Point-to-Point Communication Fortran 2008 Bindings

3 MPI_Bsend(buf, count, datatype, dest, tag, comm, ierror)

4 TYPE(*), DIMENSION(..), INTENT(IN) :: buf
5 INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count
6 TYPE(MPI_Datatype), INTENT(IN) :: datatype
7 INTEGER, INTENT(IN) :: dest, tag
8 TYPE(MPI_Comm), INTENT(IN) :: comm
9 INTEGER, OPTIONAL, INTENT(OUT) :: ierror

10 MPI_Bsend(buf, count, datatype, dest, tag, comm, ierror)

11 TYPE(*), DIMENSION(..), INTENT(IN) :: buf
12 INTEGER, INTENT(IN) :: count, dest, tag
13 TYPE(MPI_Datatype), INTENT(IN) :: datatype
14 TYPE(MPI_Comm), INTENT(IN) :: comm
15 INTEGER, OPTIONAL, INTENT(OUT) :: ierror

16 MPI_Bsend_init(buf, count, datatype, dest, tag, comm, request, ierror)

17 TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf
18 INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count
19 TYPE(MPI_Datatype), INTENT(IN) :: datatype
20 INTEGER, INTENT(IN) :: dest, tag
21 TYPE(MPI_Comm), INTENT(IN) :: comm
22 TYPE(MPI_Request), INTENT(OUT) :: request
23 INTEGER, OPTIONAL, INTENT(OUT) :: ierror

24 MPI_Bsend_init(buf, count, datatype, dest, tag, comm, request, ierror)

25 TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf
26 INTEGER, INTENT(IN) :: count, dest, tag
27 TYPE(MPI_Datatype), INTENT(IN) :: datatype
28 TYPE(MPI_Comm), INTENT(IN) :: comm
29 TYPE(MPI_Request), INTENT(OUT) :: request
30 INTEGER, OPTIONAL, INTENT(OUT) :: ierror

31 MPI_Buffer_attach(buffer, size, ierror)

32 TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buffer
33 INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: size
34 INTEGER, OPTIONAL, INTENT(OUT) :: ierror

35 MPI_Buffer_attach(buffer, size, ierror)

36 TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buffer
37 INTEGER, INTENT(IN) :: size
38 INTEGER, OPTIONAL, INTENT(OUT) :: ierror

39 MPI_Buffer_detach(buffer_addr, size, ierror)

40 USE, INTRINSIC :: ISO_C_BINDING, ONLY : C_PTR
41 TYPE(C_PTR), INTENT(OUT) :: buffer_addr
42 INTEGER(KIND=MPI_COUNT_KIND), INTENT(OUT) :: size
43 INTEGER, OPTIONAL, INTENT(OUT) :: ierror

```

MPI_Buffer_detach(buffer_addr, size, ierror) 1
  USE, INTRINSIC :: ISO_C_BINDING, ONLY : C_PTR 2
  TYPE(C_PTR), INTENT(OUT) :: buffer_addr 3
  INTEGER, INTENT(OUT) :: size 4
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror 5
6
MPI_Cancel(request, ierror) 7
  TYPE(MPI_Request), INTENT(IN) :: request 8
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror 9
10
MPI_Get_count(status, datatype, count, ierror) 11
  TYPE(MPI_Status), INTENT(IN) :: status 12
  TYPE(MPI_Datatype), INTENT(IN) :: datatype 13
  INTEGER(KIND=MPI_COUNT_KIND), INTENT(OUT) :: count 14
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror 15
16
MPI_Get_count(status, datatype, count, ierror) 17
  TYPE(MPI_Status), INTENT(IN) :: status 18
  TYPE(MPI_Datatype), INTENT(IN) :: datatype 19
  INTEGER, INTENT(OUT) :: count 20
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror 21
22
MPI_Ibsend(buf, count, datatype, dest, tag, comm, request, ierror) 23
  TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf 24
  INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count 25
  TYPE(MPI_Datatype), INTENT(IN) :: datatype 26
  INTEGER, INTENT(IN) :: dest, tag 27
  TYPE(MPI_Comm), INTENT(IN) :: comm 28
  TYPE(MPI_Request), INTENT(OUT) :: request 29
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror 30
31
MPI_Ibsend(buf, count, datatype, dest, tag, comm, request, ierror) 32
  TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf 33
  INTEGER, INTENT(IN) :: count, dest, tag 34
  TYPE(MPI_Datatype), INTENT(IN) :: datatype 35
  TYPE(MPI_Comm), INTENT(IN) :: comm 36
  TYPE(MPI_Request), INTENT(OUT) :: request 37
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror 38
39
MPI_Iprobe(source, tag, comm, flag, message, status, ierror) 40
  INTEGER, INTENT(IN) :: source, tag 41
  TYPE(MPI_Comm), INTENT(IN) :: comm 42
  LOGICAL, INTENT(OUT) :: flag 43
  TYPE(MPI_Message), INTENT(OUT) :: message 44
  TYPE(MPI_Status) :: status 45
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror 46
47
MPI_Imrecv(buf, count, datatype, message, request, ierror) 48
  TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf 49
  INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count 50
  TYPE(MPI_Datatype), INTENT(IN) :: datatype 51

```

```

1     TYPE(MPI_Message), INTENT(INOUT) :: message
2     TYPE(MPI_Request), INTENT(OUT) :: request
3     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
4
5     MPI_Imrecv(buf, count, datatype, message, request, ierror)
6     TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf
7     INTEGER, INTENT(IN) :: count
8     TYPE(MPI_Datatype), INTENT(IN) :: datatype
9     TYPE(MPI_Message), INTENT(INOUT) :: message
10    TYPE(MPI_Request), INTENT(OUT) :: request
11    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
12
13    MPI_Iprobe(source, tag, comm, flag, status, ierror)
14    INTEGER, INTENT(IN) :: source, tag
15    TYPE(MPI_Comm), INTENT(IN) :: comm
16    LOGICAL, INTENT(OUT) :: flag
17    TYPE(MPI_Status) :: status
18    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
19
20    MPI_Irecv(buf, count, datatype, source, tag, comm, request, ierror)
21    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf
22    INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count
23    TYPE(MPI_Datatype), INTENT(IN) :: datatype
24    INTEGER, INTENT(IN) :: source, tag
25    TYPE(MPI_Comm), INTENT(IN) :: comm
26    TYPE(MPI_Request), INTENT(OUT) :: request
27    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
28
29    MPI_Irecv(buf, count, datatype, source, tag, comm, request, ierror)
30    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf
31    INTEGER, INTENT(IN) :: count, source, tag
32    TYPE(MPI_Datatype), INTENT(IN) :: datatype
33    TYPE(MPI_Comm), INTENT(IN) :: comm
34    TYPE(MPI_Request), INTENT(OUT) :: request
35    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
36
37    MPI_Irsend(buf, count, datatype, dest, tag, comm, request, ierror)
38    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf
39    INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count
40    TYPE(MPI_Datatype), INTENT(IN) :: datatype
41    INTEGER, INTENT(IN) :: dest, tag
42    TYPE(MPI_Comm), INTENT(IN) :: comm
43    TYPE(MPI_Request), INTENT(OUT) :: request
44    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
45
46    MPI_Irsend(buf, count, datatype, dest, tag, comm, request, ierror)
47    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf
48    INTEGER, INTENT(IN) :: count, dest, tag
49    TYPE(MPI_Datatype), INTENT(IN) :: datatype
50    TYPE(MPI_Comm), INTENT(IN) :: comm

```



```

TYPE(MPI_Request), INTENT(OUT) :: request      1
INTEGER, OPTIONAL, INTENT(OUT) :: ierror      2
                                           3
MPI_Isend(buf, count, datatype, dest, tag, comm, request, ierror)  4
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf          5
INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count                 6
TYPE(MPI_Datatype), INTENT(IN) :: datatype                       7
INTEGER, INTENT(IN) :: dest, tag                                 8
TYPE(MPI_Comm), INTENT(IN) :: comm                              9
TYPE(MPI_Request), INTENT(OUT) :: request                       10
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                        11
                                           12
MPI_Isend(buf, count, datatype, dest, tag, comm, request, ierror) 13
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf          14
INTEGER, INTENT(IN) :: count, dest, tag                          15
TYPE(MPI_Datatype), INTENT(IN) :: datatype                       16
TYPE(MPI_Comm), INTENT(IN) :: comm                              17
TYPE(MPI_Request), INTENT(OUT) :: request                       18
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                        19
                                           20
MPI_Issend(buf, count, datatype, dest, tag, comm, request, ierror) 21
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf          22
INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count                 23
TYPE(MPI_Datatype), INTENT(IN) :: datatype                       24
INTEGER, INTENT(IN) :: dest, tag                                 25
TYPE(MPI_Comm), INTENT(IN) :: comm                              26
TYPE(MPI_Request), INTENT(OUT) :: request                       27
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                        28
                                           29
MPI_Issend(buf, count, datatype, dest, tag, comm, request, ierror) 30
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf          31
INTEGER, INTENT(IN) :: count, dest, tag                          32
TYPE(MPI_Datatype), INTENT(IN) :: datatype                       33
TYPE(MPI_Comm), INTENT(IN) :: comm                              34
TYPE(MPI_Request), INTENT(OUT) :: request                       35
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                        36
                                           37
MPI_Mprobe(source, tag, comm, message, status, ierror)            38
INTEGER, INTENT(IN) :: source, tag                               39
TYPE(MPI_Comm), INTENT(IN) :: comm                              40
TYPE(MPI_Message), INTENT(OUT) :: message                       41
TYPE(MPI_Status) :: status                                       42
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                        43
                                           44
MPI_Mrecv(buf, count, datatype, message, status, ierror)          45
TYPE(*), DIMENSION(..) :: buf                                    46
INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count                 47
TYPE(MPI_Datatype), INTENT(IN) :: datatype                       48
TYPE(MPI_Message), INTENT(INOUT) :: message                     49
TYPE(MPI_Status) :: status                                       50

```

```

1     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
2
3     MPI_Mrecv(buf, count, datatype, message, status, ierror)
4         TYPE(*), DIMENSION(..) :: buf
5         INTEGER, INTENT(IN) :: count
6         TYPE(MPI_Datatype), INTENT(IN) :: datatype
7         TYPE(MPI_Message), INTENT(INOUT) :: message
8         TYPE(MPI_Status) :: status
9         INTEGER, OPTIONAL, INTENT(OUT) :: ierror
10
11    MPI_Probe(source, tag, comm, status, ierror)
12        INTEGER, INTENT(IN) :: source, tag
13        TYPE(MPI_Comm), INTENT(IN) :: comm
14        TYPE(MPI_Status) :: status
15        INTEGER, OPTIONAL, INTENT(OUT) :: ierror
16
17    MPI_Recv(buf, count, datatype, source, tag, comm, status, ierror)
18        TYPE(*), DIMENSION(..) :: buf
19        INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count
20        TYPE(MPI_Datatype), INTENT(IN) :: datatype
21        INTEGER, INTENT(IN) :: source, tag
22        TYPE(MPI_Comm), INTENT(IN) :: comm
23        TYPE(MPI_Status) :: status
24        INTEGER, OPTIONAL, INTENT(OUT) :: ierror
25
26    MPI_Recv(buf, count, datatype, source, tag, comm, status, ierror)
27        TYPE(*), DIMENSION(..) :: buf
28        INTEGER, INTENT(IN) :: count, source, tag
29        TYPE(MPI_Datatype), INTENT(IN) :: datatype
30        TYPE(MPI_Comm), INTENT(IN) :: comm
31        TYPE(MPI_Status) :: status
32        INTEGER, OPTIONAL, INTENT(OUT) :: ierror
33
34    MPI_Recv_init(buf, count, datatype, dest, tag, comm, request, ierror)
35        TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf
36        INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count
37        TYPE(MPI_Datatype), INTENT(IN) :: datatype
38        INTEGER, INTENT(IN) :: dest, tag
39        TYPE(MPI_Comm), INTENT(IN) :: comm
40        TYPE(MPI_Request), INTENT(OUT) :: request
41        INTEGER, OPTIONAL, INTENT(OUT) :: ierror
42
43    MPI_Recv_init(buf, count, datatype, dest, tag, comm, request, ierror)
44        TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf
45        INTEGER, INTENT(IN) :: count, dest, tag
46        TYPE(MPI_Datatype), INTENT(IN) :: datatype
47        TYPE(MPI_Comm), INTENT(IN) :: comm
48        TYPE(MPI_Request), INTENT(OUT) :: request
49        INTEGER, OPTIONAL, INTENT(OUT) :: ierror
50
51    MPI_Request_free(request, ierror)

```

```

TYPE(MPI_Request), INTENT(INOUT) :: request      1
INTEGER, OPTIONAL, INTENT(OUT) :: ierror      2
MPI_Request_get_status(request, flag, status, ierror) 3
TYPE(MPI_Request), INTENT(IN) :: request      4
LOGICAL, INTENT(OUT) :: flag                  5
TYPE(MPI_Status) :: status                    6
INTEGER, OPTIONAL, INTENT(OUT) :: ierror      7
MPI_Rsend(buf, count, datatype, dest, tag, comm, ierror) 8
TYPE(*), DIMENSION(..), INTENT(IN) :: buf    9
INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count 10
TYPE(MPI_Datatype), INTENT(IN) :: datatype   11
INTEGER, INTENT(IN) :: dest, tag             12
TYPE(MPI_Comm), INTENT(IN) :: comm           13
INTEGER, OPTIONAL, INTENT(OUT) :: ierror     14
MPI_Rsend(buf, count, datatype, dest, tag, comm, ierror) 15
TYPE(*), DIMENSION(..), INTENT(IN) :: buf    16
INTEGER, INTENT(IN) :: count, dest, tag      17
TYPE(MPI_Datatype), INTENT(IN) :: datatype   18
TYPE(MPI_Comm), INTENT(IN) :: comm           19
INTEGER, OPTIONAL, INTENT(OUT) :: ierror     20
MPI_Rsend_init(buf, count, datatype, dest, tag, comm, request, ierror) 21
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf 22
INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count 23
TYPE(MPI_Datatype), INTENT(IN) :: datatype   24
INTEGER, INTENT(IN) :: dest, tag             25
TYPE(MPI_Comm), INTENT(IN) :: comm           26
TYPE(MPI_Request), INTENT(OUT) :: request    27
INTEGER, OPTIONAL, INTENT(OUT) :: ierror     28
MPI_Rsend_init(buf, count, datatype, dest, tag, comm, request, ierror) 29
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf 30
INTEGER, INTENT(IN) :: count, dest, tag      31
TYPE(MPI_Datatype), INTENT(IN) :: datatype   32
TYPE(MPI_Comm), INTENT(IN) :: comm           33
TYPE(MPI_Request), INTENT(OUT) :: request    34
INTEGER, OPTIONAL, INTENT(OUT) :: ierror     35
MPI_Send(buf, count, datatype, dest, tag, comm, ierror) 36
TYPE(*), DIMENSION(..), INTENT(IN) :: buf    37
INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count 38
TYPE(MPI_Datatype), INTENT(IN) :: datatype   39
INTEGER, INTENT(IN) :: dest, tag             40
TYPE(MPI_Comm), INTENT(IN) :: comm           41
INTEGER, OPTIONAL, INTENT(OUT) :: ierror     42
MPI_Send(buf, count, datatype, dest, tag, comm, ierror) 43
TYPE(*), DIMENSION(..), INTENT(IN) :: buf    44

```

```

1     INTEGER, INTENT(IN) :: count, dest, tag
2     TYPE(MPI_Datatype), INTENT(IN) :: datatype
3     TYPE(MPI_Comm), INTENT(IN) :: comm
4     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
5
6 MPI_Send_init(buf, count, datatype, dest, tag, comm, request, ierror)
7     TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf
8     INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count
9     TYPE(MPI_Datatype), INTENT(IN) :: datatype
10    INTEGER, INTENT(IN) :: dest, tag
11    TYPE(MPI_Comm), INTENT(IN) :: comm
12    TYPE(MPI_Request), INTENT(OUT) :: request
13    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
14
15 MPI_Send_init(buf, count, datatype, dest, tag, comm, request, ierror)
16    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf
17    INTEGER, INTENT(IN) :: count, dest, tag
18    TYPE(MPI_Datatype), INTENT(IN) :: datatype
19    TYPE(MPI_Comm), INTENT(IN) :: comm
20    TYPE(MPI_Request), INTENT(OUT) :: request
21    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
22
23 MPI_Sendrecv(sendbuf, sendcount, sendtype, dest, sendtag, recvbuf,
24             recvcount, recvtype, source, recvtag, comm, status, ierror)
25    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
26    INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: sendcount, recvcount
27    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
28    INTEGER, INTENT(IN) :: dest, sendtag, source, recvtag
29    TYPE(*), DIMENSION(..) :: recvbuf
30    TYPE(MPI_Comm), INTENT(IN) :: comm
31    TYPE(MPI_Status) :: status
32    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
33
34 MPI_Sendrecv(sendbuf, sendcount, sendtype, dest, sendtag, recvbuf,
35             recvcount, recvtype, source, recvtag, comm, status, ierror)
36    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
37    INTEGER, INTENT(IN) :: sendcount, dest, sendtag, recvcount, source,
38    recvtag
39    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
40    TYPE(*), DIMENSION(..) :: recvbuf
41    TYPE(MPI_Comm), INTENT(IN) :: comm
42    TYPE(MPI_Status) :: status
43    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
44
45 MPI_Sendrecv_replace(buf, count, datatype, dest, sendtag, source, recvtag,
46                    comm, status, ierror)
47    TYPE(*), DIMENSION(..) :: buf
48    INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count
49    TYPE(MPI_Datatype), INTENT(IN) :: datatype
50    INTEGER, INTENT(IN) :: dest, sendtag, source, recvtag

```

```

TYPE(MPI_Comm), INTENT(IN) :: comm
TYPE(MPI_Status) :: status
INTEGER, OPTIONAL, INTENT(OUT) :: ierror
MPI_Sendrecv_replace(buf, count, datatype, dest, sendtag, source, recvtag,
comm, status, ierror)
TYPE(*), DIMENSION(..) :: buf
INTEGER, INTENT(IN) :: count, dest, sendtag, source, recvtag
TYPE(MPI_Datatype), INTENT(IN) :: datatype
TYPE(MPI_Comm), INTENT(IN) :: comm
TYPE(MPI_Status) :: status
INTEGER, OPTIONAL, INTENT(OUT) :: ierror
MPI_Ssend(buf, count, datatype, dest, tag, comm, ierror)
TYPE(*), DIMENSION(..), INTENT(IN) :: buf
INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count
TYPE(MPI_Datatype), INTENT(IN) :: datatype
INTEGER, INTENT(IN) :: dest, tag
TYPE(MPI_Comm), INTENT(IN) :: comm
INTEGER, OPTIONAL, INTENT(OUT) :: ierror
MPI_Ssend(buf, count, datatype, dest, tag, comm, ierror)
TYPE(*), DIMENSION(..), INTENT(IN) :: buf
INTEGER, INTENT(IN) :: count, dest, tag
TYPE(MPI_Datatype), INTENT(IN) :: datatype
TYPE(MPI_Comm), INTENT(IN) :: comm
INTEGER, OPTIONAL, INTENT(OUT) :: ierror
MPI_Ssend_init(buf, count, datatype, dest, tag, comm, request, ierror)
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf
INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count
TYPE(MPI_Datatype), INTENT(IN) :: datatype
INTEGER, INTENT(IN) :: dest, tag
TYPE(MPI_Comm), INTENT(IN) :: comm
TYPE(MPI_Request), INTENT(OUT) :: request
INTEGER, OPTIONAL, INTENT(OUT) :: ierror
MPI_Ssend_init(buf, count, datatype, dest, tag, comm, request, ierror)
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf
INTEGER, INTENT(IN) :: count, dest, tag
TYPE(MPI_Datatype), INTENT(IN) :: datatype
TYPE(MPI_Comm), INTENT(IN) :: comm
TYPE(MPI_Request), INTENT(OUT) :: request
INTEGER, OPTIONAL, INTENT(OUT) :: ierror
MPI_Start(request, ierror)
TYPE(MPI_Request), INTENT(INOUT) :: request
INTEGER, OPTIONAL, INTENT(OUT) :: ierror
MPI_Startall(count, array_of_requests, ierror)
INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count

```

```

1     TYPE(MPI_Request), INTENT(INOUT) :: array_of_requests(count)
2     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
3
4     MPI_Startall(count, array_of_requests, ierror)
5     INTEGER, INTENT(IN) :: count
6     TYPE(MPI_Request), INTENT(INOUT) :: array_of_requests(count)
7     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
8
9     MPI_Test(request, flag, status, ierror)
10    TYPE(MPI_Request), INTENT(INOUT) :: request
11    LOGICAL, INTENT(OUT) :: flag
12    TYPE(MPI_Status) :: status
13    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
14
15    MPI_Test_cancelled(status, flag, ierror)
16    TYPE(MPI_Status), INTENT(IN) :: status
17    LOGICAL, INTENT(OUT) :: flag
18    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
19
20    MPI_Testall(count, array_of_requests, flag, array_of_statuses, ierror)
21    INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count
22    TYPE(MPI_Request), INTENT(INOUT) :: array_of_requests(count)
23    LOGICAL, INTENT(OUT) :: flag
24    TYPE(MPI_Status) :: array_of_statuses(*)
25    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
26
27    MPI_Testall(count, array_of_requests, flag, array_of_statuses, ierror)
28    INTEGER, INTENT(IN) :: count
29    TYPE(MPI_Request), INTENT(INOUT) :: array_of_requests(count)
30    LOGICAL, INTENT(OUT) :: flag
31    TYPE(MPI_Status) :: array_of_statuses(*)
32    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
33
34    MPI_Testany(count, array_of_requests, index, flag, status, ierror)
35    INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count
36    TYPE(MPI_Request), INTENT(INOUT) :: array_of_requests(count)
37    INTEGER, INTENT(OUT) :: index
38    LOGICAL, INTENT(OUT) :: flag
39    TYPE(MPI_Status) :: status
40    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
41
42    MPI_Testany(count, array_of_requests, index, flag, status, ierror)
43    INTEGER, INTENT(IN) :: count
44    TYPE(MPI_Request), INTENT(INOUT) :: array_of_requests(count)
45    INTEGER, INTENT(OUT) :: index
46    LOGICAL, INTENT(OUT) :: flag
47    TYPE(MPI_Status) :: status
48    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
49
50    MPI_Testsome(incount, array_of_requests, outcount, array_of_indices,
51                array_of_statuses, ierror)

```

```

INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: incount           1
TYPE(MPI_Request), INTENT(INOUT) :: array_of_requests(count)  2
INTEGER(KIND=MPI_COUNT_KIND), INTENT(OUT) :: outcount        3
INTEGER, INTENT(OUT) :: array_of_indices(*)                  4
TYPE(MPI_Status) :: array_of_statuses(*)                     5
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                     6
MPI_Testsome(incount, array_of_requests, outcount, array_of_indices,  7
              array_of_statuses, ierror)                      8
INTEGER, INTENT(IN) :: incount                               9
TYPE(MPI_Request), INTENT(INOUT) :: array_of_requests(count) 10
INTEGER, INTENT(OUT) :: outcount                            11
INTEGER, INTENT(OUT) :: array_of_indices(*)                  12
TYPE(MPI_Status) :: array_of_statuses(*)                     13
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                     14
MPI_Wait(request, status, ierror)                            15
TYPE(MPI_Request), INTENT(INOUT) :: request                  16
TYPE(MPI_Status) :: status                                   17
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                     18
MPI_Waitall(count, array_of_requests, array_of_statuses, ierror) 19
INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count            20
TYPE(MPI_Request), INTENT(INOUT) :: array_of_requests(count) 21
TYPE(MPI_Status) :: array_of_statuses(*)                     22
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                     23
MPI_Waitall(count, array_of_requests, array_of_statuses, ierror) 24
INTEGER, INTENT(IN) :: count                                 25
TYPE(MPI_Request), INTENT(INOUT) :: array_of_requests(count) 26
TYPE(MPI_Status) :: array_of_statuses(*)                     27
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                     28
MPI_Waitany(count, array_of_requests, index, status, ierror) 29
INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count            30
TYPE(MPI_Request), INTENT(INOUT) :: array_of_requests(count) 31
INTEGER, INTENT(OUT) :: index                                 32
TYPE(MPI_Status) :: status                                   33
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                     34
MPI_Waitany(count, array_of_requests, index, status, ierror) 35
INTEGER, INTENT(IN) :: count                                 36
TYPE(MPI_Request), INTENT(INOUT) :: array_of_requests(count) 37
INTEGER, INTENT(OUT) :: index                                 38
TYPE(MPI_Status) :: status                                   39
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                     40
MPI_Waitsome(incount, array_of_requests, outcount, array_of_indices, 41
              array_of_statuses, ierror)                      42
INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: incount           43
TYPE(MPI_Request), INTENT(INOUT) :: array_of_requests(count) 44

```

```

1      INTEGER(KIND=MPI_COUNT_KIND), INTENT(OUT) :: outcount
2      INTEGER, INTENT(OUT) :: array_of_indices(*)
3      TYPE(MPI_Status) :: array_of_statuses(*)
4      INTEGER, OPTIONAL, INTENT(OUT) :: ierror
5
6      MPI_Waitsome(incount, array_of_requests, outcount, array_of_indices,
7                  array_of_statuses, ierror)
8      INTEGER, INTENT(IN) :: incount
9      TYPE(MPI_Request), INTENT(INOUT) :: array_of_requests(count)
10     INTEGER, INTENT(OUT) :: outcount
11     INTEGER, INTENT(OUT) :: array_of_indices(*)
12     TYPE(MPI_Status) :: array_of_statuses(*)
13     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
14
15     A.3.2 Datatypes Fortran 2008 Bindings
16
17     INTEGER(KIND=MPI_ADDRESS_KIND) MPI_Aint_add(base, disp)
18         INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: base, disp
19
20     INTEGER(KIND=MPI_ADDRESS_KIND) MPI_Aint_diff(addr1, addr2)
21         INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: addr1, addr2
22
23     MPI_Get_address(location, address, ierror)
24         TYPE(*), DIMENSION(..), ASYNCHRONOUS :: location
25         INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(OUT) :: address
26         INTEGER, OPTIONAL, INTENT(OUT) :: ierror
27
28     MPI_Get_elements(status, datatype, count, ierror)
29         TYPE(MPI_Status), INTENT(IN) :: status
30         TYPE(MPI_Datatype), INTENT(IN) :: datatype
31         INTEGER, INTENT(OUT) :: count
32         INTEGER, OPTIONAL, INTENT(OUT) :: ierror
33
34     MPI_Get_elements_x(status, datatype, count, ierror)
35         TYPE(MPI_Status), INTENT(IN) :: status
36         TYPE(MPI_Datatype), INTENT(IN) :: datatype
37         INTEGER(KIND=MPI_COUNT_KIND), INTENT(OUT) :: count
38         INTEGER, OPTIONAL, INTENT(OUT) :: ierror
39
40     MPI_Pack(inbuf, incount, datatype, outbuf, outsize, position, comm, ierror)
41         TYPE(*), DIMENSION(..), INTENT(IN) :: inbuf
42         TYPE(*), DIMENSION(..) :: outbuf
43         INTEGER, INTENT(IN) :: incount, outsize
44         TYPE(MPI_Datatype), INTENT(IN) :: datatype
45         INTEGER, INTENT(INOUT) :: position
46         TYPE(MPI_Comm), INTENT(IN) :: comm
47         INTEGER, OPTIONAL, INTENT(OUT) :: ierror
48
49     MPI_Pack_external(datarep, inbuf, incount, datatype, outbuf, outsize,
50                     position, ierror)
51         CHARACTER(LEN=*), INTENT(IN) :: datarep

```



```

TYPE(*), DIMENSION(..), INTENT(IN) :: inbuf           1
TYPE(*), DIMENSION(..) :: outbuf                     2
INTEGER, INTENT(IN) :: incount                       3
TYPE(MPI_Datatype), INTENT(IN) :: datatype           4
INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: outsize 5
INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(INOUT) :: position 6
INTEGER, OPTIONAL, INTENT(OUT) :: ierror            7
                                                    8
MPI_Pack_external_size(datarep, incount, datatype, size, ierror) 9
TYPE(MPI_Datatype), INTENT(IN) :: datatype           10
INTEGER, INTENT(IN) :: incount                       11
CHARACTER(LEN=*), INTENT(IN) :: datarep             12
INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(OUT) :: size  13
INTEGER, OPTIONAL, INTENT(OUT) :: ierror            14
                                                    15
MPI_Pack_size(incount, datatype, comm, size, ierror) 16
INTEGER, INTENT(IN) :: incount                       17
TYPE(MPI_Datatype), INTENT(IN) :: datatype           18
TYPE(MPI_Comm), INTENT(IN) :: comm                  19
INTEGER, INTENT(OUT) :: size                         20
INTEGER, OPTIONAL, INTENT(OUT) :: ierror            21
                                                    22
MPI_Type_commit(datatype, ierror)                    23
TYPE(MPI_Datatype), INTENT(INOUT) :: datatype        24
INTEGER, OPTIONAL, INTENT(OUT) :: ierror            25
                                                    26
MPI_Type_contiguous(count, oldtype, newtype, ierror) 27
INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count    28
TYPE(MPI_Datatype), INTENT(IN) :: oldtype            29
TYPE(MPI_Datatype), INTENT(OUT) :: newtype           30
INTEGER, OPTIONAL, INTENT(OUT) :: ierror            31
                                                    32
MPI_Type_contiguous(count, oldtype, newtype, ierror) 33
INTEGER, INTENT(IN) :: count                         34
TYPE(MPI_Datatype), INTENT(IN) :: oldtype            35
TYPE(MPI_Datatype), INTENT(OUT) :: newtype           36
INTEGER, OPTIONAL, INTENT(OUT) :: ierror            37
                                                    38
MPI_Type_contiguous(count, oldtype, newtype, ierror) 39
INTEGER, INTENT(IN) :: count                         40
TYPE(MPI_Datatype), INTENT(IN) :: oldtype            41
TYPE(MPI_Datatype), INTENT(OUT) :: newtype           42
INTEGER, OPTIONAL, INTENT(OUT) :: ierror            43
                                                    44
MPI_Type_create_darray(size, rank, ndims, array_of_gsizes,
array_of_distribs, array_of_dargs, array_of_psize, order,
oldtype, newtype, ierror)                            45
INTEGER, INTENT(IN) :: size, rank, ndims, array_of_gsizes(ndims),
array_of_distribs(ndims), array_of_dargs(ndims),
array_of_psize(ndims), order                        46
TYPE(MPI_Datatype), INTENT(IN) :: oldtype            47
                                                    48

```

```

1     TYPE(MPI_Datatype), INTENT(OUT) :: newtype
2     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
3
4     MPI_Type_create_hindexed(count, array_of_blocklengths,
5                             array_of_displacements, oldtype, newtype, ierror)
6     INTEGER, INTENT(IN) :: count, array_of_blocklengths(count)
7     INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) ::
8         array_of_displacements(count)
9     TYPE(MPI_Datatype), INTENT(IN) :: oldtype
10    TYPE(MPI_Datatype), INTENT(OUT) :: newtype
11    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
12
13    MPI_Type_create_hindexed_block(count, blocklength, array_of_displacements,
14                                   oldtype, newtype, ierror)
15    INTEGER, INTENT(IN) :: count, blocklength
16    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) ::
17        array_of_displacements(count)
18    TYPE(MPI_Datatype), INTENT(IN) :: oldtype
19    TYPE(MPI_Datatype), INTENT(OUT) :: newtype
20    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
21
22    MPI_Type_create_hvector(count, blocklength, stride, oldtype, newtype,
23                            ierror)
24    INTEGER, INTENT(IN) :: count, blocklength
25    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: stride
26    TYPE(MPI_Datatype), INTENT(IN) :: oldtype
27    TYPE(MPI_Datatype), INTENT(OUT) :: newtype
28    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
29
30    MPI_Type_create_indexed_block(count, blocklength, array_of_displacements,
31                                   oldtype, newtype, ierror)
32    INTEGER, INTENT(IN) :: count, blocklength,
33        array_of_displacements(count)
34    TYPE(MPI_Datatype), INTENT(IN) :: oldtype
35    TYPE(MPI_Datatype), INTENT(OUT) :: newtype
36    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
37
38    MPI_Type_create_resized(oldtype, lb, extent, newtype, ierror)
39    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: lb, extent
40    TYPE(MPI_Datatype), INTENT(IN) :: oldtype
41    TYPE(MPI_Datatype), INTENT(OUT) :: newtype
42    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
43
44    MPI_Type_create_struct(count, array_of_blocklengths,
45                            array_of_displacements, array_of_types, newtype, ierror)
46    INTEGER, INTENT(IN) :: count, array_of_blocklengths(count)
47    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) ::
48        array_of_displacements(count)
49    TYPE(MPI_Datatype), INTENT(IN) :: array_of_types(count)
50    TYPE(MPI_Datatype), INTENT(OUT) :: newtype

```

```

INTEGER, OPTIONAL, INTENT(OUT) :: ierror 1
MPI_Type_create_subarray(ndims, array_of_sizes, array_of_subsizes, 2
    array_of_starts, order, oldtype, newtype, ierror) 3
INTEGER, INTENT(IN) :: ndims, array_of_sizes(ndims), 4
    array_of_subsizes(ndims), array_of_starts(ndims), order 5
TYPE(MPI_Datatype), INTENT(IN) :: oldtype 6
TYPE(MPI_Datatype), INTENT(OUT) :: newtype 7
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 8
MPI_Type_dup(oldtype, newtype, ierror) 9
TYPE(MPI_Datatype), INTENT(IN) :: oldtype 10
TYPE(MPI_Datatype), INTENT(OUT) :: newtype 11
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 12
MPI_Type_free(datatype, ierror) 13
TYPE(MPI_Datatype), INTENT(INOUT) :: datatype 14
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 15
MPI_Type_get_contents(datatype, max_integers, max_addresses, max_datatypes, 16
    array_of_integers, array_of_addresses, array_of_datatypes, 17
    ierror) 18
TYPE(MPI_Datatype), INTENT(IN) :: datatype 19
INTEGER, INTENT(IN) :: max_integers, max_addresses, max_datatypes 20
INTEGER, INTENT(OUT) :: array_of_integers(max_integers) 21
INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(OUT) :: 22
    array_of_addresses(max_addresses) 23
TYPE(MPI_Datatype), INTENT(OUT) :: array_of_datatypes(max_datatypes) 24
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 25
MPI_Type_get_envelope(datatype, num_integers, num_addresses, num_datatypes, 26
    combiner, ierror) 27
TYPE(MPI_Datatype), INTENT(IN) :: datatype 28
INTEGER, INTENT(OUT) :: num_integers, num_addresses, num_datatypes, 29
    combiner 30
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 31
MPI_Type_get_extent(datatype, lb, extent, ierror) 32
TYPE(MPI_Datatype), INTENT(IN) :: datatype 33
INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(OUT) :: lb, extent 34
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 35
MPI_Type_get_extent_x(datatype, lb, extent, ierror) 36
TYPE(MPI_Datatype), INTENT(IN) :: datatype 37
INTEGER(KIND=MPI_COUNT_KIND), INTENT(OUT) :: lb, extent 38
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 39
MPI_Type_get_true_extent(datatype, true_lb, true_extent, ierror) 40
TYPE(MPI_Datatype), INTENT(IN) :: datatype 41
INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(OUT) :: true_lb, true_extent 42
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 43

```

```

1 MPI_Type_get_true_extent_x(datatype, true_lb, true_extent, ierror)
2   TYPE(MPI_Datatype), INTENT(IN) :: datatype
3   INTEGER(KIND=MPI_COUNT_KIND), INTENT(OUT) :: true_lb, true_extent
4   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
5
6 MPI_Type_indexed(count, array_of_blocklengths, array_of_displacements,
7   oldtype, newtype, ierror)
8   INTEGER, INTENT(IN) :: count, array_of_blocklengths(count),
9   array_of_displacements(count)
10  TYPE(MPI_Datatype), INTENT(IN) :: oldtype
11  TYPE(MPI_Datatype), INTENT(OUT) :: newtype
12  INTEGER, OPTIONAL, INTENT(OUT) :: ierror
13
14 MPI_Type_size(datatype, size, ierror)
15  TYPE(MPI_Datatype), INTENT(IN) :: datatype
16  INTEGER, INTENT(OUT) :: size
17  INTEGER, OPTIONAL, INTENT(OUT) :: ierror
18
19 MPI_Type_size_x(datatype, size, ierror)
20  TYPE(MPI_Datatype), INTENT(IN) :: datatype
21  INTEGER(KIND=MPI_COUNT_KIND), INTENT(OUT) :: size
22  INTEGER, OPTIONAL, INTENT(OUT) :: ierror
23
24 MPI_Type_vector(count, blocklength, stride, oldtype, newtype, ierror)
25  INTEGER, INTENT(IN) :: count, blocklength, stride
26  TYPE(MPI_Datatype), INTENT(IN) :: oldtype
27  TYPE(MPI_Datatype), INTENT(OUT) :: newtype
28  INTEGER, OPTIONAL, INTENT(OUT) :: ierror
29
30 MPI_Unpack(inbuf, insize, position, outbuf, outcount, datatype, comm,
31  ierror)
32  TYPE(*), DIMENSION(..), INTENT(IN) :: inbuf
33  TYPE(*), DIMENSION(..) :: outbuf
34  INTEGER, INTENT(IN) :: insize, outcount
35  INTEGER, INTENT(INOUT) :: position
36  TYPE(MPI_Datatype), INTENT(IN) :: datatype
37  TYPE(MPI_Comm), INTENT(IN) :: comm
38  INTEGER, OPTIONAL, INTENT(OUT) :: ierror
39
40 MPI_Unpack_external(datarep, inbuf, insize, position, outbuf, outcount,
41  datatype, ierror)
42  CHARACTER(LEN=*), INTENT(IN) :: datarep
43  TYPE(*), DIMENSION(..), INTENT(IN) :: inbuf
44  TYPE(*), DIMENSION(..) :: outbuf
45  INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: insize
46  INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(INOUT) :: position
47  INTEGER, INTENT(IN) :: outcount
48  TYPE(MPI_Datatype), INTENT(IN) :: datatype
49  INTEGER, OPTIONAL, INTENT(OUT) :: ierror

```

A.3.3 Collective Communication Fortran 2008 Bindings

```

MPI_Allgather(sendbuf, sendcount, sendtype, recvbuf, recvcount, recvtype,
              comm, ierror)
    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
    TYPE(*), DIMENSION(..) :: recvbuf
    INTEGER, INTENT(IN) :: sendcount, recvcount
    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
    TYPE(MPI_Comm), INTENT(IN) :: comm
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Allgather_init(sendbuf, sendcount, sendtype, recvbuf, recvcount,
                  recvtype, comm, info, request, ierror)
    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
    INTEGER, INTENT(IN) :: sendcount, recvcount
    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
    TYPE(MPI_Comm), INTENT(IN) :: comm
    TYPE(MPI_Info), INTENT(IN) :: info
    TYPE(MPI_Request), INTENT(OUT) :: request
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Allgatherv(sendbuf, sendcount, sendtype, recvbuf, recvcounts, displs,
              recvtype, comm, ierror)
    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
    TYPE(*), DIMENSION(..) :: recvbuf
    INTEGER, INTENT(IN) :: sendcount, recvcounts(*), displs(*)
    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
    TYPE(MPI_Comm), INTENT(IN) :: comm
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Allgatherv_init(sendbuf, sendcount, sendtype, recvbuf, recvcounts,
                  displs, recvtype, comm, info, request, ierror)
    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
    INTEGER, INTENT(IN) :: sendcount
    INTEGER, INTENT(IN), ASYNCHRONOUS :: recvcounts(*), displs(*)
    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
    TYPE(MPI_Comm), INTENT(IN) :: comm
    TYPE(MPI_Info), INTENT(IN) :: info
    TYPE(MPI_Request), INTENT(OUT) :: request
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Allreduce(sendbuf, recvbuf, count, datatype, op, comm, ierror)
    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
    TYPE(*), DIMENSION(..) :: recvbuf
    INTEGER, INTENT(IN) :: count
    TYPE(MPI_Datatype), INTENT(IN) :: datatype
    TYPE(MPI_Op), INTENT(IN) :: op
    TYPE(MPI_Comm), INTENT(IN) :: comm

```

```

1     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
2
3     MPI_Allreduce_init(sendbuf, recvbuf, count, datatype, op, comm, info,
4         request, ierror)
5     TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
6     TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
7     INTEGER, INTENT(IN) :: count
8     TYPE(MPI_Datatype), INTENT(IN) :: datatype
9     TYPE(MPI_Op), INTENT(IN) :: op
10    TYPE(MPI_Comm), INTENT(IN) :: comm
11    TYPE(MPI_Info), INTENT(IN) :: info
12    TYPE(MPI_Request), INTENT(OUT) :: request
13    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
14
15    MPI_Alltoall(sendbuf, sendcount, sendtype, recvbuf, recvcount, recvtype,
16        comm, ierror)
17    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
18    TYPE(*), DIMENSION(..) :: recvbuf
19    INTEGER, INTENT(IN) :: sendcount, recvcount
20    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
21    TYPE(MPI_Comm), INTENT(IN) :: comm
22    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
23
24    MPI_Alltoall_init(sendbuf, sendcount, sendtype, recvbuf, recvcount,
25        recvtype, comm, info, request, ierror)
26    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
27    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
28    INTEGER, INTENT(IN) :: sendcount, recvcount
29    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
30    TYPE(MPI_Comm), INTENT(IN) :: comm
31    TYPE(MPI_Info), INTENT(IN) :: info
32    TYPE(MPI_Request), INTENT(OUT) :: request
33    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
34
35    MPI_Alltoallv(sendbuf, sendcounts, sdispls, sendtype, recvbuf, recvcounts,
36        rdispls, recvtype, comm, ierror)
37    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
38    TYPE(*), DIMENSION(..) :: recvbuf
39    INTEGER, INTENT(IN) :: sendcounts(*), sdispls(*), recvcounts(*),
40        rdispls(*)
41    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
42    TYPE(MPI_Comm), INTENT(IN) :: comm
43    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
44
45    MPI_Alltoallv_init(sendbuf, sendcounts, sdispls, sendtype, recvbuf,
46        recvcounts, rdispls, recvtype, comm, info, request, ierror)
47    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
48    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
49    INTEGER, INTENT(IN), ASYNCHRONOUS :: sendcounts(*), sdispls(*),
50        recvcounts(*), rdispls(*)

```

```

TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype      1
TYPE(MPI_Comm), INTENT(IN) :: comm                        2
TYPE(MPI_Info), INTENT(IN) :: info                       3
TYPE(MPI_Request), INTENT(OUT) :: request                4
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                 5
                                                           6
MPI_Alltoallw(sendbuf, sendcounts, sdispls, sendtypes, recvbuf, recvcounts,
              rdispls, recvtypes, comm, ierror)           7
TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf           8
TYPE(*), DIMENSION(..) :: recvbuf                       9
INTEGER, INTENT(IN) :: sendcounts(*), sdispls(*), recvcounts(*),
              rdispls(*)                                 10
TYPE(MPI_Datatype), INTENT(IN) :: sendtypes(*)          11
TYPE(MPI_Datatype), INTENT(IN) :: recvtypes(*)          12
TYPE(MPI_Comm), INTENT(IN) :: comm                      13
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                 14
                                                           15
MPI_Alltoallw_init(sendbuf, sendcounts, sdispls, sendtypes, recvbuf,
                  recvcounts, rdispls, recvtypes, comm, info, request, ierror) 16
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf 17
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf           18
INTEGER, INTENT(IN), ASYNCHRONOUS :: sendcounts(*), sdispls(*),
              recvcounts(*), rdispls(*)                   19
TYPE(MPI_Datatype), INTENT(IN), ASYNCHRONOUS :: sendtypes(*),
              recvtypes(*)                                 20
TYPE(MPI_Comm), INTENT(IN) :: comm                       21
TYPE(MPI_Info), INTENT(IN) :: info                       22
TYPE(MPI_Request), INTENT(OUT) :: request                23
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                 24
                                                           25
MPI_Barrier(comm, ierror)                                 26
TYPE(MPI_Comm), INTENT(IN) :: comm                       27
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                 28
                                                           29
MPI_Barrier_init(comm, info, request, ierror)             30
TYPE(MPI_Comm), INTENT(IN) :: comm                       31
TYPE(MPI_Info), INTENT(IN) :: info                       32
TYPE(MPI_Request), INTENT(OUT) :: request                33
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                 34
                                                           35
MPI_Barrier_init(comm, info, request, ierror)             36
TYPE(MPI_Comm), INTENT(IN) :: comm                       37
TYPE(MPI_Info), INTENT(IN) :: info                       38
TYPE(MPI_Request), INTENT(OUT) :: request                39
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                 40
                                                           41
MPI_Bcast(buffer, count, datatype, root, comm, ierror)    42
TYPE(*), DIMENSION(..) :: buffer                        43
INTEGER, INTENT(IN) :: count, root                       44
TYPE(MPI_Datatype), INTENT(IN) :: datatype              45
TYPE(MPI_Comm), INTENT(IN) :: comm                       46
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                 47
                                                           48
MPI_Bcast_init(buffer, count, datatype, root, comm, info, request, ierror)
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buffer
INTEGER, INTENT(IN) :: count, root

```

```

1     TYPE(MPI_Datatype), INTENT(IN) :: datatype
2     TYPE(MPI_Comm), INTENT(IN) :: comm
3     TYPE(MPI_Info), INTENT(IN) :: info
4     TYPE(MPI_Request), INTENT(OUT) :: request
5     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
6
7 MPI_Exscan(sendbuf, recvbuf, count, datatype, op, comm, ierror)
8     TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
9     TYPE(*), DIMENSION(..) :: recvbuf
10    INTEGER, INTENT(IN) :: count
11    TYPE(MPI_Datatype), INTENT(IN) :: datatype
12    TYPE(MPI_Op), INTENT(IN) :: op
13    TYPE(MPI_Comm), INTENT(IN) :: comm
14    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
15
16 MPI_Exscan_init(sendbuf, recvbuf, count, datatype, op, comm, info, request,
17                ierror)
18    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
19    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
20    INTEGER, INTENT(IN) :: count
21    TYPE(MPI_Datatype), INTENT(IN) :: datatype
22    TYPE(MPI_Op), INTENT(IN) :: op
23    TYPE(MPI_Comm), INTENT(IN) :: comm
24    TYPE(MPI_Info), INTENT(IN) :: info
25    TYPE(MPI_Request), INTENT(OUT) :: request
26    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
27
28 MPI_Gather(sendbuf, sendcount, sendtype, recvbuf, recvcount, recvttype,
29           root, comm, ierror)
30    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
31    TYPE(*), DIMENSION(..) :: recvbuf
32    INTEGER, INTENT(IN) :: sendcount, recvcount, root
33    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvttype
34    TYPE(MPI_Comm), INTENT(IN) :: comm
35    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
36
37 MPI_Gather_init(sendbuf, sendcount, sendtype, recvbuf, recvcount, recvttype,
38                root, comm, info, request, ierror)
39    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
40    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
41    INTEGER, INTENT(IN) :: sendcount, recvcount, root
42    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvttype
43    TYPE(MPI_Comm), INTENT(IN) :: comm
44    TYPE(MPI_Info), INTENT(IN) :: info
45    TYPE(MPI_Request), INTENT(OUT) :: request
46    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
47
48 MPI_Gatherv(sendbuf, sendcount, sendtype, recvbuf, recvcounts, displs,
49            recvttype, root, comm, ierror)
50    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf

```



```

TYPE(*), DIMENSION(..) :: recvbuf                                1
INTEGER, INTENT(IN) :: sendcount, recvcnts(*), displs(*), root  2
TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype            3
TYPE(MPI_Comm), INTENT(IN) :: comm                              4
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                        5
                                                                    6
MPI_Gatherv_init(sendbuf, sendcount, sendtype, recvbuf, recvcnts, displs,
                 recvtype, root, comm, info, request, ierror)  7
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf    8
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf                9
INTEGER, INTENT(IN) :: sendcount, root                       10
INTEGER, INTENT(IN), ASYNCHRONOUS :: recvcnts(*), displs(*)   11
TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype          12
TYPE(MPI_Comm), INTENT(IN) :: comm                            13
TYPE(MPI_Info), INTENT(IN) :: info                            14
TYPE(MPI_Request), INTENT(OUT) :: request                     15
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       16
                                                                    17
MPI_Iallgather(sendbuf, sendcount, sendtype, recvbuf, recvcnt, recvtype,
               comm, request, ierror)                           18
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf   19
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf               20
INTEGER, INTENT(IN) :: sendcount, recvcnt                      21
TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype          22
TYPE(MPI_Comm), INTENT(IN) :: comm                             23
TYPE(MPI_Request), INTENT(OUT) :: request                      24
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       25
                                                                    26
MPI_Iallgatherv(sendbuf, sendcount, sendtype, recvbuf, recvcnts, displs,
                recvtype, comm, request, ierror)                27
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf   28
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf               29
INTEGER, INTENT(IN) :: sendcount                               30
INTEGER, INTENT(IN), ASYNCHRONOUS :: recvcnts(*), displs(*)   31
TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype          32
TYPE(MPI_Comm), INTENT(IN) :: comm                             33
TYPE(MPI_Request), INTENT(OUT) :: request                      34
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       35
                                                                    36
MPI_Iallreduce(sendbuf, recvbuf, count, datatype, op, comm, request,
               ierror)                                          37
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf   38
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf               39
INTEGER, INTENT(IN) :: count                                    40
TYPE(MPI_Datatype), INTENT(IN) :: datatype                     41
TYPE(MPI_Op), INTENT(IN) :: op                                  42
TYPE(MPI_Comm), INTENT(IN) :: comm                             43
TYPE(MPI_Request), INTENT(OUT) :: request                      44
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       45
                                                                    46
MPI_Iallreduce(sendbuf, recvbuf, count, datatype, op, comm, request,
               ierror)                                          47
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf   48
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf               49
INTEGER, INTENT(IN) :: count                                    50
TYPE(MPI_Datatype), INTENT(IN) :: datatype                     51
TYPE(MPI_Op), INTENT(IN) :: op                                  52
TYPE(MPI_Comm), INTENT(IN) :: comm                             53
TYPE(MPI_Request), INTENT(OUT) :: request                      54
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       55
                                                                    56

```

```

1 MPI_Ialltoall(sendbuf, sendcount, sendtype, recvbuf, recvcount, recvtype,
2               comm, request, ierror)
3     TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
4     TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
5     INTEGER, INTENT(IN) :: sendcount, recvcount
6     TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
7     TYPE(MPI_Comm), INTENT(IN) :: comm
8     TYPE(MPI_Request), INTENT(OUT) :: request
9     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
10
11 MPI_Ialltoallv(sendbuf, sendcounts, sdispls, sendtype, recvbuf, recvcounts,
12               rdispls, recvtype, comm, request, ierror)
13     TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
14     TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
15     INTEGER, INTENT(IN), ASYNCHRONOUS :: sendcounts(*), sdispls(*),
16               recvcounts(*), rdispls(*)
17     TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
18     TYPE(MPI_Comm), INTENT(IN) :: comm
19     TYPE(MPI_Request), INTENT(OUT) :: request
20     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
21
22 MPI_Ialltoallw(sendbuf, sendcounts, sdispls, sendtypes, recvbuf,
23               recvcounts, rdispls, recvtypes, comm, request, ierror)
24     TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
25     TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
26     INTEGER, INTENT(IN), ASYNCHRONOUS :: sendcounts(*), sdispls(*),
27               recvcounts(*), rdispls(*)
28     TYPE(MPI_Datatype), INTENT(IN), ASYNCHRONOUS :: sendtypes(*),
29               recvtypes(*)
30     TYPE(MPI_Comm), INTENT(IN) :: comm
31     TYPE(MPI_Request), INTENT(OUT) :: request
32     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
33
34 MPI_Ibcast(comm, request, ierror)
35     TYPE(MPI_Comm), INTENT(IN) :: comm
36     TYPE(MPI_Request), INTENT(OUT) :: request
37     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
38
39 MPI_Ibcast(buffer, count, datatype, root, comm, request, ierror)
40     TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buffer
41     INTEGER, INTENT(IN) :: count, root
42     TYPE(MPI_Datatype), INTENT(IN) :: datatype
43     TYPE(MPI_Comm), INTENT(IN) :: comm
44     TYPE(MPI_Request), INTENT(OUT) :: request
45     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
46
47 MPI_Iexscan(sendbuf, recvbuf, count, datatype, op, comm, request, ierror)
48     TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
49     TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
50     INTEGER, INTENT(IN) :: count

```

```

TYPE(MPI_Datatype), INTENT(IN) :: datatype           1
TYPE(MPI_Op), INTENT(IN) :: op                       2
TYPE(MPI_Comm), INTENT(IN) :: comm                   3
TYPE(MPI_Request), INTENT(OUT) :: request            4
INTEGER, OPTIONAL, INTENT(OUT) :: ierror             5
                                                    6
MPI_Igather(sendbuf, sendcount, sendtype, recvbuf, recvcount, recvtpe,
            root, comm, request, ierror)              7
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf  8
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf        9
INTEGER, INTENT(IN) :: sendcount, recvcount, root      10
TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtpe    11
TYPE(MPI_Comm), INTENT(IN) :: comm                     12
TYPE(MPI_Request), INTENT(OUT) :: request              13
INTEGER, OPTIONAL, INTENT(OUT) :: ierror               14
                                                    15
MPI_Igatherv(sendbuf, sendcount, sendtype, recvbuf, recvcnts, displs,
            recvtpe, root, comm, request, ierror)      16
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf  17
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf        18
INTEGER, INTENT(IN) :: sendcount, root                 19
INTEGER, INTENT(IN), ASYNCHRONOUS :: recvcnts(*), displs(*) 20
TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtpe    21
TYPE(MPI_Comm), INTENT(IN) :: comm                     22
TYPE(MPI_Request), INTENT(OUT) :: request              23
INTEGER, OPTIONAL, INTENT(OUT) :: ierror               24
                                                    25
MPI_Ireduce(sendbuf, recvbuf, count, datatype, op, root, comm, request,
            ierror)                                    26
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf  27
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf        28
INTEGER, INTENT(IN) :: count, root                     29
TYPE(MPI_Datatype), INTENT(IN) :: datatype             30
TYPE(MPI_Op), INTENT(IN) :: op                         31
TYPE(MPI_Comm), INTENT(IN) :: comm                     32
TYPE(MPI_Request), INTENT(OUT) :: request              33
INTEGER, OPTIONAL, INTENT(OUT) :: ierror               34
                                                    35
MPI_Ireduce_scatter(sendbuf, recvbuf, recvcnts, datatype, op, comm,
            request, ierror)                            36
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf  37
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf        38
INTEGER, INTENT(IN), ASYNCHRONOUS :: recvcnts(*)       39
TYPE(MPI_Datatype), INTENT(IN) :: datatype             40
TYPE(MPI_Op), INTENT(IN) :: op                         41
TYPE(MPI_Comm), INTENT(IN) :: comm                     42
TYPE(MPI_Request), INTENT(OUT) :: request              43
INTEGER, OPTIONAL, INTENT(OUT) :: ierror               44
                                                    45
MPI_Ireduce_scatter_block(sendbuf, recvbuf, recvcount, datatype, op, comm,
            request, ierror)                            46
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf  47
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf        48
INTEGER, INTENT(IN), ASYNCHRONOUS :: recvcnts(*)       49
TYPE(MPI_Datatype), INTENT(IN) :: datatype             50
TYPE(MPI_Op), INTENT(IN) :: op                         51
TYPE(MPI_Comm), INTENT(IN) :: comm                     52
TYPE(MPI_Request), INTENT(OUT) :: request              53
INTEGER, OPTIONAL, INTENT(OUT) :: ierror               54

```

```

1         request, ierror)
2     TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
3     TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
4     INTEGER, INTENT(IN) :: recvcount
5     TYPE(MPI_Datatype), INTENT(IN) :: datatype
6     TYPE(MPI_Op), INTENT(IN) :: op
7     TYPE(MPI_Comm), INTENT(IN) :: comm
8     TYPE(MPI_Request), INTENT(OUT) :: request
9     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
10
11 MPI_Iscan(sendbuf, recvbuf, count, datatype, op, comm, request, ierror)
12     TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
13     TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
14     INTEGER, INTENT(IN) :: count
15     TYPE(MPI_Datatype), INTENT(IN) :: datatype
16     TYPE(MPI_Op), INTENT(IN) :: op
17     TYPE(MPI_Comm), INTENT(IN) :: comm
18     TYPE(MPI_Request), INTENT(OUT) :: request
19     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
20
21 MPI_Iscatter(sendbuf, sendcount, sendtype, recvbuf, recvcount, recvtype,
22             root, comm, request, ierror)
23     TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
24     TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
25     INTEGER, INTENT(IN) :: sendcount, recvcount, root
26     TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
27     TYPE(MPI_Comm), INTENT(IN) :: comm
28     TYPE(MPI_Request), INTENT(OUT) :: request
29     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
30
31 MPI_Iscatterv(sendbuf, sendcounts, displs, sendtype, recvbuf, recvcount,
32             recvtype, root, comm, request, ierror)
33     TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
34     TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
35     INTEGER, INTENT(IN), ASYNCHRONOUS :: sendcounts(*), displs(*)
36     INTEGER, INTENT(IN) :: recvcount, root
37     TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
38     TYPE(MPI_Comm), INTENT(IN) :: comm
39     TYPE(MPI_Request), INTENT(OUT) :: request
40     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
41
42 MPI_Op_commutative(op, commute, ierror)
43     TYPE(MPI_Op), INTENT(IN) :: op
44     LOGICAL, INTENT(OUT) :: commute
45     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
46
47 MPI_Op_create(user_fn, commute, op, ierror)
48     PROCEDURE(MPI_User_function) :: user_fn
49     LOGICAL, INTENT(IN) :: commute
50     TYPE(MPI_Op), INTENT(OUT) :: op

```

```

    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
1
MPI_Op_free(op, ierror)
2
    TYPE(MPI_Op), INTENT(INOUT) :: op
3
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
4
5
MPI_Reduce(sendbuf, recvbuf, count, datatype, op, root, comm, ierror)
6
    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
7
    TYPE(*), DIMENSION(..) :: recvbuf
8
    INTEGER, INTENT(IN) :: count, root
9
    TYPE(MPI_Datatype), INTENT(IN) :: datatype
10
    TYPE(MPI_Op), INTENT(IN) :: op
11
    TYPE(MPI_Comm), INTENT(IN) :: comm
12
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
13
14
MPI_Reduce_init(sendbuf, recvbuf, count, datatype, op, root, comm, info,
15
    request, ierror)
16
    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
17
    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
18
    INTEGER, INTENT(IN) :: count, root
19
    TYPE(MPI_Datatype), INTENT(IN) :: datatype
20
    TYPE(MPI_Op), INTENT(IN) :: op
21
    TYPE(MPI_Comm), INTENT(IN) :: comm
22
    TYPE(MPI_Info), INTENT(IN) :: info
23
    TYPE(MPI_Request), INTENT(OUT) :: request
24
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
25
26
MPI_Reduce_local(inbuf, inoutbuf, count, datatype, op, ierror)
27
    TYPE(*), DIMENSION(..), INTENT(IN) :: inbuf
28
    TYPE(*), DIMENSION(..) :: inoutbuf
29
    INTEGER, INTENT(IN) :: count
30
    TYPE(MPI_Datatype), INTENT(IN) :: datatype
31
    TYPE(MPI_Op), INTENT(IN) :: op
32
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
33
34
MPI_Reduce_scatter(sendbuf, recvbuf, recvcounts, datatype, op, comm,
35
    ierror)
36
    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
37
    TYPE(*), DIMENSION(..) :: recvbuf
38
    INTEGER, INTENT(IN) :: recvcounts(*)
39
    TYPE(MPI_Datatype), INTENT(IN) :: datatype
40
    TYPE(MPI_Op), INTENT(IN) :: op
41
    TYPE(MPI_Comm), INTENT(IN) :: comm
42
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
43
44
MPI_Reduce_scatter_block(sendbuf, recvbuf, recvcount, datatype, op, comm,
45
    ierror)
46
    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
47
    TYPE(*), DIMENSION(..) :: recvbuf
48
    INTEGER, INTENT(IN) :: recvcount

```

```

1     TYPE(MPI_Datatype), INTENT(IN) :: datatype
2     TYPE(MPI_Op), INTENT(IN) :: op
3     TYPE(MPI_Comm), INTENT(IN) :: comm
4     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
5
6 MPI_Reduce_scatter_block_init(sendbuf, recvbuf, recvcount, datatype, op,
7     comm, info, request, ierror)
8     TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
9     TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
10    INTEGER, INTENT(IN) :: recvcount
11    TYPE(MPI_Datatype), INTENT(IN) :: datatype
12    TYPE(MPI_Op), INTENT(IN) :: op
13    TYPE(MPI_Comm), INTENT(IN) :: comm
14    TYPE(MPI_Info), INTENT(IN) :: info
15    TYPE(MPI_Request), INTENT(OUT) :: request
16    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
17
18 MPI_Reduce_scatter_init(sendbuf, recvbuf, recvcounts, datatype, op, comm,
19     info, request, ierror)
20    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
21    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
22    INTEGER, INTENT(IN), ASYNCHRONOUS :: recvcounts(*)
23    TYPE(MPI_Datatype), INTENT(IN) :: datatype
24    TYPE(MPI_Op), INTENT(IN) :: op
25    TYPE(MPI_Comm), INTENT(IN) :: comm
26    TYPE(MPI_Info), INTENT(IN) :: info
27    TYPE(MPI_Request), INTENT(OUT) :: request
28    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
29
30 MPI_Scan(sendbuf, recvbuf, count, datatype, op, comm, ierror)
31    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
32    TYPE(*), DIMENSION(..) :: recvbuf
33    INTEGER, INTENT(IN) :: count
34    TYPE(MPI_Datatype), INTENT(IN) :: datatype
35    TYPE(MPI_Op), INTENT(IN) :: op
36    TYPE(MPI_Comm), INTENT(IN) :: comm
37    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
38
39 MPI_Scan_init(sendbuf, recvbuf, count, datatype, op, comm, info, request,
40     ierror)
41    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
42    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
43    INTEGER, INTENT(IN) :: count
44    TYPE(MPI_Datatype), INTENT(IN) :: datatype
45    TYPE(MPI_Op), INTENT(IN) :: op
46    TYPE(MPI_Comm), INTENT(IN) :: comm
47    TYPE(MPI_Info), INTENT(IN) :: info
48    TYPE(MPI_Request), INTENT(OUT) :: request
49    INTEGER, OPTIONAL, INTENT(OUT) :: ierror

```

```

MPI_Scatter(sendbuf, sendcount, sendtype, recvbuf, recvcount, recvtype,
            root, comm, ierror)
    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
    TYPE(*), DIMENSION(..) :: recvbuf
    INTEGER, INTENT(IN) :: sendcount, recvcount, root
    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
    TYPE(MPI_Comm), INTENT(IN) :: comm
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Scatter_init(sendbuf, sendcount, sendtype, recvbuf, recvcount,
                recvtype, root, comm, info, request, ierror)
    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
    INTEGER, INTENT(IN) :: sendcount, recvcount, root
    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
    TYPE(MPI_Comm), INTENT(IN) :: comm
    TYPE(MPI_Info), INTENT(IN) :: info
    TYPE(MPI_Request), INTENT(OUT) :: request
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Scatterv(sendbuf, sendcounts, displs, sendtype, recvbuf, recvcount,
            recvtype, root, comm, ierror)
    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
    TYPE(*), DIMENSION(..) :: recvbuf
    INTEGER, INTENT(IN) :: sendcounts(*), displs(*), recvcount, root
    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
    TYPE(MPI_Comm), INTENT(IN) :: comm
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Scatterv_init(sendbuf, sendcounts, displs, sendtype, recvbuf,
                recvcount, recvtype, root, comm, info, request, ierror)
    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
    INTEGER, INTENT(IN), ASYNCHRONOUS :: sendcounts(*), displs(*)
    INTEGER, INTENT(IN) :: recvcount, root
    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
    TYPE(MPI_Comm), INTENT(IN) :: comm
    TYPE(MPI_Info), INTENT(IN) :: info
    TYPE(MPI_Request), INTENT(OUT) :: request
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror

A.3.4 Groups, Contexts, Communicators, and Caching Fortran 2008 Bindings

MPI_COMM_DUP_FN(oldcomm, comm_keyval, extra_state, attribute_val_in,
                attribute_val_out, flag, ierror)
    TYPE(MPI_Comm) :: oldcomm
    INTEGER :: comm_keyval
    INTEGER(KIND=MPI_ADDRESS_KIND) :: extra_state, attribute_val_in
    INTEGER(KIND=MPI_ADDRESS_KIND) :: attribute_val_out

```

```

1     LOGICAL :: flag
2     INTEGER :: ierror
3
4     MPI_COMM_NULL_COPY_FN(oldcomm, comm_keyval, extra_state, attribute_val_in,
5         attribute_val_out, flag, ierror)
6     TYPE(MPI_Comm) :: oldcomm
7     INTEGER :: comm_keyval
8     INTEGER(KIND=MPI_ADDRESS_KIND) :: extra_state, attribute_val_in
9     INTEGER(KIND=MPI_ADDRESS_KIND) :: attribute_val_out
10    LOGICAL :: flag
11    INTEGER :: ierror
12
13    MPI_COMM_NULL_DELETE_FN(comm, comm_keyval, attribute_val, extra_state,
14        ierror)
15    TYPE(MPI_Comm) :: comm
16    INTEGER :: comm_keyval
17    INTEGER(KIND=MPI_ADDRESS_KIND) :: attribute_val, extra_state
18    INTEGER :: ierror
19
20    MPI_Comm_compare(comm1, comm2, result, ierror)
21    TYPE(MPI_Comm), INTENT(IN) :: comm1, comm2
22    INTEGER, INTENT(OUT) :: result
23    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
24
25    MPI_Comm_create(comm, group, newcomm, ierror)
26    TYPE(MPI_Comm), INTENT(IN) :: comm
27    TYPE(MPI_Group), INTENT(IN) :: group
28    TYPE(MPI_Comm), INTENT(OUT) :: newcomm
29    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
30
31    MPI_Comm_create_group(comm, group, tag, newcomm, ierror)
32    TYPE(MPI_Comm), INTENT(IN) :: comm
33    TYPE(MPI_Group), INTENT(IN) :: group
34    INTEGER, INTENT(IN) :: tag
35    TYPE(MPI_Comm), INTENT(OUT) :: newcomm
36    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
37
38    MPI_Comm_create_keyval(comm_copy_attr_fn, comm_delete_attr_fn, comm_keyval,
39        extra_state, ierror)
40    PROCEDURE(MPI_Comm_copy_attr_function) :: comm_copy_attr_fn
41    PROCEDURE(MPI_Comm_delete_attr_function) :: comm_delete_attr_fn
42    INTEGER, INTENT(OUT) :: comm_keyval
43    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: extra_state
44    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
45
46    MPI_Comm_delete_attr(comm, comm_keyval, ierror)
47    TYPE(MPI_Comm), INTENT(IN) :: comm
48    INTEGER, INTENT(IN) :: comm_keyval
49    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
50
51    MPI_Comm_dup(comm, newcomm, ierror)

```



```

TYPE(MPI_Comm), INTENT(IN) :: comm 1
TYPE(MPI_Comm), INTENT(OUT) :: newcommm 2
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 3
4
MPI_Comm_dup_with_info(comm, info, newcommm, ierror) 5
TYPE(MPI_Comm), INTENT(IN) :: comm 6
TYPE(MPI_Info), INTENT(IN) :: info 7
TYPE(MPI_Comm), INTENT(OUT) :: newcommm 8
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 9
MPI_Comm_free(comm, ierror) 10
TYPE(MPI_Comm), INTENT(INOUT) :: comm 11
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 12
13
MPI_Comm_free_keyval(comm_keyval, ierror) 14
INTEGER, INTENT(INOUT) :: comm_keyval 15
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 16
MPI_Comm_get_attr(comm, comm_keyval, attribute_val, flag, ierror) 17
TYPE(MPI_Comm), INTENT(IN) :: comm 18
INTEGER, INTENT(IN) :: comm_keyval 19
INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(OUT) :: attribute_val 20
LOGICAL, INTENT(OUT) :: flag 21
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 22
23
MPI_Comm_get_info(comm, info_used, ierror) 24
TYPE(MPI_Comm), INTENT(IN) :: comm 25
TYPE(MPI_Info), INTENT(OUT) :: info_used 26
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 27
MPI_Comm_get_name(comm, comm_name, resultlen, ierror) 28
TYPE(MPI_Comm), INTENT(IN) :: comm 29
CHARACTER(LEN=MPI_MAX_OBJECT_NAME), INTENT(OUT) :: comm_name 30
INTEGER, INTENT(OUT) :: resultlen 31
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 32
33
MPI_Comm_group(comm, group, ierror) 34
TYPE(MPI_Comm), INTENT(IN) :: comm 35
TYPE(MPI_Group), INTENT(OUT) :: group 36
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 37
MPI_Comm_idup(comm, newcommm, request, ierror) 38
TYPE(MPI_Comm), INTENT(IN) :: comm 39
TYPE(MPI_Comm), INTENT(OUT), ASYNCHRONOUS :: newcommm 40
TYPE(MPI_Request), INTENT(OUT) :: request 41
INTEGER, OPTIONAL, INTENT(OUT) :: ierror 42
43
MPI_Comm_idup_with_info(comm, info, newcommm, request, ierror) 44
TYPE(MPI_Comm), INTENT(IN) :: comm 45
TYPE(MPI_Info), INTENT(IN) :: info 46
TYPE(MPI_Comm), INTENT(OUT), ASYNCHRONOUS :: newcommm 47
TYPE(MPI_Request), INTENT(OUT) :: request 48

```

```

1     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
2
3     MPI_Comm_rank(comm, rank, ierror)
4     TYPE(MPI_Comm), INTENT(IN) :: comm
5     INTEGER, INTENT(OUT) :: rank
6     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
7
8     MPI_Comm_remote_group(comm, group, ierror)
9     TYPE(MPI_Comm), INTENT(IN) :: comm
10    TYPE(MPI_Group), INTENT(OUT) :: group
11    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
12
13    MPI_Comm_remote_size(comm, size, ierror)
14    TYPE(MPI_Comm), INTENT(IN) :: comm
15    INTEGER, INTENT(OUT) :: size
16    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
17
18    MPI_Comm_set_attr(comm, comm_keyval, attribute_val, ierror)
19    TYPE(MPI_Comm), INTENT(IN) :: comm
20    INTEGER, INTENT(IN) :: comm_keyval
21    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: attribute_val
22    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
23
24    MPI_Comm_set_info(comm, info, ierror)
25    TYPE(MPI_Comm), INTENT(IN) :: comm
26    TYPE(MPI_Info), INTENT(IN) :: info
27    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
28
29    MPI_Comm_set_name(comm, comm_name, ierror)
30    TYPE(MPI_Comm), INTENT(IN) :: comm
31    CHARACTER(LEN=*), INTENT(IN) :: comm_name
32    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
33
34    MPI_Comm_size(comm, size, ierror)
35    TYPE(MPI_Comm), INTENT(IN) :: comm
36    INTEGER, INTENT(OUT) :: size
37    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
38
39    MPI_Comm_split(comm, color, key, newcomm, ierror)
40    TYPE(MPI_Comm), INTENT(IN) :: comm
41    INTEGER, INTENT(IN) :: color, key
42    TYPE(MPI_Comm), INTENT(OUT) :: newcomm
43    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
44
45    MPI_Comm_split_type(comm, split_type, key, info, newcomm, ierror)
46    TYPE(MPI_Comm), INTENT(IN) :: comm
47    INTEGER, INTENT(IN) :: split_type, key
48    TYPE(MPI_Info), INTENT(IN) :: info
49    TYPE(MPI_Comm), INTENT(OUT) :: newcomm
50    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
51
52    MPI_Comm_test_inter(comm, flag, ierror)

```

```

TYPE(MPI_Comm), INTENT(IN) :: comm
LOGICAL, INTENT(OUT) :: flag
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Group_compare(group1, group2, result, ierror)
TYPE(MPI_Group), INTENT(IN) :: group1, group2
INTEGER, INTENT(OUT) :: result
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Group_difference(group1, group2, newgroup, ierror)
TYPE(MPI_Group), INTENT(IN) :: group1, group2
TYPE(MPI_Group), INTENT(OUT) :: newgroup
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Group_excl(group, n, ranks, newgroup, ierror)
TYPE(MPI_Group), INTENT(IN) :: group
INTEGER, INTENT(IN) :: n, ranks(n)
TYPE(MPI_Group), INTENT(OUT) :: newgroup
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Group_free(group, ierror)
TYPE(MPI_Group), INTENT(INOUT) :: group
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Group_incl(group, n, ranks, newgroup, ierror)
TYPE(MPI_Group), INTENT(IN) :: group
INTEGER, INTENT(IN) :: n, ranks(n)
TYPE(MPI_Group), INTENT(OUT) :: newgroup
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Group_intersection(group1, group2, newgroup, ierror)
TYPE(MPI_Group), INTENT(IN) :: group1, group2
TYPE(MPI_Group), INTENT(OUT) :: newgroup
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Group_range_excl(group, n, ranges, newgroup, ierror)
TYPE(MPI_Group), INTENT(IN) :: group
INTEGER, INTENT(IN) :: n, ranges(3,n)
TYPE(MPI_Group), INTENT(OUT) :: newgroup
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Group_range_incl(group, n, ranges, newgroup, ierror)
TYPE(MPI_Group), INTENT(IN) :: group
INTEGER, INTENT(IN) :: n, ranges(3,n)
TYPE(MPI_Group), INTENT(OUT) :: newgroup
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Group_rank(group, rank, ierror)
TYPE(MPI_Group), INTENT(IN) :: group
INTEGER, INTENT(OUT) :: rank
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

```

```

1 MPI_Group_size(group, size, ierror)
2     TYPE(MPI_Group), INTENT(IN) :: group
3     INTEGER, INTENT(OUT) :: size
4     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
5
6 MPI_Group_translate_ranks(group1, n, ranks1, group2, ranks2, ierror)
7     TYPE(MPI_Group), INTENT(IN) :: group1, group2
8     INTEGER, INTENT(IN) :: n, ranks1(n)
9     INTEGER, INTENT(OUT) :: ranks2(n)
10    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
11
12 MPI_Group_union(group1, group2, newgroup, ierror)
13     TYPE(MPI_Group), INTENT(IN) :: group1, group2
14     TYPE(MPI_Group), INTENT(OUT) :: newgroup
15     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
16
17 MPI_Intercomm_create(local_comm, local_leader, peer_comm, remote_leader,
18     tag, newintercomm, ierror)
19     TYPE(MPI_Comm), INTENT(IN) :: local_comm, peer_comm
20     INTEGER, INTENT(IN) :: local_leader, remote_leader, tag
21     TYPE(MPI_Comm), INTENT(OUT) :: newintercomm
22     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
23
24 MPI_Intercomm_merge(intercomm, high, newintracomm, ierror)
25     TYPE(MPI_Comm), INTENT(IN) :: intercomm
26     LOGICAL, INTENT(IN) :: high
27     TYPE(MPI_Comm), INTENT(OUT) :: newintracomm
28     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
29
30 MPI_TYPE_DUP_FN(oldtype, type_keyval, extra_state, attribute_val_in,
31     attribute_val_out, flag, ierror)
32     TYPE(MPI_Datatype) :: oldtype
33     INTEGER :: type_keyval
34     INTEGER(KIND=MPI_ADDRESS_KIND) :: extra_state, attribute_val_in
35     INTEGER(KIND=MPI_ADDRESS_KIND) :: attribute_val_out
36     LOGICAL :: flag
37     INTEGER :: ierror
38
39 MPI_TYPE_NULL_COPY_FN(oldtype, type_keyval, extra_state, attribute_val_in,
40     attribute_val_out, flag, ierror)
41     TYPE(MPI_Datatype) :: oldtype
42     INTEGER :: type_keyval
43     INTEGER(KIND=MPI_ADDRESS_KIND) :: extra_state, attribute_val_in
44     INTEGER(KIND=MPI_ADDRESS_KIND) :: attribute_val_out
45     LOGICAL :: flag
46     INTEGER :: ierror
47
48 MPI_TYPE_NULL_DELETE_FN(datatype, type_keyval, attribute_val, extra_state,
49     ierror)
50     TYPE(MPI_Datatype) :: datatype
51     INTEGER :: type_keyval

```

```

INTEGER(KIND=MPI_ADDRESS_KIND) :: attribute_val, extra_state      1
INTEGER, INTENT(OUT) :: ierror                                  2
                                                                    3
MPI_Type_create_keyval(type_copy_attr_fn, type_delete_attr_fn, type_keyval,
                       extra_state, ierror)                    4
                                                                    5
PROCEDURE(MPI_Type_copy_attr_function) :: type_copy_attr_fn    6
PROCEDURE(MPI_Type_delete_attr_function) :: type_delete_attr_fn 7
INTEGER, INTENT(OUT) :: type_keyval                            8
INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: extra_state     9
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       10
                                                                    11
MPI_Type_delete_attr(datatype, type_keyval, ierror)           12
TYPE(MPI_Datatype), INTENT(IN) :: datatype                    13
INTEGER, INTENT(IN) :: type_keyval                            14
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       15
                                                                    16
MPI_Type_free_keyval(type_keyval, ierror)                      17
INTEGER, INTENT(INOUT) :: type_keyval                         18
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       19
                                                                    20
MPI_Type_get_attr(datatype, type_keyval, attribute_val, flag, ierror)
TYPE(MPI_Datatype), INTENT(IN) :: datatype                    21
INTEGER, INTENT(IN) :: type_keyval                            22
INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(OUT) :: attribute_val 23
LOGICAL, INTENT(OUT) :: flag                                  24
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       25
                                                                    26
MPI_Type_get_name(datatype, type_name, resultlen, ierror)
TYPE(MPI_Datatype), INTENT(IN) :: datatype                    27
CHARACTER(LEN=MPI_MAX_OBJECT_NAME), INTENT(OUT) :: type_name 28
INTEGER, INTENT(OUT) :: resultlen                             29
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       30
                                                                    31
MPI_Type_set_attr(datatype, type_keyval, attribute_val, ierror)
TYPE(MPI_Datatype), INTENT(IN) :: datatype                    32
INTEGER, INTENT(IN) :: type_keyval                            33
INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: attribute_val 34
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       35
                                                                    36
MPI_Type_set_name(datatype, type_name, ierror)                 37
TYPE(MPI_Datatype), INTENT(IN) :: datatype                    38
CHARACTER(LEN=*), INTENT(IN) :: type_name                     39
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       40
                                                                    41
MPI_WIN_DUP_FN(oldwin, win_keyval, extra_state, attribute_val_in,
               attribute_val_out, flag, ierror)                42
TYPE(MPI_Win) :: oldwin                                       43
INTEGER :: win_keyval                                         44
INTEGER(KIND=MPI_ADDRESS_KIND) :: extra_state, attribute_val_in 45
INTEGER(KIND=MPI_ADDRESS_KIND) :: attribute_val_out           46
LOGICAL :: flag                                               47
                                                                    48

```

```

1     INTEGER :: ierror
2
3     MPI_WIN_NULL_COPY_FN(oldwin, win_keyval, extra_state, attribute_val_in,
4         attribute_val_out, flag, ierror)
5     TYPE(MPI_Win) :: oldwin
6     INTEGER :: win_keyval
7     INTEGER(KIND=MPI_ADDRESS_KIND) :: extra_state, attribute_val_in
8     INTEGER(KIND=MPI_ADDRESS_KIND) :: attribute_val_out
9     LOGICAL :: flag
10    INTEGER :: ierror
11
12    MPI_WIN_NULL_DELETE_FN(win, win_keyval, attribute_val, extra_state, ierror)
13    TYPE(MPI_Win) :: win
14    INTEGER :: win_keyval
15    INTEGER(KIND=MPI_ADDRESS_KIND) :: attribute_val, extra_state
16    INTEGER :: ierror
17
18    MPI_Win_create_keyval(win_copy_attr_fn, win_delete_attr_fn, win_keyval,
19        extra_state, ierror)
20    PROCEDURE(MPI_Win_copy_attr_function) :: win_copy_attr_fn
21    PROCEDURE(MPI_Win_delete_attr_function) :: win_delete_attr_fn
22    INTEGER, INTENT(OUT) :: win_keyval
23    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: extra_state
24    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
25
26    MPI_Win_delete_attr(win, win_keyval, ierror)
27    TYPE(MPI_Win), INTENT(IN) :: win
28    INTEGER, INTENT(IN) :: win_keyval
29    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
30
31    MPI_Win_free_keyval(win_keyval, ierror)
32    INTEGER, INTENT(INOUT) :: win_keyval
33    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
34
35    MPI_Win_get_attr(win, win_keyval, attribute_val, flag, ierror)
36    TYPE(MPI_Win), INTENT(IN) :: win
37    INTEGER, INTENT(IN) :: win_keyval
38    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(OUT) :: attribute_val
39    LOGICAL, INTENT(OUT) :: flag
40    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
41
42    MPI_Win_get_name(win, win_name, resultlen, ierror)
43    TYPE(MPI_Win), INTENT(IN) :: win
44    CHARACTER(LEN=MPI_MAX_OBJECT_NAME), INTENT(OUT) :: win_name
45    INTEGER, INTENT(OUT) :: resultlen
46    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
47
48    MPI_Win_set_attr(win, win_keyval, attribute_val, ierror)
49    TYPE(MPI_Win), INTENT(IN) :: win
50    INTEGER, INTENT(IN) :: win_keyval
51    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: attribute_val

```

```

    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
1
MPI_Win_set_name(win, win_name, ierror)
2
    TYPE(MPI_Win), INTENT(IN) :: win
3
    CHARACTER(LEN=*), INTENT(IN) :: win_name
4
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
5
6
7
A.3.5 Process Topologies Fortran 2008 Bindings
8
9
MPI_Cart_coords(comm, rank, maxdims, coords, ierror)
10
    TYPE(MPI_Comm), INTENT(IN) :: comm
11
    INTEGER, INTENT(IN) :: rank, maxdims
12
    INTEGER, INTENT(OUT) :: coords(maxdims)
13
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
14
MPI_Cart_create(comm_old, ndims, dims, periods, reorder, comm_cart, ierror)
15
    TYPE(MPI_Comm), INTENT(IN) :: comm_old
16
    INTEGER, INTENT(IN) :: ndims, dims(ndims)
17
    LOGICAL, INTENT(IN) :: periods(ndims), reorder
18
    TYPE(MPI_Comm), INTENT(OUT) :: comm_cart
19
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
20
21
MPI_Cart_get(comm, maxdims, dims, periods, coords, ierror)
22
    TYPE(MPI_Comm), INTENT(IN) :: comm
23
    INTEGER, INTENT(IN) :: maxdims
24
    INTEGER, INTENT(OUT) :: dims(maxdims), coords(maxdims)
25
    LOGICAL, INTENT(OUT) :: periods(maxdims)
26
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
27
28
MPI_Cart_map(comm, ndims, dims, periods, newrank, ierror)
29
    TYPE(MPI_Comm), INTENT(IN) :: comm
30
    INTEGER, INTENT(IN) :: ndims, dims(ndims)
31
    LOGICAL, INTENT(IN) :: periods(ndims)
32
    INTEGER, INTENT(OUT) :: newrank
33
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
34
MPI_Cart_rank(comm, coords, rank, ierror)
35
    TYPE(MPI_Comm), INTENT(IN) :: comm
36
    INTEGER, INTENT(IN) :: coords(*)
37
    INTEGER, INTENT(OUT) :: rank
38
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
39
40
MPI_Cart_shift(comm, direction, disp, rank_source, rank_dest, ierror)
41
    TYPE(MPI_Comm), INTENT(IN) :: comm
42
    INTEGER, INTENT(IN) :: direction, disp
43
    INTEGER, INTENT(OUT) :: rank_source, rank_dest
44
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
45
MPI_Cart_sub(comm, remain_dims, newcomm, ierror)
46
    TYPE(MPI_Comm), INTENT(IN) :: comm
47
    LOGICAL, INTENT(IN) :: remain_dims(*)
48

```

```

1     TYPE(MPI_Comm), INTENT(OUT) :: newcom
2     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
3
4 MPI_Cartdim_get(comm, ndims, ierror)
5     TYPE(MPI_Comm), INTENT(IN) :: comm
6     INTEGER, INTENT(OUT) :: ndims
7     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
8
9 MPI_Dims_create(nnodes, ndims, dims, ierror)
10    INTEGER, INTENT(IN) :: nnodes, ndims
11    INTEGER, INTENT(INOUT) :: dims(ndims)
12    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
13
14 MPI_Dist_graph_create(comm_old, n, sources, degrees, destinations, weights,
15                      info, reorder, comm_dist_graph, ierror)
16    TYPE(MPI_Comm), INTENT(IN) :: comm_old
17    INTEGER, INTENT(IN) :: n, sources(n), degrees(n), destinations(*)
18    INTEGER, INTENT(IN) :: weights(*)
19    TYPE(MPI_Info), INTENT(IN) :: info
20    LOGICAL, INTENT(IN) :: reorder
21    TYPE(MPI_Comm), INTENT(OUT) :: comm_dist_graph
22    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
23
24 MPI_Dist_graph_create_adjacent(comm_old, indegree, sources, sourceweights,
25                               outdegree, destinations, destweights, info, reorder,
26                               comm_dist_graph, ierror)
27    TYPE(MPI_Comm), INTENT(IN) :: comm_old
28    INTEGER, INTENT(IN) :: indegree, sources(indegree), outdegree,
29    destinations(outdegree)
30    INTEGER, INTENT(IN) :: sourceweights(*), destweights(*)
31    TYPE(MPI_Info), INTENT(IN) :: info
32    LOGICAL, INTENT(IN) :: reorder
33    TYPE(MPI_Comm), INTENT(OUT) :: comm_dist_graph
34    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
35
36 MPI_Dist_graph_neighbors(comm, maxindegree, sources, sourceweights,
37                          maxoutdegree, destinations, destweights, ierror)
38    TYPE(MPI_Comm), INTENT(IN) :: comm
39    INTEGER, INTENT(IN) :: maxindegree, maxoutdegree
40    INTEGER, INTENT(OUT) :: sources(maxindegree),
41    destinations(maxoutdegree)
42    INTEGER :: sourceweights(*), destweights(*)
43    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
44
45 MPI_Dist_graph_neighbors_count(comm, indegree, outdegree, weighted, ierror)
46    TYPE(MPI_Comm), INTENT(IN) :: comm
47    INTEGER, INTENT(OUT) :: indegree, outdegree
48    LOGICAL, INTENT(OUT) :: weighted
49    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
50
51 MPI_Graph_create(comm_old, nnodes, index, edges, reorder, comm_graph,

```



```

        ierror)
1
    TYPE(MPI_Comm), INTENT(IN) :: comm_old
2
    INTEGER, INTENT(IN) :: nnodes, index(nnodes), edges(*)
3
    LOGICAL, INTENT(IN) :: reorder
4
    TYPE(MPI_Comm), INTENT(OUT) :: comm_graph
5
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
6
7
MPI_Graph_get(comm, maxindex, maxedges, index, edges, ierror)
8
    TYPE(MPI_Comm), INTENT(IN) :: comm
9
    INTEGER, INTENT(IN) :: maxindex, maxedges
10
    INTEGER, INTENT(OUT) :: index(maxindex), edges(maxedges)
11
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
12
13
MPI_Graph_map(comm, nnodes, index, edges, newrank, ierror)
14
    TYPE(MPI_Comm), INTENT(IN) :: comm
15
    INTEGER, INTENT(IN) :: nnodes, index(nnodes), edges(*)
16
    INTEGER, INTENT(OUT) :: newrank
17
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
18
19
MPI_Graph_neighbors(comm, rank, maxneighbors, neighbors, ierror)
19
    TYPE(MPI_Comm), INTENT(IN) :: comm
20
    INTEGER, INTENT(IN) :: rank, maxneighbors
21
    INTEGER, INTENT(OUT) :: neighbors(maxneighbors)
22
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
23
24
MPI_Graph_neighbors_count(comm, rank, nneighbors, ierror)
24
    TYPE(MPI_Comm), INTENT(IN) :: comm
25
    INTEGER, INTENT(IN) :: rank
26
    INTEGER, INTENT(OUT) :: nneighbors
27
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
28
29
MPI_Graphdims_get(comm, nnodes, nedges, ierror)
30
    TYPE(MPI_Comm), INTENT(IN) :: comm
31
    INTEGER, INTENT(OUT) :: nnodes, nedges
32
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
33
34
MPI_Ineighbor_allgather(sendbuf, sendcount, sendtype, recvbuf, recvcount,
34
    recvtype, comm, request, ierror)
35
    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
36
    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
37
    INTEGER, INTENT(IN) :: sendcount, recvcount
38
    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
39
    TYPE(MPI_Comm), INTENT(IN) :: comm
40
    TYPE(MPI_Request), INTENT(OUT) :: request
41
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
42
43
MPI_Ineighbor_allgatherv(sendbuf, sendcount, sendtype, recvbuf, recvcoun
44
    t, displs, recvtype, comm, request, ierror)
45
    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
46
    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
47
    INTEGER, INTENT(IN) :: sendcount
48

```

```

1     INTEGER, INTENT(IN), ASYNCHRONOUS :: recvcounts(*), displs(*)
2     TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
3     TYPE(MPI_Comm), INTENT(IN) :: comm
4     TYPE(MPI_Request), INTENT(OUT) :: request
5     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
6
7 MPI_Ineighbor_alltoall(sendbuf, sendcount, sendtype, recvbuf, recvcount,
8     recvtype, comm, request, ierror)
9     TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
10    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
11    INTEGER, INTENT(IN) :: sendcount, recvcount
12    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
13    TYPE(MPI_Comm), INTENT(IN) :: comm
14    TYPE(MPI_Request), INTENT(OUT) :: request
15    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
16
17 MPI_Ineighbor_alltoallv(sendbuf, sendcounts, sdispls, sendtype, recvbuf,
18     recvcounts, rdispls, recvtype, comm, request, ierror)
19    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
20    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
21    INTEGER, INTENT(IN), ASYNCHRONOUS :: sendcounts(*), sdispls(*),
22     recvcounts(*), rdispls(*)
23    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
24    TYPE(MPI_Comm), INTENT(IN) :: comm
25    TYPE(MPI_Request), INTENT(OUT) :: request
26    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
27
28 MPI_Ineighbor_alltoallw(sendbuf, sendcounts, sdispls, sendtypes, recvbuf,
29     recvcounts, rdispls, recvtypes, comm, request, ierror)
30    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
31    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
32    INTEGER, INTENT(IN), ASYNCHRONOUS :: sendcounts(*), recvcounts(*)
33    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN), ASYNCHRONOUS ::
34     sdispls(*), rdispls(*)
35    TYPE(MPI_Datatype), INTENT(IN), ASYNCHRONOUS :: sendtypes(*),
36     recvtypes(*)
37    TYPE(MPI_Comm), INTENT(IN) :: comm
38    TYPE(MPI_Request), INTENT(OUT) :: request
39    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
40
41 MPI_Neighbor_allgather(sendbuf, sendcount, sendtype, recvbuf, recvcount,
42     recvtype, comm, ierror)
43    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
44    TYPE(*), DIMENSION(..) :: recvbuf
45    INTEGER, INTENT(IN) :: sendcount, recvcount
46    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
47    TYPE(MPI_Comm), INTENT(IN) :: comm
48    INTEGER, OPTIONAL, INTENT(OUT) :: ierror

```

```

        recvcount, recvtype, comm, info, request, ierror)
1
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
2
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
3
INTEGER, INTENT(IN) :: sendcount, recvcount
4
TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
5
TYPE(MPI_Comm), INTENT(IN) :: comm
6
TYPE(MPI_Info), INTENT(IN) :: info
7
TYPE(MPI_Request), INTENT(OUT) :: request
8
INTEGER, OPTIONAL, INTENT(OUT) :: ierror
9
10
MPI_Neighbor_allgatherv(sendbuf, sendcount, sendtype, recvbuf, recvcounts,
11
        displs, recvtype, comm, ierror)
12
TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
13
TYPE(*), DIMENSION(..) :: recvbuf
14
INTEGER, INTENT(IN) :: sendcount, recvcounts(*), displs(*)
15
TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
16
TYPE(MPI_Comm), INTENT(IN) :: comm
17
INTEGER, OPTIONAL, INTENT(OUT) :: ierror
18
19
MPI_Neighbor_allgatherv_init(sendbuf, sendcount, sendtype, recvbuf,
20
        recvcounts, displs, recvtype, comm, info, request, ierror)
21
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
22
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
23
INTEGER, INTENT(IN) :: sendcount
24
INTEGER, INTENT(IN), ASYNCHRONOUS :: recvcounts(*), displs(*)
25
TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
26
TYPE(MPI_Comm), INTENT(IN) :: comm
27
TYPE(MPI_Info), INTENT(IN) :: info
28
TYPE(MPI_Request), INTENT(OUT) :: request
29
INTEGER, OPTIONAL, INTENT(OUT) :: ierror
30
31
MPI_Neighbor_alltoall(sendbuf, sendcount, sendtype, recvbuf, recvcount,
32
        recvtype, comm, ierror)
33
TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
34
TYPE(*), DIMENSION(..) :: recvbuf
35
INTEGER, INTENT(IN) :: sendcount, recvcount
36
TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
37
TYPE(MPI_Comm), INTENT(IN) :: comm
38
INTEGER, OPTIONAL, INTENT(OUT) :: ierror
39
40
MPI_Neighbor_alltoall_init(sendbuf, sendcount, sendtype, recvbuf,
41
        recvcount, recvtype, comm, info, request, ierror)
42
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
43
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
44
INTEGER, INTENT(IN) :: sendcount, recvcount
45
TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
46
TYPE(MPI_Comm), INTENT(IN) :: comm
47
TYPE(MPI_Info), INTENT(IN) :: info
48
TYPE(MPI_Request), INTENT(OUT) :: request

```

```

1     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
2
3     MPI_Neighbor_alltoallv(sendbuf, sendcounts, sdispls, sendtype, recvbuf,
4         recvcnts, rdispls, recvtype, comm, ierror)
5     TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
6     TYPE(*), DIMENSION(..) :: recvbuf
7     INTEGER, INTENT(IN) :: sendcounts(*), sdispls(*), recvcnts(*),
8         rdispls(*)
9     TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
10    TYPE(MPI_Comm), INTENT(IN) :: comm
11    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
12
13    MPI_Neighbor_alltoallv_init(sendbuf, sendcounts, sdispls, sendtype,
14        recvbuf, recvcnts, rdispls, recvtype, comm, info, request,
15        ierror)
16    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
17    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
18    INTEGER, INTENT(IN), ASYNCHRONOUS :: sendcounts(*), sdispls(*),
19        recvcnts(*), rdispls(*)
20    TYPE(MPI_Datatype), INTENT(IN) :: sendtype, recvtype
21    TYPE(MPI_Comm), INTENT(IN) :: comm
22    TYPE(MPI_Info), INTENT(IN) :: info
23    TYPE(MPI_Request), INTENT(OUT) :: request
24    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
25
26    MPI_Neighbor_alltoallw(sendbuf, sendcounts, sdispls, sendtypes, recvbuf,
27        recvcnts, rdispls, recvtypes, comm, ierror)
28    TYPE(*), DIMENSION(..), INTENT(IN) :: sendbuf
29    TYPE(*), DIMENSION(..) :: recvbuf
30    INTEGER, INTENT(IN) :: sendcounts(*), recvcnts(*)
31    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: sdispls(*), rdispls(*)
32    TYPE(MPI_Datatype), INTENT(IN) :: sendtypes(*), recvtypes(*)
33    TYPE(MPI_Comm), INTENT(IN) :: comm
34    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
35
36    MPI_Neighbor_alltoallw_init(sendbuf, sendcounts, sdispls, sendtypes,
37        recvbuf, recvcnts, rdispls, recvtypes, comm, info, request,
38        ierror)
39    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: sendbuf
40    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: recvbuf
41    INTEGER, INTENT(IN), ASYNCHRONOUS :: sendcounts(*), recvcnts(*)
42    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN), ASYNCHRONOUS ::
43        sdispls(*), rdispls(*)
44    TYPE(MPI_Datatype), INTENT(IN), ASYNCHRONOUS :: sendtypes(*),
45        recvtypes(*)
46    TYPE(MPI_Comm), INTENT(IN) :: comm
47    TYPE(MPI_Info), INTENT(IN) :: info
48    TYPE(MPI_Request), INTENT(OUT) :: request
49    INTEGER, OPTIONAL, INTENT(OUT) :: ierror

```

```

MPI_Topo_test(comm, status, ierror)
  TYPE(MPI_Comm), INTENT(IN) :: comm
  INTEGER, INTENT(OUT) :: status
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror

A.3.6 MPI Environmental Management Fortran 2008 Bindings

DOUBLE PRECISION MPI_Wtick()

DOUBLE PRECISION MPI_Wtime()

MPI_Abort(comm, errorcode, ierror)
  TYPE(MPI_Comm), INTENT(IN) :: comm
  INTEGER, INTENT(IN) :: errorcode
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Add_error_class(errorclass, ierror)
  INTEGER, INTENT(OUT) :: errorclass
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Add_error_code(errorclass, errorcode, ierror)
  INTEGER, INTENT(IN) :: errorclass
  INTEGER, INTENT(OUT) :: errorcode
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Add_error_string(errorcode, string, ierror)
  INTEGER, INTENT(IN) :: errorcode
  CHARACTER(LEN=*), INTENT(IN) :: string
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Alloc_mem(size, info, baseptr, ierror)
  USE, INTRINSIC :: ISO_C_BINDING, ONLY : C_PTR
  INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: size
  TYPE(MPI_Info), INTENT(IN) :: info
  TYPE(C_PTR), INTENT(OUT) :: baseptr
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Comm_call_errhandler(comm, errorcode, ierror)
  TYPE(MPI_Comm), INTENT(IN) :: comm
  INTEGER, INTENT(IN) :: errorcode
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Comm_create_errhandler(comm_errhandler_fn, errhandler, ierror)
  PROCEDURE(MPI_Comm_errhandler_function) :: comm_errhandler_fn
  Type(MPI_Errhandler), INTENT(OUT) :: errhandler
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_Comm_get_errhandler(comm, errhandler, ierror)
  TYPE(MPI_Comm), INTENT(IN) :: comm
  Type(MPI_Errhandler), INTENT(OUT) :: errhandler
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror

```

```
1 MPI_Comm_set_errhandler(comm, errhandler, ierror)
2   TYPE(MPI_Comm), INTENT(IN) :: comm
3   Type(MPI_Errhandler), INTENT(IN) :: errhandler
4   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
5
6 MPI_Errhandler_free(errhandler, ierror)
7   Type(MPI_Errhandler), INTENT(INOUT) :: errhandler
8   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
9
10 MPI_Error_class(errorcode, errorclass, ierror)
11   INTEGER, INTENT(IN) :: errorcode
12   INTEGER, INTENT(OUT) :: errorclass
13   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
14
15 MPI_Error_string(errorcode, string, resultlen, ierror)
16   INTEGER, INTENT(IN) :: errorcode
17   CHARACTER(LEN=MPI_MAX_ERROR_STRING), INTENT(OUT) :: string
18   INTEGER, INTENT(OUT) :: resultlen
19   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
20
21 MPI_File_call_errhandler(fh, errorcode, ierror)
22   TYPE(MPI_File), INTENT(IN) :: fh
23   INTEGER, INTENT(IN) :: errorcode
24   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
25
26 MPI_File_create_errhandler(file_errhandler_fn, errhandler, ierror)
27   PROCEDURE(MPI_File_errhandler_function) :: file_errhandler_fn
28   Type(MPI_Errhandler), INTENT(OUT) :: errhandler
29   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
30
31 MPI_File_get_errhandler(file, errhandler, ierror)
32   TYPE(MPI_File), INTENT(IN) :: file
33   Type(MPI_Errhandler), INTENT(OUT) :: errhandler
34   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
35
36 MPI_File_set_errhandler(file, errhandler, ierror)
37   TYPE(MPI_File), INTENT(IN) :: file
38   Type(MPI_Errhandler), INTENT(IN) :: errhandler
39   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
40
41 MPI_Finalize(ierror)
42   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
43
44 MPI_Finalized(flag, ierror)
45   LOGICAL, INTENT(OUT) :: flag
46   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
47
48 MPI_Free_mem(base, ierror)
49   TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: base
50   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
51
52 MPI_Get_library_version(version, resultlen, ierror)
53   CHARACTER(LEN=MPI_MAX_LIBRARY_VERSION_STRING), INTENT(OUT) :: version
```

INTEGER, INTENT(OUT) :: resultlen	1
INTEGER, OPTIONAL, INTENT(OUT) :: ierror	2
	3
MPI_Get_processor_name(name, resultlen, ierror)	4
CHARACTER(LEN=MPI_MAX_PROCESSOR_NAME), INTENT(OUT) :: name	5
INTEGER, INTENT(OUT) :: resultlen	6
INTEGER, OPTIONAL, INTENT(OUT) :: ierror	7
	8
MPI_Get_version(version, subversion, ierror)	9
INTEGER, INTENT(OUT) :: version, subversion	10
INTEGER, OPTIONAL, INTENT(OUT) :: ierror	11
	12
MPI_Init(ierror)	13
INTEGER, OPTIONAL, INTENT(OUT) :: ierror	14
	15
MPI_Initialized(flag, ierror)	16
LOGICAL, INTENT(OUT) :: flag	17
INTEGER, OPTIONAL, INTENT(OUT) :: ierror	18
	19
MPI_Win_call_errhandler(win, errorcode, ierror)	20
TYPE(MPI_Win), INTENT(IN) :: win	21
INTEGER, INTENT(IN) :: errorcode	22
INTEGER, OPTIONAL, INTENT(OUT) :: ierror	23
	24
MPI_Win_create_errhandler(win_errhandler_fn, errhandler, ierror)	25
PROCEDURE(MPI_Win_errhandler_function) :: win_errhandler_fn	26
Type(MPI_Errhandler), INTENT(OUT) :: errhandler	27
INTEGER, OPTIONAL, INTENT(OUT) :: ierror	28
	29
MPI_Win_get_errhandler(win, errhandler, ierror)	30
TYPE(MPI_Win), INTENT(IN) :: win	31
Type(MPI_Errhandler), INTENT(OUT) :: errhandler	32
INTEGER, OPTIONAL, INTENT(OUT) :: ierror	33
	34
MPI_Win_set_errhandler(win, errhandler, ierror)	35
TYPE(MPI_Win), INTENT(IN) :: win	36
Type(MPI_Errhandler), INTENT(IN) :: errhandler	37
INTEGER, OPTIONAL, INTENT(OUT) :: ierror	38
	39
A.3.7 The Info Object Fortran 2008 Bindings	40
	41
MPI_Info_create(info, ierror)	42
TYPE(MPI_Info), INTENT(OUT) :: info	43
INTEGER, OPTIONAL, INTENT(OUT) :: ierror	44
	45
MPI_Info_delete(info, key, ierror)	46
TYPE(MPI_Info), INTENT(IN) :: info	47
CHARACTER(LEN=*), INTENT(IN) :: key	48
INTEGER, OPTIONAL, INTENT(OUT) :: ierror	49
	50
MPI_Info_dup(info, newinfo, ierror)	51
TYPE(MPI_Info), INTENT(IN) :: info	52

```

1     TYPE(MPI_Info), INTENT(OUT) :: newinfo
2     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
3
4     MPI_Info_free(info, ierror)
5     TYPE(MPI_Info), INTENT(INOUT) :: info
6     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
7
8     MPI_Info_get(info, key, valuelen, value, flag, ierror)
9     TYPE(MPI_Info), INTENT(IN) :: info
10    CHARACTER(LEN=*), INTENT(IN) :: key
11    INTEGER, INTENT(IN) :: valuelen
12    CHARACTER(LEN=valuelen), INTENT(OUT) :: value
13    LOGICAL, INTENT(OUT) :: flag
14    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
15
16    MPI_Info_get_nkeys(info, nkeys, ierror)
17    TYPE(MPI_Info), INTENT(IN) :: info
18    INTEGER, INTENT(OUT) :: nkeys
19    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
20
21    MPI_Info_get_nthkey(info, n, key, ierror)
22    TYPE(MPI_Info), INTENT(IN) :: info
23    INTEGER, INTENT(IN) :: n
24    CHARACTER(LEN=*), INTENT(OUT) :: key
25    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
26
27    MPI_Info_get_valuelen(info, key, valuelen, flag, ierror)
28    TYPE(MPI_Info), INTENT(IN) :: info
29    CHARACTER(LEN=*), INTENT(IN) :: key
30    INTEGER, INTENT(OUT) :: valuelen
31    LOGICAL, INTENT(OUT) :: flag
32    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
33
34    MPI_Info_set(info, key, value, ierror)
35    TYPE(MPI_Info), INTENT(IN) :: info
36    CHARACTER(LEN=*), INTENT(IN) :: key, value
37    INTEGER, OPTIONAL, INTENT(OUT) :: ierror

```

A.3.8 Process Creation and Management Fortran 2008 Bindings

```

38    MPI_Close_port(port_name, ierror)
39    CHARACTER(LEN=*), INTENT(IN) :: port_name
40    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
41
42    MPI_Comm_accept(port_name, info, root, comm, newcomm, ierror)
43    CHARACTER(LEN=*), INTENT(IN) :: port_name
44    TYPE(MPI_Info), INTENT(IN) :: info
45    INTEGER, INTENT(IN) :: root
46    TYPE(MPI_Comm), INTENT(IN) :: comm
47    TYPE(MPI_Comm), INTENT(OUT) :: newcomm
48    INTEGER, OPTIONAL, INTENT(OUT) :: ierror

```



```

MPI_Comm_connect(port_name, info, root, comm, newcomm, ierror)      1
    CHARACTER(LEN=*), INTENT(IN) :: port_name                       2
    TYPE(MPI_Info), INTENT(IN) :: info                             3
    INTEGER, INTENT(IN) :: root                                    4
    TYPE(MPI_Comm), INTENT(IN) :: comm                             5
    TYPE(MPI_Comm), INTENT(OUT) :: newcomm                         6
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       7
                                                                    8
MPI_Comm_disconnect(comm, ierror)                                    9
    TYPE(MPI_Comm), INTENT(INOUT) :: comm                          10
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       11
                                                                    12
MPI_Comm_get_parent(parent, ierror)                                  13
    TYPE(MPI_Comm), INTENT(OUT) :: parent                           14
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       15
                                                                    16
MPI_Comm_join(fd, intercomm, ierror)                                 17
    INTEGER, INTENT(IN) :: fd                                       18
    TYPE(MPI_Comm), INTENT(OUT) :: intercomm                         19
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       20
                                                                    21
MPI_Comm_spawn(command, argv, maxprocs, info, root, comm, intercomm,
    array_of_errcodes, ierror)                                       22
    CHARACTER(LEN=*), INTENT(IN) :: command, argv(*)                23
    INTEGER, INTENT(IN) :: maxprocs, root                           24
    TYPE(MPI_Info), INTENT(IN) :: info                               25
    TYPE(MPI_Comm), INTENT(IN) :: comm                               26
    TYPE(MPI_Comm), INTENT(OUT) :: intercomm                         27
    INTEGER :: array_of_errcodes(*)                                  28
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       29
                                                                    30
MPI_Comm_spawn_multiple(count, array_of_commands, array_of_argv,
    array_of_maxprocs, array_of_info, root, comm, intercomm,
    array_of_errcodes, ierror)                                       31
    INTEGER, INTENT(IN) :: count, array_of_maxprocs(*), root       32
    CHARACTER(LEN=*), INTENT(IN) :: array_of_commands(*)           33
    CHARACTER(LEN=*), INTENT(IN) :: array_of_argv(count, *)       34
    TYPE(MPI_Info), INTENT(IN) :: array_of_info(*)                 35
    TYPE(MPI_Comm), INTENT(IN) :: comm                               36
    TYPE(MPI_Comm), INTENT(OUT) :: intercomm                         37
    INTEGER :: array_of_errcodes(*)                                  38
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       39
                                                                    40
MPI_Lookup_name(service_name, info, port_name, ierror)             41
    CHARACTER(LEN=*), INTENT(IN) :: service_name                   42
    TYPE(MPI_Info), INTENT(IN) :: info                               43
    CHARACTER(LEN=MPI_MAX_PORT_NAME), INTENT(OUT) :: port_name     44
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror                       45
                                                                    46
MPI_Open_port(info, port_name, ierror)                               47
    TYPE(MPI_Info), INTENT(IN) :: info                               48

```

```

1     CHARACTER(LEN=MPI_MAX_PORT_NAME), INTENT(OUT) :: port_name
2     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
3
4     MPI_Publish_name(service_name, info, port_name, ierror)
5         TYPE(MPI_Info), INTENT(IN) :: info
6         CHARACTER(LEN=*), INTENT(IN) :: service_name, port_name
7         INTEGER, OPTIONAL, INTENT(OUT) :: ierror
8
9     MPI_Unpublish_name(service_name, info, port_name, ierror)
10        CHARACTER(LEN=*), INTENT(IN) :: service_name, port_name
11        TYPE(MPI_Info), INTENT(IN) :: info
12        INTEGER, OPTIONAL, INTENT(OUT) :: ierror
13

```

A.3.9 One-Sided Communications Fortran 2008 Bindings

```

15    MPI_Accumulate(origin_addr, origin_count, origin_datatype, target_rank,
16                  target_disp, target_count, target_datatype, op, win, ierror)
17        TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: origin_addr
18        INTEGER, INTENT(IN) :: origin_count, target_rank, target_count
19        TYPE(MPI_Datatype), INTENT(IN) :: origin_datatype, target_datatype
20        INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: target_disp
21        TYPE(MPI_Op), INTENT(IN) :: op
22        TYPE(MPI_Win), INTENT(IN) :: win
23        INTEGER, OPTIONAL, INTENT(OUT) :: ierror
24
25    MPI_Compare_and_swap(origin_addr, compare_addr, result_addr, datatype,
26                        target_rank, target_disp, win, ierror)
27        TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: origin_addr
28        TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: compare_addr
29        TYPE(*), DIMENSION(..), ASYNCHRONOUS :: result_addr
30        TYPE(MPI_Datatype), INTENT(IN) :: datatype
31        INTEGER, INTENT(IN) :: target_rank
32        INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: target_disp
33        TYPE(MPI_Win), INTENT(IN) :: win
34        INTEGER, OPTIONAL, INTENT(OUT) :: ierror
35
36    MPI_Fetch_and_op(origin_addr, result_addr, datatype, target_rank,
37                   target_disp, op, win, ierror)
38        TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: origin_addr
39        TYPE(*), DIMENSION(..), ASYNCHRONOUS :: result_addr
40        TYPE(MPI_Datatype), INTENT(IN) :: datatype
41        INTEGER, INTENT(IN) :: target_rank
42        INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: target_disp
43        TYPE(MPI_Op), INTENT(IN) :: op
44        TYPE(MPI_Win), INTENT(IN) :: win
45        INTEGER, OPTIONAL, INTENT(OUT) :: ierror
46
47    MPI_Get(origin_addr, origin_count, origin_datatype, target_rank,
48           target_disp, target_count, target_datatype, win, ierror)
49        TYPE(*), DIMENSION(..), ASYNCHRONOUS :: origin_addr

```

```

INTEGER, INTENT(IN) :: origin_count, target_rank, target_count      1
TYPE(MPI_Datatype), INTENT(IN) :: origin_datatype, target_datatype  2
INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: target_disp          3
TYPE(MPI_Win), INTENT(IN) :: win                                   4
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                           5
                                                                    6
MPI_Get_accumulate(origin_addr, origin_count, origin_datatype, result_addr,
                   result_count, result_datatype, target_rank, target_disp,
                   target_count, target_datatype, op, win, ierror)  7
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: origin_addr    8
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: result_addr                9
INTEGER, INTENT(IN) :: origin_count, result_count, target_rank,    10
                   target_count                                     11
TYPE(MPI_Datatype), INTENT(IN) :: origin_datatype, target_datatype,  12
                   result_datatype                               13
INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: target_disp          14
TYPE(MPI_Op), INTENT(IN) :: op                                     15
TYPE(MPI_Win), INTENT(IN) :: win                                   16
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                           17
                                                                    18
MPI_Put(origin_addr, origin_count, origin_datatype, target_rank,
         target_disp, target_count, target_datatype, win, ierror)  19
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: origin_addr    20
INTEGER, INTENT(IN) :: origin_count, target_rank, target_count      21
TYPE(MPI_Datatype), INTENT(IN) :: origin_datatype, target_datatype  22
INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: target_disp          23
TYPE(MPI_Win), INTENT(IN) :: win                                   24
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                           25
                                                                    26
MPI_Raccumulate(origin_addr, origin_count, origin_datatype, target_rank,
                target_disp, target_count, target_datatype, op, win, request,
                ierror)                                           27
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: origin_addr    28
INTEGER, INTENT(IN) :: origin_count, target_rank, target_count      29
TYPE(MPI_Datatype), INTENT(IN) :: origin_datatype, target_datatype  30
INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: target_disp          31
TYPE(MPI_Op), INTENT(IN) :: op                                     32
TYPE(MPI_Win), INTENT(IN) :: win                                   33
TYPE(MPI_Request), INTENT(OUT) :: request                          34
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                           35
                                                                    36
MPI_Rget(origin_addr, origin_count, origin_datatype, target_rank,
          target_disp, target_count, target_datatype, win, request,
          ierror)                                                 37
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: origin_addr                38
INTEGER, INTENT(IN) :: origin_count, target_rank, target_count      39
TYPE(MPI_Datatype), INTENT(IN) :: origin_datatype, target_datatype  40
INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: target_disp          41
TYPE(MPI_Win), INTENT(IN) :: win                                   42
                                                                    43

```

```

1     TYPE(MPI_Request), INTENT(OUT) :: request
2     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
3
4     MPI_Rget_accumulate(origin_addr, origin_count, origin_datatype,
5         result_addr, result_count, result_datatype, target_rank,
6         target_disp, target_count, target_datatype, op, win, request,
7         ierror)
8     TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: origin_addr
9     TYPE(*), DIMENSION(..), ASYNCHRONOUS :: result_addr
10    INTEGER, INTENT(IN) :: origin_count, result_count, target_rank,
11        target_count
12    TYPE(MPI_Datatype), INTENT(IN) :: origin_datatype, target_datatype,
13        result_datatype
14    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: target_disp
15    TYPE(MPI_Op), INTENT(IN) :: op
16    TYPE(MPI_Win), INTENT(IN) :: win
17    TYPE(MPI_Request), INTENT(OUT) :: request
18    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
19
20    MPI_Rput(origin_addr, origin_count, origin_datatype, target_rank,
21        target_disp, target_count, target_datatype, win, request,
22        ierror)
23    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: origin_addr
24    INTEGER, INTENT(IN) :: origin_count, target_rank, target_count
25    TYPE(MPI_Datatype), INTENT(IN) :: origin_datatype, target_datatype
26    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: target_disp
27    TYPE(MPI_Win), INTENT(IN) :: win
28    TYPE(MPI_Request), INTENT(OUT) :: request
29    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
30
31    MPI_Win_allocate(size, disp_unit, info, comm, baseptr, win, ierror)
32    USE, INTRINSIC :: ISO_C_BINDING, ONLY : C_PTR
33    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: size
34    INTEGER, INTENT(IN) :: disp_unit
35    TYPE(MPI_Info), INTENT(IN) :: info
36    TYPE(MPI_Comm), INTENT(IN) :: comm
37    TYPE(C_PTR), INTENT(OUT) :: baseptr
38    TYPE(MPI_Win), INTENT(OUT) :: win
39    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
40
41    MPI_Win_allocate_shared(size, disp_unit, info, comm, baseptr, win, ierror)
42    USE, INTRINSIC :: ISO_C_BINDING, ONLY : C_PTR
43    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: size
44    INTEGER, INTENT(IN) :: disp_unit
45    TYPE(MPI_Info), INTENT(IN) :: info
46    TYPE(MPI_Comm), INTENT(IN) :: comm
47    TYPE(C_PTR), INTENT(OUT) :: baseptr
48    TYPE(MPI_Win), INTENT(OUT) :: win
49    INTEGER, OPTIONAL, INTENT(OUT) :: ierror

```

```

MPI_Win_attach(win, base, size, ierror)                                1
  TYPE(MPI_Win), INTENT(IN) :: win                                    2
  TYPE(*), DIMENSION(..), ASYNCHRONOUS :: base                      3
  INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: size                 4
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror                           5
                                                                    6
MPI_Win_complete(win, ierror)                                         7
  TYPE(MPI_Win), INTENT(IN) :: win                                    8
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror                           9
                                                                    10
MPI_Win_create(base, size, disp_unit, info, comm, win, ierror)       11
  TYPE(*), DIMENSION(..), ASYNCHRONOUS :: base                      12
  INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: size                 13
  INTEGER, INTENT(IN) :: disp_unit                                   14
  TYPE(MPI_Info), INTENT(IN) :: info                                 15
  TYPE(MPI_Comm), INTENT(IN) :: comm                                16
  TYPE(MPI_Win), INTENT(OUT) :: win                                  17
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror                           18
                                                                    19
MPI_Win_create_dynamic(info, comm, win, ierror)                       20
  TYPE(MPI_Info), INTENT(IN) :: info                                 21
  TYPE(MPI_Comm), INTENT(IN) :: comm                                22
  TYPE(MPI_Win), INTENT(OUT) :: win                                  23
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror                           24
                                                                    25
MPI_Win_detach(win, base, ierror)                                     26
  TYPE(MPI_Win), INTENT(IN) :: win                                    27
  TYPE(*), DIMENSION(..), ASYNCHRONOUS :: base                      28
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror                           29
                                                                    30
MPI_Win_fence(assert, win, ierror)                                    31
  INTEGER, INTENT(IN) :: assert                                      32
  TYPE(MPI_Win), INTENT(IN) :: win                                    33
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror                           34
                                                                    35
MPI_Win_flush(rank, win, ierror)                                      36
  INTEGER, INTENT(IN) :: rank                                        37
  TYPE(MPI_Win), INTENT(IN) :: win                                    38
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror                           39
                                                                    40
MPI_Win_flush_all(win, ierror)                                        41
  TYPE(MPI_Win), INTENT(IN) :: win                                    42
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror                           43
                                                                    44
MPI_Win_flush_local(rank, win, ierror)                                45
  INTEGER, INTENT(IN) :: rank                                        46
  TYPE(MPI_Win), INTENT(IN) :: win                                    47
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror                           48
                                                                    49
MPI_Win_flush_local_all(win, ierror)                                  50
  TYPE(MPI_Win), INTENT(IN) :: win                                    51
  INTEGER, OPTIONAL, INTENT(OUT) :: ierror                           52

```

```
1 MPI_Win_free(win, ierror)
2     TYPE(MPI_Win), INTENT(INOUT) :: win
3     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
4
5 MPI_Win_get_group(win, group, ierror)
6     TYPE(MPI_Win), INTENT(IN) :: win
7     TYPE(MPI_Group), INTENT(OUT) :: group
8     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
9
10 MPI_Win_get_info(win, info_used, ierror)
11     TYPE(MPI_Win), INTENT(IN) :: win
12     TYPE(MPI_Info), INTENT(OUT) :: info_used
13     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
14
15 MPI_Win_lock(lock_type, rank, assert, win, ierror)
16     INTEGER, INTENT(IN) :: lock_type, rank, assert
17     TYPE(MPI_Win), INTENT(IN) :: win
18     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
19
20 MPI_Win_lock_all(assert, win, ierror)
21     INTEGER, INTENT(IN) :: assert
22     TYPE(MPI_Win), INTENT(IN) :: win
23     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
24
25 MPI_Win_post(group, assert, win, ierror)
26     TYPE(MPI_Group), INTENT(IN) :: group
27     INTEGER, INTENT(IN) :: assert
28     TYPE(MPI_Win), INTENT(IN) :: win
29     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
30
31 MPI_Win_set_info(win, info, ierror)
32     TYPE(MPI_Win), INTENT(IN) :: win
33     TYPE(MPI_Info), INTENT(IN) :: info
34     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
35
36 MPI_Win_shared_query(win, rank, size, disp_unit, baseptr, ierror)
37     USE, INTRINSIC :: ISO_C_BINDING, ONLY : C_PTR
38     TYPE(MPI_Win), INTENT(IN) :: win
39     INTEGER, INTENT(IN) :: rank
40     INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(OUT) :: size
41     INTEGER, INTENT(OUT) :: disp_unit
42     TYPE(C_PTR), INTENT(OUT) :: baseptr
43     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
44
45 MPI_Win_start(group, assert, win, ierror)
46     TYPE(MPI_Group), INTENT(IN) :: group
47     INTEGER, INTENT(IN) :: assert
48     TYPE(MPI_Win), INTENT(IN) :: win
49     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
50
51 MPI_Win_sync(win, ierror)
52     TYPE(MPI_Win), INTENT(IN) :: win
```

```

    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
1
MPI_Win_test(win, flag, ierror)
2
    TYPE(MPI_Win), INTENT(IN) :: win
3
    LOGICAL, INTENT(OUT) :: flag
4
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
5
6
MPI_Win_unlock(rank, win, ierror)
7
    INTEGER, INTENT(IN) :: rank
8
    TYPE(MPI_Win), INTENT(IN) :: win
9
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
10
11
MPI_Win_unlock_all(win, ierror)
12
    TYPE(MPI_Win), INTENT(IN) :: win
13
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
14
15
MPI_Win_wait(win, ierror)
16
    TYPE(MPI_Win), INTENT(IN) :: win
17
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
18
19
A.3.10 External Interfaces Fortran 2008 Bindings
20
MPI_Grequest_complete(request, ierror)
21
    TYPE(MPI_Request), INTENT(IN) :: request
22
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
23
24
MPI_Grequest_start(query_fn, free_fn, cancel_fn, extra_state, request,
25
    ierror)
26
    PROCEDURE(MPI_Grequest_query_function) :: query_fn
27
    PROCEDURE(MPI_Grequest_free_function) :: free_fn
28
    PROCEDURE(MPI_Grequest_cancel_function) :: cancel_fn
29
    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: extra_state
30
    TYPE(MPI_Request), INTENT(OUT) :: request
31
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
32
33
MPI_Init_thread(required, provided, ierror)
34
    INTEGER, INTENT(IN) :: required
35
    INTEGER, INTENT(OUT) :: provided
36
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
37
38
MPI_Is_thread_main(flag, ierror)
39
    LOGICAL, INTENT(OUT) :: flag
40
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
41
42
MPI_Query_thread(provided, ierror)
43
    INTEGER, INTENT(OUT) :: provided
44
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
45
46
MPI_Status_set_cancelled(status, flag, ierror)
47
    TYPE(MPI_Status), INTENT(INOUT) :: status
48
    LOGICAL, INTENT(IN) :: flag
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror

```

```

1 MPI_Status_set_elements(status, datatype, count, ierror)
2   TYPE(MPI_Status), INTENT(INOUT) :: status
3   TYPE(MPI_Datatype), INTENT(IN) :: datatype
4   INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count
5   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
6
7 MPI_Status_set_elements(status, datatype, count, ierror)
8   TYPE(MPI_Status), INTENT(INOUT) :: status
9   TYPE(MPI_Datatype), INTENT(IN) :: datatype
10  INTEGER, INTENT(IN) :: count
11  INTEGER, OPTIONAL, INTENT(OUT) :: ierror
12
13 MPI_Status_set_elements_x(status, datatype, count, ierror)
14  TYPE(MPI_Status), INTENT(INOUT) :: status
15  TYPE(MPI_Datatype), INTENT(IN) :: datatype
16  INTEGER(KIND=MPI_COUNT_KIND), INTENT(IN) :: count
17  INTEGER, OPTIONAL, INTENT(OUT) :: ierror
18
19 A.3.11 I/O Fortran 2008 Bindings
20
21 MPI_CONVERSION_FN_NULL(userbuf, datatype, count, filebuf, position,
22   extra_state, ierror)
23   USE, INTRINSIC :: ISO_C_BINDING, ONLY : C_PTR
24   TYPE(C_PTR), VALUE :: userbuf, filebuf
25   TYPE(MPI_Datatype) :: datatype
26   INTEGER :: count, ierror
27   INTEGER(KIND=MPI_OFFSET_KIND) :: position
28   INTEGER(KIND=MPI_ADDRESS_KIND) :: extra_state
29
30 MPI_File_close(fh, ierror)
31   TYPE(MPI_File), INTENT(INOUT) :: fh
32   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
33
34 MPI_File_delete(filename, info, ierror)
35   CHARACTER(LEN=*), INTENT(IN) :: filename
36   TYPE(MPI_Info), INTENT(IN) :: info
37   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
38
39 MPI_File_get_amode(fh, amode, ierror)
40   TYPE(MPI_File), INTENT(IN) :: fh
41   INTEGER, INTENT(OUT) :: amode
42   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
43
44 MPI_File_get_atomicsity(fh, flag, ierror)
45   TYPE(MPI_File), INTENT(IN) :: fh
46   LOGICAL, INTENT(OUT) :: flag
47   INTEGER, OPTIONAL, INTENT(OUT) :: ierror
48
49 MPI_File_get_byte_offset(fh, offset, disp, ierror)
50   TYPE(MPI_File), INTENT(IN) :: fh
51   INTEGER(KIND=MPI_OFFSET_KIND), INTENT(IN) :: offset

```



```

    INTEGER(KIND=MPI_OFFSET_KIND), INTENT(OUT) :: disp
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
MPI_File_get_group(fh, group, ierror)
    TYPE(MPI_File), INTENT(IN) :: fh
    TYPE(MPI_Group), INTENT(OUT) :: group
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
MPI_File_get_info(fh, info_used, ierror)
    TYPE(MPI_File), INTENT(IN) :: fh
    TYPE(MPI_Info), INTENT(OUT) :: info_used
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
MPI_File_get_position(fh, offset, ierror)
    TYPE(MPI_File), INTENT(IN) :: fh
    INTEGER(KIND=MPI_OFFSET_KIND), INTENT(OUT) :: offset
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
MPI_File_get_position_shared(fh, offset, ierror)
    TYPE(MPI_File), INTENT(IN) :: fh
    INTEGER(KIND=MPI_OFFSET_KIND), INTENT(OUT) :: offset
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
MPI_File_get_size(fh, size, ierror)
    TYPE(MPI_File), INTENT(IN) :: fh
    INTEGER(KIND=MPI_OFFSET_KIND), INTENT(OUT) :: size
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
MPI_File_get_type_extent(fh, datatype, extent, ierror)
    TYPE(MPI_File), INTENT(IN) :: fh
    TYPE(MPI_Datatype), INTENT(IN) :: datatype
    INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(OUT) :: extent
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
MPI_File_get_view(fh, disp, etype, filetype, datarep, ierror)
    TYPE(MPI_File), INTENT(IN) :: fh
    INTEGER(KIND=MPI_OFFSET_KIND), INTENT(OUT) :: disp
    TYPE(MPI_Datatype), INTENT(OUT) :: etype, filetype
    CHARACTER(LEN=*), INTENT(OUT) :: datarep
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
MPI_File_iread(fh, buf, count, datatype, request, ierror)
    TYPE(MPI_File), INTENT(IN) :: fh
    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf
    INTEGER, INTENT(IN) :: count
    TYPE(MPI_Datatype), INTENT(IN) :: datatype
    TYPE(MPI_Request), INTENT(OUT) :: request
    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
MPI_File_iread_all(fh, buf, count, datatype, request, ierror)
    TYPE(MPI_File), INTENT(IN) :: fh
    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf

```

```

1     INTEGER, INTENT(IN) :: count
2     TYPE(MPI_Datatype), INTENT(IN) :: datatype
3     TYPE(MPI_Request), INTENT(OUT) :: request
4     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
5
6 MPI_File_iread_at(fh, offset, buf, count, datatype, request, ierror)
7     TYPE(MPI_File), INTENT(IN) :: fh
8     INTEGER(KIND=MPI_OFFSET_KIND), INTENT(IN) :: offset
9     TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf
10    INTEGER, INTENT(IN) :: count
11    TYPE(MPI_Datatype), INTENT(IN) :: datatype
12    TYPE(MPI_Request), INTENT(OUT) :: request
13    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
14
15 MPI_File_iread_at_all(fh, offset, buf, count, datatype, request, ierror)
16    TYPE(MPI_File), INTENT(IN) :: fh
17    INTEGER(KIND=MPI_OFFSET_KIND), INTENT(IN) :: offset
18    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf
19    INTEGER, INTENT(IN) :: count
20    TYPE(MPI_Datatype), INTENT(IN) :: datatype
21    TYPE(MPI_Request), INTENT(OUT) :: request
22    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
23
24 MPI_File_iread_shared(fh, buf, count, datatype, request, ierror)
25    TYPE(MPI_File), INTENT(IN) :: fh
26    TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf
27    INTEGER, INTENT(IN) :: count
28    TYPE(MPI_Datatype), INTENT(IN) :: datatype
29    TYPE(MPI_Request), INTENT(OUT) :: request
30    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
31
32 MPI_File_iread_shared(fh, buf, count, datatype, request, ierror)
33    TYPE(MPI_File), INTENT(IN) :: fh
34    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf
35    INTEGER, INTENT(IN) :: count
36    TYPE(MPI_Datatype), INTENT(IN) :: datatype
37    TYPE(MPI_Request), INTENT(OUT) :: request
38    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
39
40 MPI_File_iread_shared(fh, buf, count, datatype, request, ierror)
41    TYPE(MPI_File), INTENT(IN) :: fh
42    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf
43    INTEGER, INTENT(IN) :: count
44    TYPE(MPI_Datatype), INTENT(IN) :: datatype
45    TYPE(MPI_Request), INTENT(OUT) :: request
46    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
47
48 MPI_File_iread_at(fh, offset, buf, count, datatype, request, ierror)
49    TYPE(MPI_File), INTENT(IN) :: fh
50    INTEGER(KIND=MPI_OFFSET_KIND), INTENT(IN) :: offset

```

```

TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf           1
INTEGER, INTENT(IN) :: count                                     2
TYPE(MPI_Datatype), INTENT(IN) :: datatype                     3
TYPE(MPI_Request), INTENT(OUT) :: request                     4
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                      5
                                                                6
MPI_File_iread_at_all(fh, offset, buf, count, datatype, request, ierror) 7
TYPE(MPI_File), INTENT(IN) :: fh                               8
INTEGER(KIND=MPI_OFFSET_KIND), INTENT(IN) :: offset           9
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf      10
INTEGER, INTENT(IN) :: count                                  11
TYPE(MPI_Datatype), INTENT(IN) :: datatype                   12
TYPE(MPI_Request), INTENT(OUT) :: request                    13
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                      14
                                                                15
MPI_File_iread_shared(fh, buf, count, datatype, request, ierror)
TYPE(MPI_File), INTENT(IN) :: fh                               16
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf      17
INTEGER, INTENT(IN) :: count                                  18
TYPE(MPI_Datatype), INTENT(IN) :: datatype                   19
TYPE(MPI_Request), INTENT(OUT) :: request                    20
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                      21
                                                                22
MPI_File_iread(comm, filename, amode, info, fh, ierror)
TYPE(MPI_Comm), INTENT(IN) :: comm                             23
CHARACTER(LEN=*), INTENT(IN) :: filename                      24
INTEGER, INTENT(IN) :: amode                                   25
TYPE(MPI_Info), INTENT(IN) :: info                            26
TYPE(MPI_File), INTENT(OUT) :: fh                             27
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                      28
                                                                29
MPI_File_iread_all(fh, size, ierror)
TYPE(MPI_File), INTENT(IN) :: fh                               30
INTEGER(KIND=MPI_OFFSET_KIND), INTENT(IN) :: size            31
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                      32
                                                                33
MPI_File_iread(fh, buf, count, datatype, status, ierror)
TYPE(MPI_File), INTENT(IN) :: fh                               34
TYPE(*), DIMENSION(..) :: buf                                 35
INTEGER, INTENT(IN) :: count                                  36
TYPE(MPI_Datatype), INTENT(IN) :: datatype                   37
TYPE(MPI_Status) :: status                                    38
INTEGER, OPTIONAL, INTENT(OUT) :: ierror                      39
                                                                40
MPI_File_iread_all(fh, buf, count, datatype, status, ierror)
TYPE(MPI_File), INTENT(IN) :: fh                               41
TYPE(*), DIMENSION(..) :: buf                                 42
INTEGER, INTENT(IN) :: count                                  43
TYPE(MPI_Datatype), INTENT(IN) :: datatype                   44
TYPE(MPI_Status) :: status                                    45
                                                                46
MPI_File_iread_all(fh, buf, count, datatype, status, ierror)
TYPE(MPI_File), INTENT(IN) :: fh                               47
TYPE(*), DIMENSION(..) :: buf                                 48
INTEGER, INTENT(IN) :: count                                  49
TYPE(MPI_Datatype), INTENT(IN) :: datatype                   50
TYPE(MPI_Status) :: status                                    51
                                                                52

```

```

1     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
2
3     MPI_File_read_all_begin(fh, buf, count, datatype, ierror)
4         TYPE(MPI_File), INTENT(IN) :: fh
5         TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf
6         INTEGER, INTENT(IN) :: count
7         TYPE(MPI_Datatype), INTENT(IN) :: datatype
8         INTEGER, OPTIONAL, INTENT(OUT) :: ierror
9
10    MPI_File_read_all_end(fh, buf, status, ierror)
11        TYPE(MPI_File), INTENT(IN) :: fh
12        TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf
13        TYPE(MPI_Status) :: status
14        INTEGER, OPTIONAL, INTENT(OUT) :: ierror
15
16    MPI_File_read_at(fh, offset, buf, count, datatype, status, ierror)
17        TYPE(MPI_File), INTENT(IN) :: fh
18        INTEGER(KIND=MPI_OFFSET_KIND), INTENT(IN) :: offset
19        TYPE(*), DIMENSION(..) :: buf
20        INTEGER, INTENT(IN) :: count
21        TYPE(MPI_Datatype), INTENT(IN) :: datatype
22        TYPE(MPI_Status) :: status
23        INTEGER, OPTIONAL, INTENT(OUT) :: ierror
24
25    MPI_File_read_at_all(fh, offset, buf, count, datatype, status, ierror)
26        TYPE(MPI_File), INTENT(IN) :: fh
27        INTEGER(KIND=MPI_OFFSET_KIND), INTENT(IN) :: offset
28        TYPE(*), DIMENSION(..) :: buf
29        INTEGER, INTENT(IN) :: count
30        TYPE(MPI_Datatype), INTENT(IN) :: datatype
31        TYPE(MPI_Status) :: status
32        INTEGER, OPTIONAL, INTENT(OUT) :: ierror
33
34    MPI_File_read_at_all_begin(fh, offset, buf, count, datatype, ierror)
35        TYPE(MPI_File), INTENT(IN) :: fh
36        INTEGER(KIND=MPI_OFFSET_KIND), INTENT(IN) :: offset
37        TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf
38        INTEGER, INTENT(IN) :: count
39        TYPE(MPI_Datatype), INTENT(IN) :: datatype
40        INTEGER, OPTIONAL, INTENT(OUT) :: ierror
41
42    MPI_File_read_at_all_end(fh, buf, status, ierror)
43        TYPE(MPI_File), INTENT(IN) :: fh
44        TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf
45        TYPE(MPI_Status) :: status
46        INTEGER, OPTIONAL, INTENT(OUT) :: ierror
47
48    MPI_File_read_ordered(fh, buf, count, datatype, status, ierror)
49        TYPE(MPI_File), INTENT(IN) :: fh
50        TYPE(*), DIMENSION(..) :: buf
51        INTEGER, INTENT(IN) :: count

```

```

TYPE(MPI_Datatype), INTENT(IN) :: datatype           1
TYPE(MPI_Status) :: status                           2
INTEGER, OPTIONAL, INTENT(OUT) :: ierror            3
                                                    4
MPI_File_read_ordered_begin(fh, buf, count, datatype, ierror) 5
TYPE(MPI_File), INTENT(IN) :: fh                    6
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf        7
INTEGER, INTENT(IN) :: count                        8
TYPE(MPI_Datatype), INTENT(IN) :: datatype          9
INTEGER, OPTIONAL, INTENT(OUT) :: ierror           10
                                                    11
MPI_File_read_ordered_end(fh, buf, status, ierror)  12
TYPE(MPI_File), INTENT(IN) :: fh                    13
TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf        14
TYPE(MPI_Status) :: status                          15
INTEGER, OPTIONAL, INTENT(OUT) :: ierror           16
                                                    17
MPI_File_read_shared(fh, buf, count, datatype, status, ierror) 18
TYPE(MPI_File), INTENT(IN) :: fh                    19
TYPE(*), DIMENSION(..) :: buf                      20
INTEGER, INTENT(IN) :: count                        21
TYPE(MPI_Datatype), INTENT(IN) :: datatype          22
TYPE(MPI_Status) :: status                          23
INTEGER, OPTIONAL, INTENT(OUT) :: ierror           24
                                                    25
MPI_File_seek(fh, offset, whence, ierror)           26
TYPE(MPI_File), INTENT(IN) :: fh                    27
INTEGER(KIND=MPI_OFFSET_KIND), INTENT(IN) :: offset 28
INTEGER, INTENT(IN) :: whence                       29
INTEGER, OPTIONAL, INTENT(OUT) :: ierror           30
                                                    31
MPI_File_seek_shared(fh, offset, whence, ierror)   32
TYPE(MPI_File), INTENT(IN) :: fh                    33
INTEGER(KIND=MPI_OFFSET_KIND), INTENT(IN) :: offset 34
INTEGER, INTENT(IN) :: whence                       35
INTEGER, OPTIONAL, INTENT(OUT) :: ierror           36
                                                    37
MPI_File_set_atomicity(fh, flag, ierror)           38
TYPE(MPI_File), INTENT(IN) :: fh                    39
LOGICAL, INTENT(IN) :: flag                         40
INTEGER, OPTIONAL, INTENT(OUT) :: ierror           41
                                                    42
MPI_File_set_info(fh, info, ierror)                43
TYPE(MPI_File), INTENT(IN) :: fh                    44
TYPE(MPI_Info), INTENT(IN) :: info                  45
INTEGER, OPTIONAL, INTENT(OUT) :: ierror           46
                                                    47
MPI_File_set_size(fh, size, ierror)                48
TYPE(MPI_File), INTENT(IN) :: fh                    49
INTEGER(KIND=MPI_OFFSET_KIND), INTENT(IN) :: size   50
INTEGER, OPTIONAL, INTENT(OUT) :: ierror           51

```

```

1 MPI_File_set_view(fh, disp, etype, filetype, datarep, info, ierror)
2     TYPE(MPI_File), INTENT(IN) :: fh
3     INTEGER(KIND=MPI_OFFSET_KIND), INTENT(IN) :: disp
4     TYPE(MPI_Datatype), INTENT(IN) :: etype, filetype
5     CHARACTER(LEN=*), INTENT(IN) :: datarep
6     TYPE(MPI_Info), INTENT(IN) :: info
7     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
8
9 MPI_File_sync(fh, ierror)
10    TYPE(MPI_File), INTENT(IN) :: fh
11    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
12
13 MPI_File_write(fh, buf, count, datatype, status, ierror)
14    TYPE(MPI_File), INTENT(IN) :: fh
15    TYPE(*), DIMENSION(..), INTENT(IN) :: buf
16    INTEGER, INTENT(IN) :: count
17    TYPE(MPI_Datatype), INTENT(IN) :: datatype
18    TYPE(MPI_Status) :: status
19    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
20
21 MPI_File_write_all(fh, buf, count, datatype, status, ierror)
22    TYPE(MPI_File), INTENT(IN) :: fh
23    TYPE(*), DIMENSION(..), INTENT(IN) :: buf
24    INTEGER, INTENT(IN) :: count
25    TYPE(MPI_Datatype), INTENT(IN) :: datatype
26    TYPE(MPI_Status) :: status
27    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
28
29 MPI_File_write_all_begin(fh, buf, count, datatype, ierror)
30    TYPE(MPI_File), INTENT(IN) :: fh
31    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf
32    INTEGER, INTENT(IN) :: count
33    TYPE(MPI_Datatype), INTENT(IN) :: datatype
34    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
35
36 MPI_File_write_all_end(fh, buf, status, ierror)
37    TYPE(MPI_File), INTENT(IN) :: fh
38    TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf
39    TYPE(MPI_Status) :: status
40    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
41
42 MPI_File_write_at(fh, offset, buf, count, datatype, status, ierror)
43    TYPE(MPI_File), INTENT(IN) :: fh
44    INTEGER(KIND=MPI_OFFSET_KIND), INTENT(IN) :: offset
45    TYPE(*), DIMENSION(..), INTENT(IN) :: buf
46    INTEGER, INTENT(IN) :: count
47    TYPE(MPI_Datatype), INTENT(IN) :: datatype
48    TYPE(MPI_Status) :: status
49    INTEGER, OPTIONAL, INTENT(OUT) :: ierror
50
51 MPI_File_write_at_all(fh, offset, buf, count, datatype, status, ierror)

```

```

TYPE(MPI_File), INTENT(IN) :: fh
INTEGER(KIND=MPI_OFFSET_KIND), INTENT(IN) :: offset
TYPE(*), DIMENSION(..), INTENT(IN) :: buf
INTEGER, INTENT(IN) :: count
TYPE(MPI_Datatype), INTENT(IN) :: datatype
TYPE(MPI_Status) :: status
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_File_write_at_all_begin(fh, offset, buf, count, datatype, ierror)
TYPE(MPI_File), INTENT(IN) :: fh
INTEGER(KIND=MPI_OFFSET_KIND), INTENT(IN) :: offset
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf
INTEGER, INTENT(IN) :: count
TYPE(MPI_Datatype), INTENT(IN) :: datatype
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_File_write_at_all_end(fh, buf, status, ierror)
TYPE(MPI_File), INTENT(IN) :: fh
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf
TYPE(MPI_Status) :: status
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_File_write_ordered(fh, buf, count, datatype, status, ierror)
TYPE(MPI_File), INTENT(IN) :: fh
TYPE(*), DIMENSION(..), INTENT(IN) :: buf
INTEGER, INTENT(IN) :: count
TYPE(MPI_Datatype), INTENT(IN) :: datatype
TYPE(MPI_Status) :: status
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_File_write_ordered_begin(fh, buf, count, datatype, ierror)
TYPE(MPI_File), INTENT(IN) :: fh
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf
INTEGER, INTENT(IN) :: count
TYPE(MPI_Datatype), INTENT(IN) :: datatype
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_File_write_ordered_end(fh, buf, status, ierror)
TYPE(MPI_File), INTENT(IN) :: fh
TYPE(*), DIMENSION(..), INTENT(IN), ASYNCHRONOUS :: buf
TYPE(MPI_Status) :: status
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

MPI_File_write_shared(fh, buf, count, datatype, status, ierror)
TYPE(MPI_File), INTENT(IN) :: fh
TYPE(*), DIMENSION(..), INTENT(IN) :: buf
INTEGER, INTENT(IN) :: count
TYPE(MPI_Datatype), INTENT(IN) :: datatype
TYPE(MPI_Status) :: status
INTEGER, OPTIONAL, INTENT(OUT) :: ierror

```

```

1 MPI_Register_datarep(datarep, read_conversion_fn, write_conversion_fn,
2     dtype_file_extent_fn, extra_state, ierror)
3     CHARACTER(LEN=*), INTENT(IN) :: datarep
4     PROCEDURE(MPI_Datarep_conversion_function) :: read_conversion_fn
5     PROCEDURE(MPI_Datarep_conversion_function) :: write_conversion_fn
6     PROCEDURE(MPI_Datarep_extent_function) :: dtype_file_extent_fn
7     INTEGER(KIND=MPI_ADDRESS_KIND), INTENT(IN) :: extra_state
8     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
9
10

```

11 A.3.12 Language Bindings Fortran 2008 Bindings

```

12 MPI_F_sync_reg(buf)
13     TYPE(*), DIMENSION(..), ASYNCHRONOUS :: buf
14
15 MPI_Sizeof(x, size, ierror)
16     TYPE(*), DIMENSION(..) :: x
17     INTEGER, INTENT(OUT) :: size
18     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
19
20 MPI_Status_f082f(f08_status, f_status, ierror)
21     TYPE(MPI_Status), INTENT(IN) :: f08_status
22     INTEGER, INTENT(OUT) :: f_status(MPI_STATUS_SIZE)
23     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
24
25 MPI_Status_f2f08(f_status, f08_status, ierror)
26     INTEGER, INTENT(IN) :: f_status(MPI_STATUS_SIZE)
27     TYPE(MPI_Status), INTENT(OUT) :: f08_status
28     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
29
30 MPI_Type_create_f90_complex(p, r, newtype, ierror)
31     INTEGER, INTENT(IN) :: p, r
32     TYPE(MPI_Datatype), INTENT(OUT) :: newtype
33     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
34
35 MPI_Type_create_f90_integer(r, newtype, ierror)
36     INTEGER, INTENT(IN) :: r
37     TYPE(MPI_Datatype), INTENT(OUT) :: newtype
38     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
39
40 MPI_Type_create_f90_real(p, r, newtype, ierror)
41     INTEGER, INTENT(IN) :: p, r
42     TYPE(MPI_Datatype), INTENT(OUT) :: newtype
43     INTEGER, OPTIONAL, INTENT(OUT) :: ierror
44
45 MPI_Type_match_size(typeclass, size, datatype, ierror)
46     INTEGER, INTENT(IN) :: typeclass, size
47     TYPE(MPI_Datatype), INTENT(OUT) :: datatype
48     INTEGER, OPTIONAL, INTENT(OUT) :: ierror

```


A.3.13 Tools / Profiling Interface Fortran 2008 Bindings

```
MPI_Pcontrol(level)
  INTEGER, INTENT(IN) :: level
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48

A.4 Fortran Bindings with mpif.h or the mpi Module

A.4.1 Point-to-Point Communication Fortran Bindings

MPI_BSEND(BUF, COUNT, DATATYPE, DEST, TAG, COMM, IERROR)

<type> BUF(*)

INTEGER COUNT, DATATYPE, DEST, TAG, COMM, IERROR

MPI_BSEND_INIT(BUF, COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR)

<type> BUF(*)

INTEGER COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR

MPI_BUFFER_ATTACH(BUFFER, SIZE, IERROR)

<type> BUFFER(*)

INTEGER SIZE, IERROR

MPI_BUFFER_DETACH(BUFFER_ADDR, SIZE, IERROR)

INTEGER(KIND=MPI_ADDRESS_KIND) BUFFER_ADDR

INTEGER SIZE, IERROR

MPI_CANCEL(REQUEST, IERROR)

INTEGER REQUEST, IERROR

MPI_GET_COUNT(STATUS, DATATYPE, COUNT, IERROR)

INTEGER STATUS(MPI_STATUS_SIZE), DATATYPE, COUNT, IERROR

MPI_IBSEND(BUF, COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR)

<type> BUF(*)

INTEGER COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR

MPI_IMPROBE(SOURCE, TAG, COMM, FLAG, MESSAGE, STATUS, IERROR)

INTEGER SOURCE, TAG, COMM

LOGICAL FLAG

INTEGER MESSAGE, STATUS(MPI_STATUS_SIZE), IERROR

MPI_IMRECV(BUF, COUNT, DATATYPE, MESSAGE, REQUEST, IERROR)

<type> BUF(*)

INTEGER COUNT, DATATYPE, MESSAGE, REQUEST, IERROR

MPI_IPROBE(SOURCE, TAG, COMM, FLAG, STATUS, IERROR)

INTEGER SOURCE, TAG, COMM

LOGICAL FLAG

INTEGER STATUS(MPI_STATUS_SIZE), IERROR

MPI_IRECV(BUF, COUNT, DATATYPE, SOURCE, TAG, COMM, REQUEST, IERROR)

<type> BUF(*)

INTEGER COUNT, DATATYPE, SOURCE, TAG, COMM, REQUEST, IERROR

MPI_IRSEND(BUF, COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR)

<type> BUF(*)

INTEGER COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR

MPI_ISEND(BUF, COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR)

<type> BUF(*)

```

    INTEGER COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR
    1
MPI_ISSEND(BUF, COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR)
    2
    <type> BUF(*)
    3
    INTEGER COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR
    4
    5
MPI_MPROBE(SOURCE, TAG, COMM, MESSAGE, STATUS, IERROR)
    6
    INTEGER SOURCE, TAG, COMM, MESSAGE, STATUS(MPI_STATUS_SIZE), IERROR
    7
    8
MPI_MRECV(BUF, COUNT, DATATYPE, MESSAGE, STATUS, IERROR)
    9
    <type> BUF(*)
    10
    INTEGER COUNT, DATATYPE, MESSAGE, STATUS(MPI_STATUS_SIZE), IERROR
    11
MPI_PROBE(SOURCE, TAG, COMM, STATUS, IERROR)
    12
    INTEGER SOURCE, TAG, COMM, STATUS(MPI_STATUS_SIZE), IERROR
    13
    14
MPI_RECV(BUF, COUNT, DATATYPE, SOURCE, TAG, COMM, STATUS, IERROR)
    15
    <type> BUF(*)
    16
    INTEGER COUNT, DATATYPE, SOURCE, TAG, COMM, STATUS(MPI_STATUS_SIZE),
    17
    IERROR
    18
MPI_RECV_INIT(BUF, COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR)
    19
    <type> BUF(*)
    20
    INTEGER COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR
    21
    22
MPI_REQUEST_FREE(REQUEST, IERROR)
    23
    INTEGER REQUEST, IERROR
    24
    25
MPI_REQUEST_GET_STATUS(REQUEST, FLAG, STATUS, IERROR)
    26
    INTEGER REQUEST
    27
    LOGICAL FLAG
    28
    INTEGER STATUS(MPI_STATUS_SIZE), IERROR
    29
MPI_RSEND(BUF, COUNT, DATATYPE, DEST, TAG, COMM, IERROR)
    30
    <type> BUF(*)
    31
    INTEGER COUNT, DATATYPE, DEST, TAG, COMM, IERROR
    32
    33
MPI_RSEND_INIT(BUF, COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR)
    34
    <type> BUF(*)
    35
    INTEGER COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR
    36
MPI_SEND(BUF, COUNT, DATATYPE, DEST, TAG, COMM, IERROR)
    37
    <type> BUF(*)
    38
    INTEGER COUNT, DATATYPE, DEST, TAG, COMM, IERROR
    39
    40
MPI_SENDRECV(SENDBUF, SENDCOUNT, SENDTYPE, DEST, SENDTAG, RECVBUF,
    41
    RECVCOUNT, RECVMODE, SOURCE, RECVMODE, COMM, STATUS, IERROR)
    42
    <type> SENDBUF(*)
    43
    INTEGER SENDCOUNT, SENDTYPE, DEST, SENDTAG
    44
    <type> RECVBUF(*)
    45
    INTEGER RECVCOUNT, RECVMODE, SOURCE, RECVMODE, COMM,
    46
    STATUS(MPI_STATUS_SIZE), IERROR
    47
MPI_SENDRECV_REPLACE(BUF, COUNT, DATATYPE, DEST, SENDTAG, SOURCE, RECVMODE,
    48

```

```

1         COMM, STATUS, IERROR)
2     <type> BUF(*)
3     INTEGER COUNT, DATATYPE, DEST, SENDTAG, SOURCE, RECVTAG, COMM,
4     STATUS(MPI_STATUS_SIZE), IERROR
5
6 MPI_SEND_INIT(BUF, COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR)
7     <type> BUF(*)
8     INTEGER COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR
9
10 MPI_SSEND(BUF, COUNT, DATATYPE, DEST, TAG, COMM, IERROR)
11     <type> BUF(*)
12     INTEGER COUNT, DATATYPE, DEST, TAG, COMM, IERROR
13
14 MPI_SSEND_INIT(BUF, COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR)
15     <type> BUF(*)
16     INTEGER COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR
17
18 MPI_START(REQUEST, IERROR)
19     INTEGER REQUEST, IERROR
20
21 MPI_STARTALL(COUNT, ARRAY_OF_REQUESTS, IERROR)
22     INTEGER COUNT, ARRAY_OF_REQUESTS(*), IERROR
23
24 MPI_TEST(REQUEST, FLAG, STATUS, IERROR)
25     INTEGER REQUEST
26     LOGICAL FLAG
27     INTEGER STATUS(MPI_STATUS_SIZE), IERROR
28
29 MPI_TESTALL(COUNT, ARRAY_OF_REQUESTS, FLAG, ARRAY_OF_STATUSES, IERROR)
30     INTEGER COUNT, ARRAY_OF_REQUESTS(*)
31     LOGICAL FLAG
32     INTEGER ARRAY_OF_STATUSES(MPI_STATUS_SIZE,*), IERROR
33
34 MPI_TESTANY(COUNT, ARRAY_OF_REQUESTS, INDEX, FLAG, STATUS, IERROR)
35     INTEGER COUNT, ARRAY_OF_REQUESTS(*), INDEX
36     LOGICAL FLAG
37     INTEGER STATUS(MPI_STATUS_SIZE), IERROR
38
39 MPI_TESTSOME(INCOUNT, ARRAY_OF_REQUESTS, OUTCOUNT, ARRAY_OF_INDICES,
40             ARRAY_OF_STATUSES, IERROR)
41     INTEGER INCOUNT, ARRAY_OF_REQUESTS(*), OUTCOUNT, ARRAY_OF_INDICES(*),
42     ARRAY_OF_STATUSES(MPI_STATUS_SIZE,*), IERROR
43
44 MPI_TEST_CANCELLED(STATUS, FLAG, IERROR)
45     INTEGER STATUS(MPI_STATUS_SIZE)
46     LOGICAL FLAG
47     INTEGER IERROR
48
49 MPI_WAIT(REQUEST, STATUS, IERROR)
50     INTEGER REQUEST, STATUS(MPI_STATUS_SIZE), IERROR
51
52 MPI_WAITALL(COUNT, ARRAY_OF_REQUESTS, ARRAY_OF_STATUSES, IERROR)
53     INTEGER COUNT, ARRAY_OF_REQUESTS(*),

```

```

    ARRAY_OF_STATUSES(MPI_STATUS_SIZE,*), IERROR                                1
MPI_WAITANY(COUNT, ARRAY_OF_REQUESTS, INDEX, STATUS, IERROR)                  2
    INTEGER COUNT, ARRAY_OF_REQUESTS(*), INDEX, STATUS(MPI_STATUS_SIZE),      3
    IERROR                                                                      4
MPI_WAITSOME(INCOUNT, ARRAY_OF_REQUESTS, OUTCOUNT, ARRAY_OF_INDICES,       5
    ARRAY_OF_STATUSES, IERROR)                                                6
    INTEGER INCOUNT, ARRAY_OF_REQUESTS(*), OUTCOUNT, ARRAY_OF_INDICES(*),    7
    ARRAY_OF_STATUSES(MPI_STATUS_SIZE,*), IERROR                              8
                                                                              9
                                                                              10
                                                                              11
A.4.2 Datatypes Fortran Bindings                                           12
                                                                              13
INTEGER(KIND=MPI_ADDRESS_KIND) MPI_AINT_ADD(BASE, DISP)                       14
    INTEGER(KIND=MPI_ADDRESS_KIND) BASE, DISP                                 15
                                                                              16
INTEGER(KIND=MPI_ADDRESS_KIND) MPI_AINT_DIFF(ADDR1, ADDR2)                   17
    INTEGER(KIND=MPI_ADDRESS_KIND) ADDR1, ADDR2                              18
MPI_GET_ADDRESS(LOCATION, ADDRESS, IERROR)                                     19
    <type> LOCATION(*)                                                       20
    INTEGER IERROR                                                            21
    INTEGER(KIND=MPI_ADDRESS_KIND) ADDRESS                                    22
MPI_GET_ELEMENTS(STATUS, DATATYPE, COUNT, IERROR)                            23
    INTEGER STATUS(MPI_STATUS_SIZE), DATATYPE, COUNT, IERROR                 24
                                                                              25
MPI_GET_ELEMENTS_X(STATUS, DATATYPE, COUNT, IERROR)                          26
    INTEGER STATUS(MPI_STATUS_SIZE), DATATYPE, IERROR                        27
    INTEGER(KIND=MPI_COUNT_KIND) COUNT                                       28
MPI_PACK(INBUF, INCOUNT, DATATYPE, OUTBUF, OUTSIZE, POSITION, COMM, IERROR)    29
    <type> INBUF(*), OUTBUF(*)                                               30
    INTEGER INCOUNT, DATATYPE, OUTSIZE, POSITION, COMM, IERROR                 31
                                                                              32
MPI_PACK_EXTERNAL(DATAREP, INBUF, INCOUNT, DATATYPE, OUTBUF, OUTSIZE,        33
    POSITION, IERROR)                                                         34
    INTEGER INCOUNT, DATATYPE, IERROR                                         35
    INTEGER(KIND=MPI_ADDRESS_KIND) OUTSIZE, POSITION                           36
    CHARACTER*(*) DATAREP                                                    37
    <type> INBUF(*), OUTBUF(*)                                               38
MPI_PACK_EXTERNAL_SIZE(DATAREP, INCOUNT, DATATYPE, SIZE, IERROR)             39
    INTEGER INCOUNT, DATATYPE, IERROR                                         40
    INTEGER(KIND=MPI_ADDRESS_KIND) SIZE                                       41
    CHARACTER*(*) DATAREP                                                    42
                                                                              43
MPI_PACK_SIZE(INCOUNT, DATATYPE, COMM, SIZE, IERROR)                        44
    INTEGER INCOUNT, DATATYPE, COMM, SIZE, IERROR                            45
MPI_TYPE_COMMIT(DATATYPE, IERROR)                                           46
    INTEGER DATATYPE, IERROR                                                  47
                                                                              48

```

```

1 MPI_TYPE_CONTIGUOUS(COUNT, OLDTYPE, NEWTYPE, IERROR)
2     INTEGER COUNT, OLDTYPE, NEWTYPE, IERROR
3
4 MPI_TYPE_CONTIGUOUS(COUNT, OLDTYPE, NEWTYPE, IERROR)
5     INTEGER COUNT, OLDTYPE, NEWTYPE, IERROR
6
7 MPI_TYPE_CREATE_DARRAY(SIZE, RANK, NDIMS, ARRAY_OF_GSIZES,
8     ARRAY_OF_DISTRIBS, ARRAY_OF_DARGS, ARRAY_OF_PSIZEs, ORDER,
9     OLDTYPE, NEWTYPE, IERROR)
10    INTEGER SIZE, RANK, NDIMS, ARRAY_OF_GSIZES(*), ARRAY_OF_DISTRIBS(*),
11    ARRAY_OF_DARGS(*), ARRAY_OF_PSIZEs(*), ORDER, OLDTYPE,
12    NEWTYPE, IERROR
13
14 MPI_TYPE_CREATE_HINDEXED(COUNT, ARRAY_OF_BLOCKLENGTHS,
15    ARRAY_OF_DISPLACEMENTS, OLDTYPE, NEWTYPE, IERROR)
16    INTEGER COUNT, ARRAY_OF_BLOCKLENGTHS(*), OLDTYPE, NEWTYPE, IERROR
17    INTEGER(KIND=MPI_ADDRESS_KIND) ARRAY_OF_DISPLACEMENTS(*)
18
19 MPI_TYPE_CREATE_HINDEXED_BLOCK(COUNT, BLOCKLENGTH, ARRAY_OF_DISPLACEMENTS,
20    OLDTYPE, NEWTYPE, IERROR)
21    INTEGER COUNT, BLOCKLENGTH, OLDTYPE, NEWTYPE, IERROR
22    INTEGER(KIND=MPI_ADDRESS_KIND) ARRAY_OF_DISPLACEMENTS(*)
23
24 MPI_TYPE_CREATE_HVECTOR(COUNT, BLOCKLENGTH, STRIDE, OLDTYPE, NEWTYPE,
25    IERROR)
26    INTEGER COUNT, BLOCKLENGTH, OLDTYPE, NEWTYPE, IERROR
27    INTEGER(KIND=MPI_ADDRESS_KIND) STRIDE
28
29 MPI_TYPE_CREATE_INDEXED_BLOCK(COUNT, BLOCKLENGTH, ARRAY_OF_DISPLACEMENTS,
30    OLDTYPE, NEWTYPE, IERROR)
31    INTEGER COUNT, BLOCKLENGTH, ARRAY_OF_DISPLACEMENTS(*), OLDTYPE,
32    NEWTYPE, IERROR
33
34 MPI_TYPE_CREATE_RESIZED(OLDTYPE, LB, EXTENT, NEWTYPE, IERROR)
35    INTEGER OLDTYPE, NEWTYPE, IERROR
36    INTEGER(KIND=MPI_ADDRESS_KIND) LB, EXTENT
37
38 MPI_TYPE_CREATE_STRUCT(COUNT, ARRAY_OF_BLOCKLENGTHS,
39    ARRAY_OF_DISPLACEMENTS, ARRAY_OF_TYPES, NEWTYPE, IERROR)
40    INTEGER COUNT, ARRAY_OF_BLOCKLENGTHS(*), ARRAY_OF_TYPES(*), NEWTYPE,
41    IERROR
42    INTEGER(KIND=MPI_ADDRESS_KIND) ARRAY_OF_DISPLACEMENTS(*)
43
44 MPI_TYPE_CREATE_SUBARRAY(NDIMS, ARRAY_OF_SIZES, ARRAY_OF_SUBSIZES,
45    ARRAY_OF_STARTS, ORDER, OLDTYPE, NEWTYPE, IERROR)
46    INTEGER NDIMS, ARRAY_OF_SIZES(*), ARRAY_OF_SUBSIZES(*),
47    ARRAY_OF_STARTS(*), ORDER, OLDTYPE, NEWTYPE, IERROR
48
49 MPI_TYPE_DUP(OLDTYPE, NEWTYPE, IERROR)
50    INTEGER OLDTYPE, NEWTYPE, IERROR
51
52 MPI_TYPE_FREE(DATATYPE, IERROR)
53    INTEGER DATATYPE, IERROR

```

```

MPI_TYPE_GET_CONTENTS(DATATYPE, MAX_INTEGERS, MAX_ADDRESSES, MAX_DATATYPES,
    ARRAY_OF_INTEGERS, ARRAY_OF_ADDRESSES, ARRAY_OF_DATATYPES,
    IERROR)
    INTEGER DATATYPE, MAX_INTEGERS, MAX_ADDRESSES, MAX_DATATYPES,
        ARRAY_OF_INTEGERS(*), ARRAY_OF_DATATYPES(*), IERROR
    INTEGER(KIND=MPI_ADDRESS_KIND) ARRAY_OF_ADDRESSES(*)
MPI_TYPE_GET_ENVELOPE(DATATYPE, NUM_INTEGERS, NUM_ADDRESSES, NUM_DATATYPES,
    COMBINER, IERROR)
    INTEGER DATATYPE, NUM_INTEGERS, NUM_ADDRESSES, NUM_DATATYPES, COMBINER,
    IERROR
MPI_TYPE_GET_EXTENT(DATATYPE, LB, EXTENT, IERROR)
    INTEGER DATATYPE, IERROR
    INTEGER(KIND=MPI_ADDRESS_KIND) LB, EXTENT
MPI_TYPE_GET_EXTENT_X(DATATYPE, LB, EXTENT, IERROR)
    INTEGER DATATYPE, IERROR
    INTEGER(KIND=MPI_COUNT_KIND) LB, EXTENT
MPI_TYPE_GET_TRUE_EXTENT(DATATYPE, TRUE_LB, TRUE_EXTENT, IERROR)
    INTEGER DATATYPE, IERROR
    INTEGER(KIND=MPI_ADDRESS_KIND) TRUE_LB, TRUE_EXTENT
MPI_TYPE_GET_TRUE_EXTENT_X(DATATYPE, TRUE_LB, TRUE_EXTENT, IERROR)
    INTEGER DATATYPE, IERROR
    INTEGER(KIND=MPI_COUNT_KIND) TRUE_LB, TRUE_EXTENT
MPI_TYPE_INDEXED(COUNT, ARRAY_OF_BLOCKLENGTHS, ARRAY_OF_DISPLACEMENTS,
    OLDTYPE, NEWTYPE, IERROR)
    INTEGER COUNT, ARRAY_OF_BLOCKLENGTHS(*), ARRAY_OF_DISPLACEMENTS(*),
    OLDTYPE, NEWTYPE, IERROR
MPI_TYPE_SIZE(DATATYPE, SIZE, IERROR)
    INTEGER DATATYPE, SIZE, IERROR
MPI_TYPE_SIZE_X(DATATYPE, SIZE, IERROR)
    INTEGER DATATYPE, IERROR
    INTEGER(KIND=MPI_COUNT_KIND) SIZE
MPI_TYPE_VECTOR(COUNT, BLOCKLENGTH, STRIDE, OLDTYPE, NEWTYPE, IERROR)
    INTEGER COUNT, BLOCKLENGTH, STRIDE, OLDTYPE, NEWTYPE, IERROR
MPI_UNPACK(INBUF, INSIZE, POSITION, OUTBUF, OUTCOUNT, DATATYPE, COMM,
    IERROR)
    <type> INBUF(*), OUTBUF(*)
    INTEGER INSIZE, POSITION, OUTCOUNT, DATATYPE, COMM, IERROR
MPI_UNPACK_EXTERNAL(DATAREP, INBUF, INSIZE, POSITION, OUTBUF, OUTCOUNT,
    DATATYPE, IERROR)
    INTEGER OUTCOUNT, DATATYPE, IERROR
    INTEGER(KIND=MPI_ADDRESS_KIND) INSIZE, POSITION
    CHARACTER*(*) DATAREP

```

1 <type> INBUF(*), OUTBUF(*)
 2
 3

4 A.4.3 Collective Communication Fortran Bindings

5 MPI_ALLGATHER(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNT, RECVTYPE,
 6 COMM, IERROR)

7 <type> SENDBUF(*), RECVBUF(*)

8 INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVTYPE, COMM, IERROR
 9

10 MPI_ALLGATHERV(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNTS, DISPLS,
 11 RECVTYPE, COMM, IERROR)

12 <type> SENDBUF(*), RECVBUF(*)

13 INTEGER SENDCOUNT, SENDTYPE, RECVCOUNTS(*), DISPLS(*), RECVTYPE, COMM,
 14 IERROR

15 MPI_ALLGATHERV_INIT(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNTS,
 16 DISPLS, RECVTYPE, COMM, INFO, REQUEST, IERROR)

17 <type> SENDBUF(*), RECVBUF(*)

18 INTEGER SENDCOUNT, SENDTYPE, RECVCOUNTS(*), DISPLS(*), RECVTYPE, COMM,
 19 INFO, REQUEST, IERROR
 20

21 MPI_ALLGATHER_INIT(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNT,
 22 RECVTYPE, COMM, INFO, REQUEST, IERROR)

23 <type> SENDBUF(*), RECVBUF(*)

24 INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVTYPE, COMM, INFO, REQUEST,
 25 IERROR

26 MPI_ALLREDUCE(SENDBUF, RECVBUF, COUNT, DATATYPE, OP, COMM, IERROR)

27 <type> SENDBUF(*), RECVBUF(*)

28 INTEGER COUNT, DATATYPE, OP, COMM, IERROR
 29

30 MPI_ALLREDUCE_INIT(SENDBUF, RECVBUF, COUNT, DATATYPE, OP, COMM, INFO,
 31 REQUEST, IERROR)

32 <type> SENDBUF(*), RECVBUF(*)

33 INTEGER COUNT, DATATYPE, OP, COMM, INFO, REQUEST, IERROR
 34

35 MPI_ALLTOALL(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNT, RECVTYPE,
 36 COMM, IERROR)

37 <type> SENDBUF(*), RECVBUF(*)

38 INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVTYPE, COMM, IERROR

39 MPI_ALLTOALLV(SENDBUF, SENDCOUNTS, SDISPLS, SENDTYPE, RECVBUF, RECVCOUNTS,
 40 RDISPLS, RECVTYPE, COMM, IERROR)

41 <type> SENDBUF(*), RECVBUF(*)

42 INTEGER SENDCOUNTS(*), SDISPLS(*), SENDTYPE, RECVCOUNTS(*), RDISPLS(*),
 43 RECVTYPE, COMM, IERROR
 44

45 MPI_ALLTOALLV_INIT(SENDBUF, SENDCOUNTS, SDISPLS, SENDTYPE, RECVBUF,
 46 RECVCOUNTS, RDISPLS, RECVTYPE, COMM, INFO, REQUEST, IERROR)

47 <type> SENDBUF(*), RECVBUF(*)

48 INTEGER SENDCOUNTS(*), SDISPLS(*), SENDTYPE, RECVCOUNTS(*), RDISPLS(*),


```

    RECVTYPE, COMM, INFO, REQUEST, IERROR)
1
MPI_ALLTOALLW(SENDBUF, SENDCOUNTS, SDISPLS, SENDTYPES, RECVBUF, RECVCOUNTS,
2
    RDISPLS, RECVTYPES, COMM, IERROR)
3
    <type> SENDBUF(*), RECVBUF(*)
4
    INTEGER SENDCOUNTS(*), SDISPLS(*), SENDTYPES(*), RECVCOUNTS(*),
5
    RDISPLS(*), RECVTYPES(*), COMM, IERROR
6
MPI_ALLTOALLW_INIT(SENDBUF, SENDCOUNTS, SDISPLS, SENDTYPES, RECVBUF,
7
    RECVCOUNTS, RDISPLS, RECVTYPES, COMM, INFO, REQUEST, IERROR)
8
    <type> SENDBUF(*), RECVBUF(*)
9
    INTEGER SENDCOUNTS(*), SDISPLS(*), SENDTYPES(*), RECVCOUNTS(*),
10
    RDISPLS(*), RECVTYPES(*), COMM, INFO, REQUEST, IERROR
11
MPI_ALLTOALL_INIT(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNT,
12
    RECVTYPE, COMM, INFO, REQUEST, IERROR)
13
    <type> SENDBUF(*), RECVBUF(*)
14
    INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVTYPE, COMM, INFO, REQUEST,
15
    IERROR
16
MPI_BARRIER(COMM, IERROR)
17
    INTEGER COMM, IERROR
18
MPI_BARRIER_INIT(COMM, INFO, REQUEST, IERROR)
19
    INTEGER COMM, INFO, REQUEST, IERROR
20
MPI_BCAST(BUFFER, COUNT, DATATYPE, ROOT, COMM, IERROR)
21
    <type> BUFFER(*)
22
    INTEGER COUNT, DATATYPE, ROOT, COMM, IERROR
23
MPI_BCAST_INIT(BUFFER, COUNT, DATATYPE, ROOT, COMM, INFO, REQUEST, IERROR)
24
    <type> BUFFER(*)
25
    INTEGER COUNT, DATATYPE, ROOT, COMM, INFO, REQUEST, IERROR
26
MPI_EXSCAN(SENDBUF, RECVBUF, COUNT, DATATYPE, OP, COMM, IERROR)
27
    <type> SENDBUF(*), RECVBUF(*)
28
    INTEGER COUNT, DATATYPE, OP, COMM, IERROR
29
MPI_EXSCAN_INIT(SENDBUF, RECVBUF, COUNT, DATATYPE, OP, COMM, INFO, REQUEST,
30
    IERROR)
31
    <type> SENDBUF(*), RECVBUF(*)
32
    INTEGER COUNT, DATATYPE, OP, COMM, INFO, REQUEST, IERROR
33
MPI_GATHER(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNT, RECVTYPE,
34
    ROOT, COMM, IERROR)
35
    <type> SENDBUF(*), RECVBUF(*)
36
    INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVTYPE, ROOT, COMM, IERROR
37
MPI_GATHERV(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNTS, DISPLS,
38
    RECVTYPE, ROOT, COMM, IERROR)
39
    <type> SENDBUF(*), RECVBUF(*)
40
    INTEGER SENDCOUNT, SENDTYPE, RECVCOUNTS(*), DISPLS(*), RECVTYPE, ROOT,
41
    COMM, IERROR
42
43
44
45
46
47
48

```

```

1 MPI_GATHERV_INIT(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNTS, DISPLS,
2     RECVTYPE, ROOT, COMM, INFO, REQUEST, IERROR)
3     <type> SENDBUF(*), RECVBUF(*)
4     INTEGER SENDCOUNT, SENDTYPE, RECVCOUNTS(*), DISPLS(*), RECVTYPE, ROOT,
5     COMM, INFO, REQUEST, IERROR
6
7 MPI_GATHER_INIT(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNT, RECVTYPE,
8     ROOT, COMM, INFO, REQUEST, IERROR)
9     <type> SENDBUF(*), RECVBUF(*)
10    INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVTYPE, ROOT, COMM, INFO,
11    REQUEST, IERROR
12
13 MPI_IALLGATHER(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNT, RECVTYPE,
14    COMM, REQUEST, IERROR)
15    <type> SENDBUF(*), RECVBUF(*)
16    INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVTYPE, COMM, REQUEST, IERROR
17
18 MPI_IALLGATHERV(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNTS, DISPLS,
19    RECVTYPE, COMM, REQUEST, IERROR)
20    <type> SENDBUF(*), RECVBUF(*)
21    INTEGER SENDCOUNT, SENDTYPE, RECVCOUNTS(*), DISPLS(*), RECVTYPE, COMM,
22    REQUEST, IERROR
23
24 MPI_IALLREDUCE(SENDBUF, RECVBUF, COUNT, DATATYPE, OP, COMM, REQUEST,
25    IERROR)
26    <type> SENDBUF(*), RECVBUF(*)
27    INTEGER COUNT, DATATYPE, OP, COMM, REQUEST, IERROR
28
29 MPI_IALLTOALL(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNT, RECVTYPE,
30    COMM, REQUEST, IERROR)
31    <type> SENDBUF(*), RECVBUF(*)
32    INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVTYPE, COMM, REQUEST, IERROR
33
34 MPI_IALLTOALLV(SENDBUF, SENDCOUNTS, SDISPLS, SENDTYPE, RECVBUF, RECVCOUNTS,
35    RDISPLS, RECVTYPE, COMM, REQUEST, IERROR)
36    <type> SENDBUF(*), RECVBUF(*)
37    INTEGER SENDCOUNTS(*), SDISPLS(*), SENDTYPE, RECVCOUNTS(*), RDISPLS(*),
38    RECVTYPE, COMM, REQUEST, IERROR
39
40 MPI_IALLTOALLW(SENDBUF, SENDCOUNTS, SDISPLS, SENDTYPES, RECVBUF,
41    RECVCOUNTS, RDISPLS, RECVTYPES, COMM, REQUEST, IERROR)
42    <type> SENDBUF(*), RECVBUF(*)
43    INTEGER SENDCOUNTS(*), SDISPLS(*), SENDTYPES(*), RECVCOUNTS(*),
44    RDISPLS(*), RECVTYPES(*), COMM, REQUEST, IERROR
45
46 MPI_IBARRIER(COMM, REQUEST, IERROR)
47    INTEGER COMM, REQUEST, IERROR
48
49 MPI_IBCAST(BUFFER, COUNT, DATATYPE, ROOT, COMM, REQUEST, IERROR)
50    <type> BUFFER(*)
51    INTEGER COUNT, DATATYPE, ROOT, COMM, REQUEST, IERROR

```

```

MPI_IEXSCAN(SENDBUF, RECVBUF, COUNT, DATATYPE, OP, COMM, REQUEST, IERROR) 1
  <type> SENDBUF(*), RECVBUF(*) 2
  INTEGER COUNT, DATATYPE, OP, COMM, REQUEST, IERROR 3
MPI_IGATHER(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNT, RECVTYPE, 4
  ROOT, COMM, REQUEST, IERROR) 5
  <type> SENDBUF(*), RECVBUF(*) 6
  INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVTYPE, ROOT, COMM, REQUEST, 7
  IERROR 8
MPI_IGATHERV(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNTS, DISPLS, 9
  RECVTYPE, ROOT, COMM, REQUEST, IERROR) 10
  <type> SENDBUF(*), RECVBUF(*) 11
  INTEGER SENDCOUNT, SENDTYPE, RECVCOUNTS(*), DISPLS(*), RECVTYPE, ROOT, 12
  COMM, REQUEST, IERROR 13
MPI_IREDUCE(SENDBUF, RECVBUF, COUNT, DATATYPE, OP, ROOT, COMM, REQUEST, 14
  IERROR) 15
  <type> SENDBUF(*), RECVBUF(*) 16
  INTEGER COUNT, DATATYPE, OP, ROOT, COMM, REQUEST, IERROR 17
MPI_IREDUCE_SCATTER(SENDBUF, RECVBUF, RECVCOUNTS, DATATYPE, OP, COMM, 18
  REQUEST, IERROR) 19
  <type> SENDBUF(*), RECVBUF(*) 20
  INTEGER RECVCOUNTS(*), DATATYPE, OP, COMM, REQUEST, IERROR 21
MPI_IREDUCE_SCATTER_BLOCK(SENDBUF, RECVBUF, RECVCOUNT, DATATYPE, OP, COMM, 22
  REQUEST, IERROR) 23
  <type> SENDBUF(*), RECVBUF(*) 24
  INTEGER RECVCOUNT, DATATYPE, OP, COMM, REQUEST, IERROR 25
MPI_ISCAN(SENDBUF, RECVBUF, COUNT, DATATYPE, OP, COMM, REQUEST, IERROR) 26
  <type> SENDBUF(*), RECVBUF(*) 27
  INTEGER COUNT, DATATYPE, OP, COMM, REQUEST, IERROR 28
MPI_ISCATTER(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNT, RECVTYPE, 29
  ROOT, COMM, REQUEST, IERROR) 30
  <type> SENDBUF(*), RECVBUF(*) 31
  INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVTYPE, ROOT, COMM, REQUEST, 32
  IERROR 33
MPI_ISCATTERV(SENDBUF, SENDCOUNTS, DISPLS, SENDTYPE, RECVBUF, RECVCOUNT, 34
  RECVTYPE, ROOT, COMM, REQUEST, IERROR) 35
  <type> SENDBUF(*), RECVBUF(*) 36
  INTEGER SENDCOUNTS(*), DISPLS(*), SENDTYPE, RECVCOUNT, RECVTYPE, ROOT, 37
  COMM, REQUEST, IERROR 38
MPI_OP_COMMUTATIVE(OP, COMMUTE, IERROR) 39
  LOGICAL COMMUTE 40
  INTEGER OP, IERROR 41
MPI_OP_CREATE( USER_FN, COMMUTE, OP, IERROR) 42

```

```

1      EXTERNAL USER_FN
2      LOGICAL COMMUTE
3      INTEGER OP, IERROR
4
5      MPI_OP_FREE(OP, IERROR)
6      INTEGER OP, IERROR
7
8      MPI_REDUCE(SENDBUF, RECVBUFF, COUNT, DATATYPE, OP, ROOT, COMM, IERROR)
9      <type> SENDBUF(*), RECVBUFF(*)
10     INTEGER COUNT, DATATYPE, OP, ROOT, COMM, IERROR
11
12     MPI_REDUCE_INIT(SENDBUF, RECVBUFF, COUNT, DATATYPE, OP, ROOT, COMM, INFO,
13     REQUEST, IERROR)
14     <type> SENDBUF(*), RECVBUFF(*)
15     INTEGER COUNT, DATATYPE, OP, ROOT, COMM, INFO, REQUEST, IERROR
16
17     MPI_REDUCE_LOCAL(INBUF, INOUTBUF, COUNT, DATATYPE, OP, IERROR)
18     <type> INBUF(*), INOUTBUF(*)
19     INTEGER COUNT, DATATYPE, OP, IERROR
20
21     MPI_REDUCE_SCATTER(SENDBUF, RECVBUFF, RECVCOUNTS, DATATYPE, OP, COMM,
22     IERROR)
23     <type> SENDBUF(*), RECVBUFF(*)
24     INTEGER RECVCOUNTS(*), DATATYPE, OP, COMM, IERROR
25
26     MPI_REDUCE_SCATTER_BLOCK(SENDBUF, RECVBUFF, RECVCOUNT, DATATYPE, OP, COMM,
27     IERROR)
28     <type> SENDBUF(*), RECVBUFF(*)
29     INTEGER RECVCOUNT, DATATYPE, OP, COMM, IERROR
30
31     MPI_REDUCE_SCATTER_BLOCK_INIT(SENDBUF, RECVBUFF, RECVCOUNT, DATATYPE, OP,
32     COMM, INFO, REQUEST, IERROR)
33     <type> SENDBUF(*), RECVBUFF(*)
34     INTEGER RECVCOUNT, DATATYPE, OP, COMM, INFO, REQUEST, IERROR
35
36     MPI_REDUCE_SCATTER_INIT(SENDBUF, RECVBUFF, RECVCOUNTS, DATATYPE, OP, COMM,
37     INFO, REQUEST, IERROR)
38     <type> SENDBUF(*), RECVBUFF(*)
39     INTEGER RECVCOUNTS(*), DATATYPE, OP, COMM, INFO, REQUEST, IERROR
40
41     MPI_SCAN(SENDBUF, RECVBUFF, COUNT, DATATYPE, OP, COMM, IERROR)
42     <type> SENDBUF(*), RECVBUFF(*)
43     INTEGER COUNT, DATATYPE, OP, COMM, IERROR
44
45     MPI_SCAN_INIT(SENDBUF, RECVBUFF, COUNT, DATATYPE, OP, COMM, INFO, REQUEST,
46     IERROR)
47     <type> SENDBUF(*), RECVBUFF(*)
48     INTEGER COUNT, DATATYPE, OP, COMM, INFO, REQUEST, IERROR

```

```

MPI_SCATTERV(SENDBUF, SENDCOUNTS, DISPLS, SENDTYPE, RECVBUF, RECVCOUNT,
             RECVTYPE, ROOT, COMM, IERROR)
    <type> SENDBUF(*), RECVBUF(*)
    INTEGER SENDCOUNTS(*), DISPLS(*), SENDTYPE, RECVCOUNT, RECVTYPE, ROOT,
             COMM, IERROR

MPI_SCATTERV_INIT(SENDBUF, SENDCOUNTS, DISPLS, SENDTYPE, RECVBUF,
                  RECVCOUNT, RECVTYPE, ROOT, COMM, INFO, REQUEST, IERROR)
    <type> SENDBUF(*), RECVBUF(*)
    INTEGER SENDCOUNTS(*), DISPLS(*), SENDTYPE, RECVCOUNT, RECVTYPE, ROOT,
             COMM, INFO, REQUEST, IERROR

MPI_SCATTER_INIT(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNT,
                 RECVTYPE, ROOT, COMM, INFO, REQUEST, IERROR)
    <type> SENDBUF(*), RECVBUF(*)
    INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVTYPE, ROOT, COMM, INFO,
             REQUEST, IERROR

A.4.4 Groups, Contexts, Communicators, and Caching Fortran Bindings

MPI_COMM_COMPARE(COMM1, COMM2, RESULT, IERROR)
    INTEGER COMM1, COMM2, RESULT, IERROR

MPI_COMM_CREATE(COMM, GROUP, NEWCOMM, IERROR)
    INTEGER COMM, GROUP, NEWCOMM, IERROR

MPI_COMM_CREATE_GROUP(COMM, GROUP, TAG, NEWCOMM, IERROR)
    INTEGER COMM, GROUP, TAG, NEWCOMM, IERROR

MPI_COMM_CREATE_KEYVAL(COMM_COPY_ATTR_FN, COMM_DELETE_ATTR_FN, COMM_KEYVAL,
                       EXTRA_STATE, IERROR)
    EXTERNAL COMM_COPY_ATTR_FN, COMM_DELETE_ATTR_FN
    INTEGER COMM_KEYVAL, IERROR
    INTEGER(KIND=MPI_ADDRESS_KIND) EXTRA_STATE

MPI_COMM_DELETE_ATTR(COMM, COMM_KEYVAL, IERROR)
    INTEGER COMM, COMM_KEYVAL, IERROR

MPI_COMM_DUP(COMM, NEWCOMM, IERROR)
    INTEGER COMM, NEWCOMM, IERROR

MPI_COMM_DUP_FN(OLDCOMM, COMM_KEYVAL, EXTRA_STATE, ATTRIBUTE_VAL_IN,
                ATTRIBUTE_VAL_OUT, FLAG, IERROR)
    INTEGER OLDCOMM, COMM_KEYVAL, IERROR
    INTEGER(KIND=MPI_ADDRESS_KIND) EXTRA_STATE, ATTRIBUTE_VAL_IN,
                ATTRIBUTE_VAL_OUT
    LOGICAL FLAG

MPI_COMM_DUP_WITH_INFO(COMM, INFO, NEWCOMM, IERROR)
    INTEGER COMM, INFO, NEWCOMM, IERROR

MPI_COMM_FREE(COMM, IERROR)

```

```
1     INTEGER COMM, IERROR
2
3     MPI_COMM_FREE_KEYVAL(COMM_KEYVAL, IERROR)
4     INTEGER COMM_KEYVAL, IERROR
5
6     MPI_COMM_GET_ATTR(COMM, COMM_KEYVAL, ATTRIBUTE_VAL, FLAG, IERROR)
7     INTEGER COMM, COMM_KEYVAL, IERROR
8     INTEGER(KIND=MPI_ADDRESS_KIND) ATTRIBUTE_VAL
9     LOGICAL FLAG
10
11    MPI_COMM_GET_INFO(COMM, INFO_USED, IERROR)
12    INTEGER COMM, INFO_USED, IERROR
13
14    MPI_COMM_GET_NAME(COMM, COMM_NAME, RESULTLEN, IERROR)
15    INTEGER COMM, RESULTLEN, IERROR
16    CHARACTER*(*) COMM_NAME
17
18    MPI_COMM_GROUP(COMM, GROUP, IERROR)
19    INTEGER COMM, GROUP, IERROR
20
21    MPI_COMM_IDUP(COMM, NEWCOMM, REQUEST, IERROR)
22    INTEGER COMM, NEWCOMM, REQUEST, IERROR
23
24    MPI_COMM_IDUP_WITH_INFO(COMM, INFO, NEWCOMM, REQUEST, IERROR)
25    INTEGER COMM, INFO, NEWCOMM, REQUEST, IERROR
26
27    MPI_COMM_NULL_COPY_FN(OLDCOMM, COMM_KEYVAL, EXTRA_STATE, ATTRIBUTE_VAL_IN,
28    ATTRIBUTE_VAL_OUT, FLAG, IERROR)
29    INTEGER OLDCOMM, COMM_KEYVAL, IERROR
30    INTEGER(KIND=MPI_ADDRESS_KIND) EXTRA_STATE, ATTRIBUTE_VAL_IN,
31    ATTRIBUTE_VAL_OUT
32    LOGICAL FLAG
33
34    MPI_COMM_NULL_DELETE_FN(COMM, COMM_KEYVAL, ATTRIBUTE_VAL, EXTRA_STATE,
35    IERROR)
36    INTEGER COMM, COMM_KEYVAL, IERROR
37    INTEGER(KIND=MPI_ADDRESS_KIND) ATTRIBUTE_VAL, EXTRA_STATE
38
39    MPI_COMM_RANK(COMM, RANK, IERROR)
40    INTEGER COMM, RANK, IERROR
41
42    MPI_COMM_REMOTE_GROUP(COMM, GROUP, IERROR)
43    INTEGER COMM, GROUP, IERROR
44
45    MPI_COMM_REMOTE_SIZE(COMM, SIZE, IERROR)
46    INTEGER COMM, SIZE, IERROR
47
48    MPI_COMM_SET_ATTR(COMM, COMM_KEYVAL, ATTRIBUTE_VAL, IERROR)
49    INTEGER COMM, COMM_KEYVAL, IERROR
50    INTEGER(KIND=MPI_ADDRESS_KIND) ATTRIBUTE_VAL
51
52    MPI_COMM_SET_INFO(COMM, INFO, IERROR)
53    INTEGER COMM, INFO, IERROR
54
55    MPI_COMM_SET_NAME(COMM, COMM_NAME, IERROR)
```

INTEGER COMM, IERROR	1
CHARACTER*(*) COMM_NAME	2
	3
MPI_COMM_SIZE(COMM, SIZE, IERROR)	4
INTEGER COMM, SIZE, IERROR	5
	6
MPI_COMM_SPLIT(COMM, COLOR, KEY, NEWCOMM, IERROR)	7
INTEGER COMM, COLOR, KEY, NEWCOMM, IERROR	8
	9
MPI_COMM_SPLIT_TYPE(COMM, SPLIT_TYPE, KEY, INFO, NEWCOMM, IERROR)	10
INTEGER COMM, SPLIT_TYPE, KEY, INFO, NEWCOMM, IERROR	11
	12
MPI_COMM_TEST_INTER(COMM, FLAG, IERROR)	13
INTEGER COMM, IERROR	14
LOGICAL FLAG	15
	16
MPI_GROUP_COMPARE(GROUP1, GROUP2, RESULT, IERROR)	17
INTEGER GROUP1, GROUP2, RESULT, IERROR	18
	19
MPI_GROUP_DIFFERENCE(GROUP1, GROUP2, NEWGROUP, IERROR)	20
INTEGER GROUP1, GROUP2, NEWGROUP, IERROR	21
	22
MPI_GROUP_EXCL(GROUP, N, RANKS, NEWGROUP, IERROR)	23
INTEGER GROUP, N, RANKS(*), NEWGROUP, IERROR	24
	25
MPI_GROUP_FREE(GROUP, IERROR)	26
INTEGER GROUP, IERROR	27
	28
MPI_GROUP_INCL(GROUP, N, RANKS, NEWGROUP, IERROR)	29
INTEGER GROUP, N, RANKS(*), NEWGROUP, IERROR	30
	31
MPI_GROUP_INTERSECTION(GROUP1, GROUP2, NEWGROUP, IERROR)	32
INTEGER GROUP1, GROUP2, NEWGROUP, IERROR	33
	34
MPI_GROUP_RANGE_EXCL(GROUP, N, RANGES, NEWGROUP, IERROR)	35
INTEGER GROUP, N, RANGES(3,*), NEWGROUP, IERROR	36
	37
MPI_GROUP_RANGE_INCL(GROUP, N, RANGES, NEWGROUP, IERROR)	38
INTEGER GROUP, N, RANGES(3,*), NEWGROUP, IERROR	39
	40
MPI_GROUP_RANK(GROUP, RANK, IERROR)	41
INTEGER GROUP, RANK, IERROR	42
	43
MPI_GROUP_SIZE(GROUP, SIZE, IERROR)	44
INTEGER GROUP, SIZE, IERROR	45
	46
MPI_GROUP_TRANSLATE_RANKS(GROUP1, N, RANKS1, GROUP2, RANKS2, IERROR)	47
INTEGER GROUP1, N, RANKS1(*), GROUP2, RANKS2(*), IERROR	48
	49
MPI_GROUP_UNION(GROUP1, GROUP2, NEWGROUP, IERROR)	50
INTEGER GROUP1, GROUP2, NEWGROUP, IERROR	51
	52
MPI_INTERCOMM_CREATE(LOCAL_COMM, LOCAL_LEADER, PEER_COMM, REMOTE_LEADER, TAG, NEWINTERCOMM, IERROR)	53
INTEGER LOCAL_COMM, LOCAL_LEADER, PEER_COMM, REMOTE_LEADER, TAG,	54
	55

```

1           NEWINTERCOMM, IERROR
2
3 MPI_INTERCOMM_MERGE(INTERCOMM, HIGH, NEWINTRACOMM, IERROR)
4     INTEGER INTERCOMM, NEWINTRACOMM, IERROR
5     LOGICAL HIGH
6
7 MPI_TYPE_CREATE_KEYVAL(TYPE_COPY_ATTR_FN, TYPE_DELETE_ATTR_FN, TYPE_KEYVAL,
8     EXTRA_STATE, IERROR)
9     EXTERNAL TYPE_COPY_ATTR_FN, TYPE_DELETE_ATTR_FN
10    INTEGER TYPE_KEYVAL, IERROR
11    INTEGER(KIND=MPI_ADDRESS_KIND) EXTRA_STATE
12
13 MPI_TYPE_DELETE_ATTR(DATATYPE, TYPE_KEYVAL, IERROR)
14    INTEGER DATATYPE, TYPE_KEYVAL, IERROR
15
16 MPI_TYPE_DUP_FN(OLDTYPE, TYPE_KEYVAL, EXTRA_STATE, ATTRIBUTE_VAL_IN,
17     ATTRIBUTE_VAL_OUT, FLAG, IERROR)
18    INTEGER OLDTYPE, TYPE_KEYVAL, IERROR
19    INTEGER(KIND=MPI_ADDRESS_KIND) EXTRA_STATE, ATTRIBUTE_VAL_IN,
20     ATTRIBUTE_VAL_OUT
21    LOGICAL FLAG
22
23 MPI_TYPE_FREE_KEYVAL(TYPE_KEYVAL, IERROR)
24    INTEGER TYPE_KEYVAL, IERROR
25
26 MPI_TYPE_GET_ATTR(DATATYPE, TYPE_KEYVAL, ATTRIBUTE_VAL, FLAG, IERROR)
27    INTEGER DATATYPE, TYPE_KEYVAL, IERROR
28    INTEGER(KIND=MPI_ADDRESS_KIND) ATTRIBUTE_VAL
29    LOGICAL FLAG
30
31 MPI_TYPE_GET_NAME(DATATYPE, TYPE_NAME, RESULTLEN, IERROR)
32    INTEGER DATATYPE, RESULTLEN, IERROR
33    CHARACTER*(*) TYPE_NAME
34
35 MPI_TYPE_NULL_COPY_FN(OLDTYPE, TYPE_KEYVAL, EXTRA_STATE, ATTRIBUTE_VAL_IN,
36     ATTRIBUTE_VAL_OUT, FLAG, IERROR)
37    INTEGER OLDTYPE, TYPE_KEYVAL, IERROR
38    INTEGER(KIND=MPI_ADDRESS_KIND) EXTRA_STATE, ATTRIBUTE_VAL_IN,
39     ATTRIBUTE_VAL_OUT
40    LOGICAL FLAG
41
42 MPI_TYPE_NULL_DELETE_FN(DATATYPE, TYPE_KEYVAL, ATTRIBUTE_VAL, EXTRA_STATE,
43     IERROR)
44    INTEGER DATATYPE, TYPE_KEYVAL, IERROR
45    INTEGER(KIND=MPI_ADDRESS_KIND) ATTRIBUTE_VAL, EXTRA_STATE
46
47 MPI_TYPE_SET_ATTR(DATATYPE, TYPE_KEYVAL, ATTRIBUTE_VAL, IERROR)
48    INTEGER DATATYPE, TYPE_KEYVAL, IERROR
49    INTEGER(KIND=MPI_ADDRESS_KIND) ATTRIBUTE_VAL
50
51 MPI_TYPE_SET_NAME(DATATYPE, TYPE_NAME, IERROR)
52    INTEGER DATATYPE, IERROR
53    CHARACTER*(*) TYPE_NAME

```



```

MPI_WIN_CREATE_KEYVAL(WIN_COPY_ATTR_FN, WIN_DELETE_ATTR_FN, WIN_KEYVAL,
    EXTRA_STATE, IERROR)
    EXTERNAL WIN_COPY_ATTR_FN, WIN_DELETE_ATTR_FN
    INTEGER WIN_KEYVAL, IERROR
    INTEGER(KIND=MPI_ADDRESS_KIND) EXTRA_STATE
MPI_WIN_DELETE_ATTR(WIN, WIN_KEYVAL, IERROR)
    INTEGER WIN, WIN_KEYVAL, IERROR
MPI_WIN_DUP_FN(OLDWIN, WIN_KEYVAL, EXTRA_STATE, ATTRIBUTE_VAL_IN,
    ATTRIBUTE_VAL_OUT, FLAG, IERROR)
    INTEGER OLDWIN, WIN_KEYVAL, IERROR
    INTEGER(KIND=MPI_ADDRESS_KIND) EXTRA_STATE, ATTRIBUTE_VAL_IN,
    ATTRIBUTE_VAL_OUT
    LOGICAL FLAG
MPI_WIN_FREE_KEYVAL(WIN_KEYVAL, IERROR)
    INTEGER WIN_KEYVAL, IERROR
MPI_WIN_GET_ATTR(WIN, WIN_KEYVAL, ATTRIBUTE_VAL, FLAG, IERROR)
    INTEGER WIN, WIN_KEYVAL, IERROR
    INTEGER(KIND=MPI_ADDRESS_KIND) ATTRIBUTE_VAL
    LOGICAL FLAG
MPI_WIN_GET_NAME(WIN, WIN_NAME, RESULTLEN, IERROR)
    INTEGER WIN, RESULTLEN, IERROR
    CHARACTER*(*) WIN_NAME
MPI_WIN_NULL_COPY_FN(OLDWIN, WIN_KEYVAL, EXTRA_STATE, ATTRIBUTE_VAL_IN,
    ATTRIBUTE_VAL_OUT, FLAG, IERROR)
    INTEGER OLDWIN, WIN_KEYVAL, IERROR
    INTEGER(KIND=MPI_ADDRESS_KIND) EXTRA_STATE, ATTRIBUTE_VAL_IN,
    ATTRIBUTE_VAL_OUT
    LOGICAL FLAG
MPI_WIN_NULL_DELETE_FN(WIN, WIN_KEYVAL, ATTRIBUTE_VAL, EXTRA_STATE, IERROR)
    INTEGER WIN, WIN_KEYVAL, IERROR
    INTEGER(KIND=MPI_ADDRESS_KIND) ATTRIBUTE_VAL, EXTRA_STATE
MPI_WIN_SET_ATTR(WIN, WIN_KEYVAL, ATTRIBUTE_VAL, IERROR)
    INTEGER WIN, WIN_KEYVAL, IERROR
    INTEGER(KIND=MPI_ADDRESS_KIND) ATTRIBUTE_VAL
MPI_WIN_SET_NAME(WIN, WIN_NAME, IERROR)
    INTEGER WIN, IERROR
    CHARACTER*(*) WIN_NAME
A.4.5 Process Topologies Fortran Bindings
MPI_CARTDIM_GET(COMM, NDIMS, IERROR)
    INTEGER COMM, NDIMS, IERROR

```

```

1 MPI_CART_COORDS(COMM, RANK, MAXDIMS, COORDS, IERROR)
2     INTEGER COMM, RANK, MAXDIMS, COORDS(*), IERROR
3
4 MPI_CART_CREATE(COMM_OLD, NDIMS, DIMS, PERIODS, REORDER, COMM_CART, IERROR)
5     INTEGER COMM_OLD, NDIMS, DIMS(*), COMM_CART, IERROR
6     LOGICAL PERIODS(*), REORDER
7
8 MPI_CART_GET(COMM, MAXDIMS, DIMS, PERIODS, COORDS, IERROR)
9     INTEGER COMM, MAXDIMS, DIMS(*), COORDS(*), IERROR
10    LOGICAL PERIODS(*)
11
12 MPI_CART_MAP(COMM, NDIMS, DIMS, PERIODS, NEWRANK, IERROR)
13    INTEGER COMM, NDIMS, DIMS(*), NEWRANK, IERROR
14    LOGICAL PERIODS(*)
15
16 MPI_CART_RANK(COMM, COORDS, RANK, IERROR)
17    INTEGER COMM, COORDS(*), RANK, IERROR
18
19 MPI_CART_SHIFT(COMM, DIRECTION, DISP, RANK_SOURCE, RANK_DEST, IERROR)
20    INTEGER COMM, DIRECTION, DISP, RANK_SOURCE, RANK_DEST, IERROR
21
22 MPI_CART_SUB(COMM, REMAIN_DIMS, NEWCOMM, IERROR)
23    INTEGER COMM, NEWCOMM, IERROR
24    LOGICAL REMAIN_DIMS(*)
25
26 MPI_DIMS_CREATE(NNODES, NDIMS, DIMS, IERROR)
27    INTEGER NNODES, NDIMS, DIMS(*), IERROR
28
29 MPI_DIST_GRAPH_CREATE(COMM_OLD, N, SOURCES, DEGREES, DESTINATIONS, WEIGHTS,
30     INFO, REORDER, COMM_DIST_GRAPH, IERROR)
31    INTEGER COMM_OLD, N, SOURCES(*), DEGREES(*), DESTINATIONS(*),
32     WEIGHTS(*), INFO, COMM_DIST_GRAPH, IERROR
33    LOGICAL REORDER
34
35 MPI_DIST_GRAPH_CREATE_ADJACENT(COMM_OLD, INDEGREE, SOURCES, SOURCEWEIGHTS,
36     OUTDEGREE, DESTINATIONS, DESTWEIGHTS, INFO, REORDER,
37     COMM_DIST_GRAPH, IERROR)
38    INTEGER COMM_OLD, INDEGREE, SOURCES(*), SOURCEWEIGHTS(*), OUTDEGREE,
39     DESTINATIONS(*), DESTWEIGHTS(*), INFO, COMM_DIST_GRAPH,
40     IERROR
41    LOGICAL REORDER
42
43 MPI_DIST_GRAPH_NEIGHBORS(COMM, MAXINDEGREE, SOURCES, SOURCEWEIGHTS,
44     MAXOUTDEGREE, DESTINATIONS, DESTWEIGHTS, IERROR)
45    INTEGER COMM, MAXINDEGREE, SOURCES(*), SOURCEWEIGHTS(*), MAXOUTDEGREE,
46     DESTINATIONS(*), DESTWEIGHTS(*), IERROR
47
48 MPI_DIST_GRAPH_NEIGHBORS_COUNT(COMM, INDEGREE, OUTDEGREE, WEIGHTED, IERROR)
49    INTEGER COMM, INDEGREE, OUTDEGREE, IERROR
50    LOGICAL WEIGHTED
51
52 MPI_GRAPHDIMS_GET(COMM, NNODES, NEDGES, IERROR)
53    INTEGER COMM, NNODES, NEDGES, IERROR

```

```

MPI_GRAPH_CREATE(COMM_OLD, NNODES, INDEX, EDGES, REORDER, COMM_GRAPH,
                IERROR)
    INTEGER COMM_OLD, NNODES, INDEX(*), EDGES(*), COMM_GRAPH, IERROR
    LOGICAL REORDER
MPI_GRAPH_GET(COMM, MAXINDEX, MAXEDGES, INDEX, EDGES, IERROR)
    INTEGER COMM, MAXINDEX, MAXEDGES, INDEX(*), EDGES(*), IERROR
MPI_GRAPH_MAP(COMM, NNODES, INDEX, EDGES, NEWRANK, IERROR)
    INTEGER COMM, NNODES, INDEX(*), EDGES(*), NEWRANK, IERROR
MPI_GRAPH_NEIGHBORS(COMM, RANK, MAXNEIGHBORS, NEIGHBORS, IERROR)
    INTEGER COMM, RANK, MAXNEIGHBORS, NEIGHBORS(*), IERROR
MPI_GRAPH_NEIGHBORS_COUNT(COMM, RANK, NNEIGHBORS, IERROR)
    INTEGER COMM, RANK, NNEIGHBORS, IERROR
MPI_INEIGHBOR_ALLGATHER(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUFF, RECVCOUNT,
                      RECVMODE, COMM, REQUEST, IERROR)
    <type> SENDBUF(*), RECVBUFF(*)
    INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVMODE, COMM, REQUEST, IERROR
MPI_INEIGHBOR_ALLGATHERV(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUFF, RECVCOUNTS,
                       DISPLS, RECVMODE, COMM, REQUEST, IERROR)
    <type> SENDBUF(*), RECVBUFF(*)
    INTEGER SENDCOUNT, SENDTYPE, RECVCOUNTS(*), DISPLS(*), RECVMODE, COMM,
    REQUEST, IERROR
MPI_INEIGHBOR_ALLTOALL(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUFF, RECVCOUNT,
                      RECVMODE, COMM, REQUEST, IERROR)
    <type> SENDBUF(*), RECVBUFF(*)
    INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVMODE, COMM, REQUEST, IERROR
MPI_INEIGHBOR_ALLTOALLV(SENDBUF, SENDCOUNTS, SDISPLS, SENDTYPE, RECVBUFF,
                       RECVCOUNTS, RDISPLS, RECVMODE, COMM, REQUEST, IERROR)
    <type> SENDBUF(*), RECVBUFF(*)
    INTEGER SENDCOUNTS(*), SDISPLS(*), SENDTYPE, RECVCOUNTS(*), RDISPLS(*),
    RECVMODE, COMM, REQUEST, IERROR
MPI_INEIGHBOR_ALLTOALLW(SENDBUF, SENDCOUNTS, SDISPLS, SENDTYPES, RECVBUFF,
                       RECVCOUNTS, RDISPLS, RECVMODES, COMM, REQUEST, IERROR)
    <type> SENDBUF(*), RECVBUFF(*)
    INTEGER(KIND=MPI_ADDRESS_KIND) SDISPLS(*), RDISPLS(*)
    INTEGER SENDCOUNTS(*), SENDTYPES(*), RECVCOUNTS(*), RECVMODES(*), COMM,
    REQUEST, IERROR
MPI_NEIGHBOR_ALLGATHER(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUFF, RECVCOUNT,
                      RECVMODE, COMM, IERROR)
    <type> SENDBUF(*), RECVBUFF(*)
    INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVMODE, COMM, IERROR
MPI_NEIGHBOR_ALLGATHERV(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUFF, RECVCOUNTS,
                       DISPLS, RECVMODE, COMM, IERROR)

```

```

1     <type> SENDBUF(*), RECVBUF(*)
2     INTEGER SENDCOUNT, SENDTYPE, RECVCOUNTS(*), DISPLS(*), RECVTYPE, COMM,
3         IERROR
4
5     MPI_NEIGHBOR_ALLGATHERV_INIT(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF,
6         RECVCOUNTS, DISPLS, RECVTYPE, COMM, INFO, REQUEST, IERROR)
7     <type> SENDBUF(*), RECVBUF(*)
8     INTEGER SENDCOUNT, SENDTYPE, RECVCOUNTS(*), DISPLS(*), RECVTYPE, COMM,
9         INFO, REQUEST, IERROR
10
11    MPI_NEIGHBOR_ALLGATHER_INIT(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF,
12        RECVCOUNT, RECVTYPE, COMM, INFO, REQUEST, IERROR)
13    <type> SENDBUF(*), RECVBUF(*)
14    INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVTYPE, COMM, INFO, REQUEST,
15        IERROR
16
17    MPI_NEIGHBOR_ALLTOALL(SENDBUF, SENDCOUNT, SENDTYPE, RECVBUF, RECVCOUNT,
18        RECVTYPE, COMM, IERROR)
19    <type> SENDBUF(*), RECVBUF(*)
20    INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVTYPE, COMM, IERROR
21
22    MPI_NEIGHBOR_ALLTOALLV(SENDBUF, SENDCOUNTS, SDISPLS, SENDTYPE, RECVBUF,
23        RECVCOUNTS, RDISPLS, RECVTYPE, COMM, IERROR)
24    <type> SENDBUF(*), RECVBUF(*)
25    INTEGER SENDCOUNTS(*), SDISPLS(*), SENDTYPE, RECVCOUNTS(*), RDISPLS(*),
26        RECVTYPE, COMM, IERROR
27
28    MPI_NEIGHBOR_ALLTOALLV_INIT(SENDBUF, SENDCOUNTS, SDISPLS, SENDTYPE,
29        RECVBUF, RECVCOUNTS, RDISPLS, RECVTYPE, COMM, INFO, REQUEST,
30        IERROR)
31    <type> SENDBUF(*), RECVBUF(*)
32    INTEGER SENDCOUNTS(*), SDISPLS(*), SENDTYPE, RECVCOUNTS(*), RDISPLS(*),
33        RECVTYPE, COMM, INFO, REQUEST, IERROR
34
35    MPI_NEIGHBOR_ALLTOALLW(SENDBUF, SENDCOUNTS, SDISPLS, SENDTYPES, RECVBUF,
36        RECVCOUNTS, RDISPLS, RECVTYPES, COMM, IERROR)
37    <type> SENDBUF(*), RECVBUF(*)
38    INTEGER(KIND=MPI_ADDRESS_KIND) SDISPLS(*), RDISPLS(*)
39    INTEGER SENDCOUNTS(*), SENDTYPES(*), RECVCOUNTS(*), RECVTYPES(*), COMM,
40        IERROR
41
42    MPI_NEIGHBOR_ALLTOALLW_INIT(SENDBUF, SENDCOUNTS, SDISPLS, SENDTYPES,
43        RECVBUF, RECVCOUNTS, RDISPLS, RECVTYPES, COMM, INFO, REQUEST,
44        IERROR)
45    <type> SENDBUF(*), RECVBUF(*)
46    INTEGER(KIND=MPI_ADDRESS_KIND) SDISPLS(*), RDISPLS(*)
47    INTEGER SENDCOUNTS(*), SENDTYPES(*), RECVCOUNTS(*), RECVTYPES(*), COMM,
48        INFO, REQUEST, IERROR

```

<type> SENDBUF(*), RECVBUF(*)	1
INTEGER SENDCOUNT, SENDTYPE, RECVCOUNT, RECVMODE, COMM, INFO, REQUEST,	2
IERROR	3
	4
MPI_TOPO_TEST(COMM, STATUS, IERROR)	5
INTEGER COMM, STATUS, IERROR	6
	7
A.4.6 MPI Environmental Management Fortran Bindings	8
	9
DOUBLE PRECISION MPI_WTICK()	10
	11
DOUBLE PRECISION MPI_WTIME()	12
	13
MPI_ABORT(COMM, ERRORCODE, IERROR)	14
INTEGER COMM, ERRORCODE, IERROR	15
	16
MPI_ADD_ERROR_CLASS(ERRORCLASS, IERROR)	17
INTEGER ERRORCLASS, IERROR	18
	19
MPI_ADD_ERROR_CODE(ERRORCLASS, ERRORCODE, IERROR)	20
INTEGER ERRORCLASS, ERRORCODE, IERROR	21
	22
MPI_ADD_ERROR_STRING(ERRORCODE, STRING, IERROR)	23
INTEGER ERRORCODE	24
CHARACTER*(*) STRING	25
INTEGER IERROR	26
	27
MPI_ALLOC_MEM(SIZE, INFO, BASEPTR, IERROR)	28
INTEGER(KIND=MPI_ADDRESS_KIND) SIZE	29
INTEGER INFO	30
INTEGER(KIND=MPI_ADDRESS_KIND) BASEPTR	31
INTEGER IERROR	32
	33
MPI_COMM_CALL_ERRHANDLER(COMM, ERRORCODE, IERROR)	34
INTEGER COMM, ERRORCODE, IERROR	35
	36
MPI_COMM_CREATE_ERRHANDLER(COMM_ERRHANDLER_FN, ERRHANDLER, IERROR)	37
EXTERNAL COMM_ERRHANDLER_FN	38
INTEGER ERRHANDLER, IERROR	39
	40
MPI_COMM_GET_ERRHANDLER(COMM, ERRHANDLER, IERROR)	41
INTEGER COMM, ERRHANDLER, IERROR	42
	43
MPI_COMM_SET_ERRHANDLER(COMM, ERRHANDLER, IERROR)	44
INTEGER COMM, ERRHANDLER, IERROR	45
	46
MPI_ERRHANDLER_FREE(ERRHANDLER, IERROR)	47
INTEGER ERRHANDLER, IERROR	48
MPI_ERROR_CLASS(ERRORCODE, ERRORCLASS, IERROR)	
INTEGER ERRORCODE, ERRORCLASS, IERROR	
MPI_ERROR_STRING(ERRORCODE, STRING, RESULTLEN, IERROR)	
INTEGER ERRORCODE	

```
1     CHARACTER*(*) STRING
2     INTEGER RESULTLEN, IERROR
3
4     MPI_FILE_CALL_ERRHANDLER(FH, ERRORCODE, IERROR)
5     INTEGER FH, ERRORCODE, IERROR
6
7     MPI_FILE_CREATE_ERRHANDLER(FILE_ERRHANDLER_FN, ERRHANDLER, IERROR)
8     EXTERNAL FILE_ERRHANDLER_FN
9     INTEGER ERRHANDLER, IERROR
10
11    MPI_FILE_GET_ERRHANDLER(FILE, ERRHANDLER, IERROR)
12    INTEGER FILE, ERRHANDLER, IERROR
13
14    MPI_FILE_SET_ERRHANDLER(FILE, ERRHANDLER, IERROR)
15    INTEGER FILE, ERRHANDLER, IERROR
16
17    MPI_FINALIZE(IERROR)
18    INTEGER IERROR
19
20    MPI_FINALIZED(FLAG, IERROR)
21    LOGICAL FLAG
22    INTEGER IERROR
23
24    MPI_FREE_MEM(BASE, IERROR)
25    <type> BASE(*)
26    INTEGER IERROR
27
28    MPI_GET_LIBRARY_VERSION(VERSION, RESULTLEN, IERROR)
29    CHARACTER*(*) VERSION
30    INTEGER RESULTLEN, IERROR
31
32    MPI_GET_PROCESSOR_NAME(NAME, RESULTLEN, IERROR)
33    CHARACTER*(*) NAME
34    INTEGER RESULTLEN, IERROR
35
36    MPI_GET_VERSION(VERSION, SUBVERSION, IERROR)
37    INTEGER VERSION, SUBVERSION, IERROR
38
39    MPI_INIT(IERROR)
40    INTEGER IERROR
41
42    MPI_INITIALIZED(FLAG, IERROR)
43    LOGICAL FLAG
44    INTEGER IERROR
45
46    MPI_WIN_CALL_ERRHANDLER(WIN, ERRORCODE, IERROR)
47    INTEGER WIN, ERRORCODE, IERROR
48
49    MPI_WIN_CREATE_ERRHANDLER(WIN_ERRHANDLER_FN, ERRHANDLER, IERROR)
50    EXTERNAL WIN_ERRHANDLER_FN
51    INTEGER ERRHANDLER, IERROR
52
53    MPI_WIN_GET_ERRHANDLER(WIN, ERRHANDLER, IERROR)
54    INTEGER WIN, ERRHANDLER, IERROR
```

MPI_WIN_SET_ERRHANDLER(WIN, ERRHANDLER, IERROR)	1
INTEGER WIN, ERRHANDLER, IERROR	2
	3
	4
A.4.7 The Info Object Fortran Bindings	5
MPI_INFO_CREATE(INFO, IERROR)	6
INTEGER INFO, IERROR	7
	8
MPI_INFO_DELETE(INFO, KEY, IERROR)	9
INTEGER INFO, IERROR	10
CHARACTER*(*) KEY	11
	12
MPI_INFO_DUP(INFO, NEWINFO, IERROR)	13
INTEGER INFO, NEWINFO, IERROR	14
	15
MPI_INFO_FREE(INFO, IERROR)	16
INTEGER INFO, IERROR	17
MPI_INFO_GET(INFO, KEY, VALUELEN, VALUE, FLAG, IERROR)	18
INTEGER INFO, VALUELEN, IERROR	19
CHARACTER*(*) KEY, VALUE	20
LOGICAL FLAG	21
	22
MPI_INFO_GET_NKEYS(INFO, NKEYS, IERROR)	23
INTEGER INFO, NKEYS, IERROR	24
	25
MPI_INFO_GET_NTHKEY(INFO, N, KEY, IERROR)	26
INTEGER INFO, N, IERROR	27
CHARACTER*(*) KEY	28
	29
MPI_INFO_GET_VALUELEN(INFO, KEY, VALUELEN, FLAG, IERROR)	30
INTEGER INFO, VALUELEN, IERROR	31
LOGICAL FLAG	32
CHARACTER*(*) KEY	33
	34
MPI_INFO_SET(INFO, KEY, VALUE, IERROR)	35
INTEGER INFO, IERROR	36
CHARACTER*(*) KEY, VALUE	37
	38
A.4.8 Process Creation and Management Fortran Bindings	39
MPI_CLOSE_PORT(PORT_NAME, IERROR)	40
CHARACTER*(*) PORT_NAME	41
INTEGER IERROR	42
	43
MPI_COMM_ACCEPT(PORT_NAME, INFO, ROOT, COMM, NEWCOMM, IERROR)	44
CHARACTER*(*) PORT_NAME	45
INTEGER INFO, ROOT, COMM, NEWCOMM, IERROR	46
	47
MPI_COMM_CONNECT(PORT_NAME, INFO, ROOT, COMM, NEWCOMM, IERROR)	48
CHARACTER*(*) PORT_NAME	
INTEGER INFO, ROOT, COMM, NEWCOMM, IERROR	

```

1 MPI_COMM_DISCONNECT(COMM, IERROR)
2     INTEGER COMM, IERROR
3
4 MPI_COMM_GET_PARENT(PARENT, IERROR)
5     INTEGER PARENT, IERROR
6
7 MPI_COMM_JOIN(FD, INTERCOMM, IERROR)
8     INTEGER FD, INTERCOMM, IERROR
9
10 MPI_COMM_SPAWN(COMMAND, ARGV, MAXPROCS, INFO, ROOT, COMM, INTERCOMM,
11     ARRAY_OF_ERRCODES, IERROR)
12     CHARACTER*(*) COMMAND, ARGV(*)
13     INTEGER INFO, MAXPROCS, ROOT, COMM, INTERCOMM, ARRAY_OF_ERRCODES(*),
14     IERROR
15
16 MPI_COMM_SPAWN_MULTIPLE(COUNT, ARRAY_OF_COMMANDS, ARRAY_OF_ARGV,
17     ARRAY_OF_MAXPROCS, ARRAY_OF_INFO, ROOT, COMM, INTERCOMM,
18     ARRAY_OF_ERRCODES, IERROR)
19     INTEGER COUNT, ARRAY_OF_INFO(*), ARRAY_OF_MAXPROCS(*), ROOT, COMM,
20     INTERCOMM, ARRAY_OF_ERRCODES(*), IERROR
21     CHARACTER*(*) ARRAY_OF_COMMANDS(*), ARRAY_OF_ARGV(COUNT, *)
22
23 MPI_LOOKUP_NAME(SERVICE_NAME, INFO, PORT_NAME, IERROR)
24     CHARACTER*(*) SERVICE_NAME, PORT_NAME
25     INTEGER INFO, IERROR
26
27 MPI_OPEN_PORT(INFO, PORT_NAME, IERROR)
28     CHARACTER*(*) PORT_NAME
29     INTEGER INFO, IERROR
30
31 MPI_PUBLISH_NAME(SERVICE_NAME, INFO, PORT_NAME, IERROR)
32     INTEGER INFO, IERROR
33     CHARACTER*(*) SERVICE_NAME, PORT_NAME
34
35 MPI_UNPUBLISH_NAME(SERVICE_NAME, INFO, PORT_NAME, IERROR)
36     INTEGER INFO, IERROR
37     CHARACTER*(*) SERVICE_NAME, PORT_NAME
38
39 A.4.9 One-Sided Communications Fortran Bindings
40
41 MPI_ACCUMULATE(ORIGIN_ADDR, ORIGIN_COUNT, ORIGIN_DATATYPE, TARGET_RANK,
42     TARGET_DISP, TARGET_COUNT, TARGET_DATATYPE, OP, WIN, IERROR)
43     <type> ORIGIN_ADDR(*)
44     INTEGER(KIND=MPI_ADDRESS_KIND) TARGET_DISP
45     INTEGER ORIGIN_COUNT, ORIGIN_DATATYPE, TARGET_RANK, TARGET_COUNT,
46     TARGET_DATATYPE, OP, WIN, IERROR
47
48 MPI_COMPARE_AND_SWAP(ORIGIN_ADDR, COMPARE_ADDR, RESULT_ADDR, DATATYPE,
49     TARGET_RANK, TARGET_DISP, WIN, IERROR)
50     <type> ORIGIN_ADDR(*), COMPARE_ADDR(*), RESULT_ADDR(*)
51     INTEGER(KIND=MPI_ADDRESS_KIND) TARGET_DISP
52     INTEGER DATATYPE, TARGET_RANK, WIN, IERROR

```



```

MPI_FETCH_AND_OP(ORIGIN_ADDR, RESULT_ADDR, DATATYPE, TARGET_RANK,      1
                 TARGET_DISP, OP, WIN, IERROR)                        2
    <type> ORIGIN_ADDR(*), RESULT_ADDR(*)                             3
    INTEGER(KIND=MPI_ADDRESS_KIND) TARGET_DISP                       4
    INTEGER DATATYPE, TARGET_RANK, OP, WIN, IERROR                   5
                                                                    6
MPI_GET(ORIGIN_ADDR, ORIGIN_COUNT, ORIGIN_DATATYPE, TARGET_RANK,      7
        TARGET_DISP, TARGET_COUNT, TARGET_DATATYPE, WIN, IERROR)     8
    <type> ORIGIN_ADDR(*)                                             9
    INTEGER(KIND=MPI_ADDRESS_KIND) TARGET_DISP                       10
    INTEGER ORIGIN_COUNT, ORIGIN_DATATYPE, TARGET_RANK, TARGET_COUNT, 11
        TARGET_DATATYPE, WIN, IERROR                                12
                                                                    13
MPI_GET_ACCUMULATE(ORIGIN_ADDR, ORIGIN_COUNT, ORIGIN_DATATYPE, RESULT_ADDR, 14
                  RESULT_COUNT, RESULT_DATATYPE, TARGET_RANK, TARGET_DISP, 15
                  TARGET_COUNT, TARGET_DATATYPE, OP, WIN, IERROR)    16
    <type> ORIGIN_ADDR(*), RESULT_ADDR(*)                             17
    INTEGER(KIND=MPI_ADDRESS_KIND) TARGET_DISP                       18
    INTEGER ORIGIN_COUNT, ORIGIN_DATATYPE, RESULT_COUNT, RESULT_DATATYPE, 19
        TARGET_RANK, TARGET_COUNT, TARGET_DATATYPE, OP, WIN, IERROR 20
                                                                    21
MPI_PUT(ORIGIN_ADDR, ORIGIN_COUNT, ORIGIN_DATATYPE, TARGET_RANK,      22
        TARGET_DISP, TARGET_COUNT, TARGET_DATATYPE, WIN, IERROR)     23
    <type> ORIGIN_ADDR(*)                                             24
    INTEGER(KIND=MPI_ADDRESS_KIND) TARGET_DISP                       25
    INTEGER ORIGIN_COUNT, ORIGIN_DATATYPE, TARGET_RANK, TARGET_COUNT, 26
        TARGET_DATATYPE, WIN, IERROR                                27
                                                                    28
MPI_RACCUMULATE(ORIGIN_ADDR, ORIGIN_COUNT, ORIGIN_DATATYPE, TARGET_RANK, 29
               TARGET_DISP, TARGET_COUNT, TARGET_DATATYPE, OP, WIN, REQUEST, 30
               IERROR)                                               31
    <type> ORIGIN_ADDR(*)                                             32
    INTEGER(KIND=MPI_ADDRESS_KIND) TARGET_DISP                       33
    INTEGER ORIGIN_COUNT, ORIGIN_DATATYPE, TARGET_RANK, TARGET_COUNT, 34
        TARGET_DATATYPE, OP, WIN, REQUEST, IERROR                   35
                                                                    36
MPI_RGET(ORIGIN_ADDR, ORIGIN_COUNT, ORIGIN_DATATYPE, TARGET_RANK,      37
         TARGET_DISP, TARGET_COUNT, TARGET_DATATYPE, WIN, REQUEST, 38
         IERROR)                                                     39
    <type> ORIGIN_ADDR(*)                                             40
    INTEGER(KIND=MPI_ADDRESS_KIND) TARGET_DISP                       41
    INTEGER ORIGIN_COUNT, ORIGIN_DATATYPE, TARGET_RANK, TARGET_COUNT, 42
        TARGET_DATATYPE, WIN, REQUEST, IERROR                       43
                                                                    44
MPI_RGET_ACCUMULATE(ORIGIN_ADDR, ORIGIN_COUNT, ORIGIN_DATATYPE,      45
                   RESULT_ADDR, RESULT_COUNT, RESULT_DATATYPE, TARGET_RANK, 46
                   TARGET_DISP, TARGET_COUNT, TARGET_DATATYPE, OP, WIN, REQUEST, 47
                   IERROR)                                           48
    <type> ORIGIN_ADDR(*), RESULT_ADDR(*)                             49
    INTEGER(KIND=MPI_ADDRESS_KIND) TARGET_DISP                       50

```

```

1      INTEGER ORIGIN_COUNT, ORIGIN_DATATYPE, RESULT_COUNT, RESULT_DATATYPE,
2          TARGET_RANK, TARGET_COUNT, TARGET_DATATYPE, OP, WIN, REQUEST,
3      IERROR
4
5      MPI_RPUT(ORIGIN_ADDR, ORIGIN_COUNT, ORIGIN_DATATYPE, TARGET_RANK,
6          TARGET_DISP, TARGET_COUNT, TARGET_DATATYPE, WIN, REQUEST,
7          IERROR)
8      <type> ORIGIN_ADDR(*)
9      INTEGER(KIND=MPI_ADDRESS_KIND) TARGET_DISP
10     INTEGER ORIGIN_COUNT, ORIGIN_DATATYPE, TARGET_RANK, TARGET_COUNT,
11     TARGET_DATATYPE, WIN, REQUEST, IERROR
12
13     MPI_WIN_ALLOCATE(SIZE, DISP_UNIT, INFO, COMM, BASEPTR, WIN, IERROR)
14     INTEGER DISP_UNIT, INFO, COMM, WIN, IERROR
15     INTEGER(KIND=MPI_ADDRESS_KIND) SIZE, BASEPTR
16
17     If the Fortran compiler provides TYPE(C_PTR), then overloaded by:
18     INTERFACE MPI_WIN_ALLOCATE
19         SUBROUTINE MPI_WIN_ALLOCATE(SIZE, DISP_UNIT, INFO, COMM, BASEPTR, &
20             WIN, IERROR)
21             IMPORT :: MPI_ADDRESS_KIND
22             INTEGER :: DISP_UNIT, INFO, COMM, WIN, IERROR
23             INTEGER(KIND=MPI_ADDRESS_KIND) :: SIZE, BASEPTR
24         END SUBROUTINE
25         SUBROUTINE MPI_WIN_ALLOCATE_CPTR(SIZE, DISP_UNIT, INFO, COMM, BASEPTR, &
26             WIN, IERROR)
27             USE, INTRINSIC :: ISO_C_BINDING, ONLY : C_PTR
28             IMPORT :: MPI_ADDRESS_KIND
29             INTEGER :: DISP_UNIT, INFO, COMM, WIN, IERROR
30             INTEGER(KIND=MPI_ADDRESS_KIND) :: SIZE
31             TYPE(C_PTR) :: BASEPTR
32         END SUBROUTINE
33     END INTERFACE
34
35     MPI_WIN_ALLOCATE_SHARED(SIZE, DISP_UNIT, INFO, COMM, BASEPTR, WIN, IERROR)
36     INTEGER DISP_UNIT, INFO, COMM, WIN, IERROR
37     INTEGER(KIND=MPI_ADDRESS_KIND) SIZE, BASEPTR
38
39     If the Fortran compiler provides TYPE(C_PTR), then overloaded by:
40     INTERFACE MPI_WIN_ALLOCATE_SHARED
41         SUBROUTINE MPI_WIN_ALLOCATE_SHARED(SIZE, DISP_UNIT, INFO, COMM, &
42             BASEPTR, WIN, IERROR)
43             IMPORT :: MPI_ADDRESS_KIND
44             INTEGER :: DISP_UNIT, INFO, COMM, WIN, IERROR
45             INTEGER(KIND=MPI_ADDRESS_KIND) :: SIZE, BASEPTR
46         END SUBROUTINE
47         SUBROUTINE MPI_WIN_ALLOCATE_SHARED_CPTR(SIZE, DISP_UNIT, INFO, COMM, &
48             BASEPTR, WIN, IERROR)
49             USE, INTRINSIC :: ISO_C_BINDING, ONLY : C_PTR
50             IMPORT :: MPI_ADDRESS_KIND
51             INTEGER :: DISP_UNIT, INFO, COMM, WIN, IERROR

```

```
        INTEGER(KIND=MPI_ADDRESS_KIND) ::  SIZE           1
        TYPE(C_PTR) ::  BASEPTR                           2
    END SUBROUTINE                                       3
END INTERFACE                                           4
                                                    5
MPI_WIN_ATTACH(WIN, BASE, SIZE, IERROR)                 6
    INTEGER WIN, IERROR                                  7
    <type> BASE(*)                                       8
    INTEGER (KIND=MPI_ADDRESS_KIND) SIZE                 9
                                                    10
MPI_WIN_COMPLETE(WIN, IERROR)                           11
    INTEGER WIN, IERROR                                  12
                                                    13
MPI_WIN_CREATE(BASE, SIZE, DISP_UNIT, INFO, COMM, WIN, IERROR)
    <type> BASE(*)                                       14
    INTEGER(KIND=MPI_ADDRESS_KIND) SIZE                 15
    INTEGER DISP_UNIT, INFO, COMM, WIN, IERROR          16
                                                    17
MPI_WIN_CREATE_DYNAMIC(INFO, COMM, WIN, IERROR)         18
    INTEGER INFO, COMM, WIN, IERROR                    19
                                                    20
MPI_WIN_DETACH(WIN, BASE, IERROR)                       21
    INTEGER WIN, IERROR                                  22
    <type> BASE(*)                                       23
                                                    24
MPI_WIN_FENCE(ASSERT, WIN, IERROR)                      25
    INTEGER ASSERT, WIN, IERROR                        26
                                                    27
MPI_WIN_FLUSH(RANK, WIN, IERROR)                        28
    INTEGER RANK, WIN, IERROR                          29
                                                    30
MPI_WIN_FLUSH_ALL(WIN, IERROR)                          31
    INTEGER WIN, IERROR                                32
                                                    33
MPI_WIN_FLUSH_LOCAL(RANK, WIN, IERROR)                  34
    INTEGER RANK, WIN, IERROR                          35
                                                    36
MPI_WIN_FLUSH_LOCAL_ALL(WIN, IERROR)                   37
    INTEGER WIN, IERROR                                38
                                                    39
MPI_WIN_FREE(WIN, IERROR)                                40
    INTEGER WIN, IERROR                                41
                                                    42
MPI_WIN_GET_GROUP(WIN, GROUP, IERROR)                   43
    INTEGER WIN, GROUP, IERROR                         44
                                                    45
MPI_WIN_GET_INFO(WIN, INFO_USED, IERROR)                46
    INTEGER WIN, INFO_USED, IERROR                    47
                                                    48
MPI_WIN_LOCK(LOCK_TYPE, RANK, ASSERT, WIN, IERROR)
    INTEGER LOCK_TYPE, RANK, ASSERT, WIN, IERROR
MPI_WIN_LOCK_ALL(ASSERT, WIN, IERROR)
    INTEGER ASSERT, WIN, IERROR
```

```

1  MPI_WIN_POST(GROUP, ASSERT, WIN, IERROR)
2      INTEGER GROUP, ASSERT, WIN, IERROR
3
4  MPI_WIN_SET_INFO(WIN, INFO, IERROR)
5      INTEGER WIN, INFO, IERROR
6
7  MPI_WIN_SHARED_QUERY(WIN, RANK, SIZE, DISP_UNIT, BASEPTR, IERROR)
8      INTEGER WIN, RANK, DISP_UNIT, IERROR
9      INTEGER (KIND=MPI_ADDRESS_KIND) SIZE, BASEPTR
10
11  If the Fortran compiler provides TYPE(C_PTR), then overloaded by:
12
13  INTERFACE MPI_WIN_SHARED_QUERY
14      SUBROUTINE MPI_WIN_SHARED_QUERY(WIN, RANK, SIZE, DISP_UNIT, &
15          BASEPTR, IERROR)
16          IMPORT :: MPI_ADDRESS_KIND
17          INTEGER :: WIN, RANK, DISP_UNIT, IERROR
18          INTEGER(KIND=MPI_ADDRESS_KIND) :: SIZE, BASEPTR
19      END SUBROUTINE
20      SUBROUTINE MPI_WIN_SHARED_QUERY_CPTR(WIN, RANK, SIZE, DISP_UNIT, &
21          BASEPTR, IERROR)
22          USE, INTRINSIC :: ISO_C_BINDING, ONLY : C_PTR
23          IMPORT :: MPI_ADDRESS_KIND
24          INTEGER :: WIN, RANK, DISP_UNIT, IERROR
25          INTEGER(KIND=MPI_ADDRESS_KIND) :: SIZE
26          TYPE(C_PTR) :: BASEPTR
27      END SUBROUTINE
28  END INTERFACE
29
30  MPI_WIN_START(GROUP, ASSERT, WIN, IERROR)
31      INTEGER GROUP, ASSERT, WIN, IERROR
32
33  MPI_WIN_SYNC(WIN, IERROR)
34      INTEGER WIN, IERROR
35
36  MPI_WIN_TEST(WIN, FLAG, IERROR)
37      INTEGER WIN, IERROR
38      LOGICAL FLAG
39
40  MPI_WIN_UNLOCK(RANK, WIN, IERROR)
41      INTEGER RANK, WIN, IERROR
42
43  MPI_WIN_UNLOCK_ALL(WIN, IERROR)
44      INTEGER WIN, IERROR
45
46  MPI_WIN_WAIT(WIN, IERROR)
47      INTEGER WIN, IERROR
48
49
50  A.4.10 External Interfaces Fortran Bindings
51
52  MPI_GREQUEST_COMPLETE(REQUEST, IERROR)
53      INTEGER REQUEST, IERROR

```

```

MPI_GREQUEST_START(QUERY_FN, FREE_FN, CANCEL_FN, EXTRA_STATE, REQUEST,
                  IERROR)
    EXTERNAL QUERY_FN, FREE_FN, CANCEL_FN
    INTEGER(KIND=MPI_ADDRESS_KIND) EXTRA_STATE
    INTEGER REQUEST, IERROR

MPI_INIT_THREAD(REQUIRED, PROVIDED, IERROR)
    INTEGER REQUIRED, PROVIDED, IERROR

MPI_IS_THREAD_MAIN(FLAG, IERROR)
    LOGICAL FLAG
    INTEGER IERROR

MPI_QUERY_THREAD(PROVIDED, IERROR)
    INTEGER PROVIDED, IERROR

MPI_STATUS_SET_CANCELLED(STATUS, FLAG, IERROR)
    INTEGER STATUS(MPI_STATUS_SIZE)
    LOGICAL FLAG
    INTEGER IERROR

MPI_STATUS_SET_ELEMENTS(STATUS, DATATYPE, COUNT, IERROR)
    INTEGER STATUS(MPI_STATUS_SIZE), DATATYPE, COUNT, IERROR

MPI_STATUS_SET_ELEMENTS_X(STATUS, DATATYPE, COUNT, IERROR)
    INTEGER STATUS(MPI_STATUS_SIZE), DATATYPE
    INTEGER(KIND=MPI_COUNT_KIND) COUNT
    INTEGER IERROR

A.4.11 I/O Fortran Bindings

MPI_CONVERSION_FN_NULL(USERBUF, DATATYPE, COUNT, FILEBUF, POSITION,
                      EXTRA_STATE, IERROR)
    <TYPE> USERBUF(*), FILEBUF(*)
    INTEGER COUNT, DATATYPE, IERROR
    INTEGER(KIND=MPI_OFFSET_KIND) POSITION
    INTEGER(KIND=MPI_ADDRESS_KIND) EXTRA_STATE

MPI_FILE_CLOSE(FH, IERROR)
    INTEGER FH, IERROR

MPI_FILE_DELETE(FILENAME, INFO, IERROR)
    CHARACTER*(*) FILENAME
    INTEGER INFO, IERROR

MPI_FILE_GET_AMODE(FH, AMODE, IERROR)
    INTEGER FH, AMODE, IERROR

MPI_FILE_GET_ATOMICITY(FH, FLAG, IERROR)
    INTEGER FH, IERROR
    LOGICAL FLAG

MPI_FILE_GET_BYTE_OFFSET(FH, OFFSET, DISP, IERROR)

```

```

1      INTEGER FH, IERROR
2      INTEGER(KIND=MPI_OFFSET_KIND) OFFSET, DISP
3
4      MPI_FILE_GET_GROUP(FH, GROUP, IERROR)
5      INTEGER FH, GROUP, IERROR
6
7      MPI_FILE_GET_INFO(FH, INFO_USED, IERROR)
8      INTEGER FH, INFO_USED, IERROR
9
10     MPI_FILE_GET_POSITION(FH, OFFSET, IERROR)
11     INTEGER FH, IERROR
12     INTEGER(KIND=MPI_OFFSET_KIND) OFFSET
13
14     MPI_FILE_GET_POSITION_SHARED(FH, OFFSET, IERROR)
15     INTEGER FH, IERROR
16     INTEGER(KIND=MPI_OFFSET_KIND) OFFSET
17
18     MPI_FILE_GET_SIZE(FH, SIZE, IERROR)
19     INTEGER FH, IERROR
20     INTEGER(KIND=MPI_OFFSET_KIND) SIZE
21
22     MPI_FILE_GET_TYPE_EXTENT(FH, DATATYPE, EXTENT, IERROR)
23     INTEGER FH, DATATYPE, IERROR
24     INTEGER(KIND=MPI_ADDRESS_KIND) EXTENT
25
26     MPI_FILE_GET_VIEW(FH, DISP, ETYPE, FILETYPE, DATAREP, IERROR)
27     INTEGER FH, ETYPE, FILETYPE, IERROR
28     CHARACTER*(*) DATAREP
29     INTEGER(KIND=MPI_OFFSET_KIND) DISP
30
31     MPI_FILE_IREAD(FH, BUF, COUNT, DATATYPE, REQUEST, IERROR)
32     <type> BUF(*)
33     INTEGER FH, COUNT, DATATYPE, REQUEST, IERROR
34
35     MPI_FILE_IREAD_ALL(FH, BUF, COUNT, DATATYPE, REQUEST, IERROR)
36     <type> BUF(*)
37     INTEGER FH, COUNT, DATATYPE, REQUEST, IERROR
38
39     MPI_FILE_IREAD_AT(FH, OFFSET, BUF, COUNT, DATATYPE, REQUEST, IERROR)
40     <type> BUF(*)
41     INTEGER FH, COUNT, DATATYPE, REQUEST, IERROR
42     INTEGER(KIND=MPI_OFFSET_KIND) OFFSET
43
44     MPI_FILE_IREAD_SHARED(FH, BUF, COUNT, DATATYPE, REQUEST, IERROR)
45     <type> BUF(*)
46     INTEGER FH, COUNT, DATATYPE, REQUEST, IERROR
47
48     MPI_FILE_IWRITE(FH, BUF, COUNT, DATATYPE, REQUEST, IERROR)
49     <type> BUF(*)

```

```

    INTEGER FH, COUNT, DATATYPE, REQUEST, IERROR                                1
MPI_FILE_IWRITE_ALL(FH, BUF, COUNT, DATATYPE, REQUEST, IERROR)                2
    <type> BUF(*)                                                                3
    INTEGER FH, COUNT, DATATYPE, REQUEST, IERROR                                4
MPI_FILE_IWRITE_AT(FH, OFFSET, BUF, COUNT, DATATYPE, REQUEST, IERROR)        6
    <type> BUF(*)                                                                7
    INTEGER FH, COUNT, DATATYPE, REQUEST, IERROR                                8
    INTEGER(KIND=MPI_OFFSET_KIND) OFFSET                                        9
MPI_FILE_IWRITE_AT_ALL(FH, OFFSET, BUF, COUNT, DATATYPE, REQUEST, IERROR)    11
    <type> BUF(*)                                                                12
    INTEGER FH, COUNT, DATATYPE, REQUEST, IERROR                                13
    INTEGER(KIND=MPI_OFFSET_KIND) OFFSET                                        14
MPI_FILE_IWRITE_SHARED(FH, BUF, COUNT, DATATYPE, REQUEST, IERROR)            15
    <type> BUF(*)                                                                16
    INTEGER FH, COUNT, DATATYPE, REQUEST, IERROR                                17
MPI_FILE_OPEN(COMM, FILENAME, AMODE, INFO, FH, IERROR)                        19
    CHARACTER*(*) FILENAME                                                       20
    INTEGER COMM, AMODE, INFO, FH, IERROR                                        21
MPI_FILE_PREALLOCATE(FH, SIZE, IERROR)                                        22
    INTEGER FH, IERROR                                                            23
    INTEGER(KIND=MPI_OFFSET_KIND) SIZE                                           24
MPI_FILE_READ(FH, BUF, COUNT, DATATYPE, STATUS, IERROR)                       26
    <type> BUF(*)                                                                27
    INTEGER FH, COUNT, DATATYPE, STATUS(MPI_STATUS_SIZE), IERROR                28
MPI_FILE_READ_ALL(FH, BUF, COUNT, DATATYPE, STATUS, IERROR)                  29
    <type> BUF(*)                                                                30
    INTEGER FH, COUNT, DATATYPE, STATUS(MPI_STATUS_SIZE), IERROR                31
MPI_FILE_READ_ALL_BEGIN(FH, BUF, COUNT, DATATYPE, IERROR)                    33
    <type> BUF(*)                                                                34
    INTEGER FH, COUNT, DATATYPE, IERROR                                          35
MPI_FILE_READ_ALL_END(FH, BUF, STATUS, IERROR)                                36
    <type> BUF(*)                                                                37
    INTEGER FH, STATUS(MPI_STATUS_SIZE), IERROR                                  38
MPI_FILE_READ_AT(FH, OFFSET, BUF, COUNT, DATATYPE, STATUS, IERROR)           40
    <type> BUF(*)                                                                41
    INTEGER FH, COUNT, DATATYPE, STATUS(MPI_STATUS_SIZE), IERROR                42
    INTEGER(KIND=MPI_OFFSET_KIND) OFFSET                                        43
MPI_FILE_READ_AT_ALL(FH, OFFSET, BUF, COUNT, DATATYPE, STATUS, IERROR)       44
    <type> BUF(*)                                                                45
    INTEGER FH, COUNT, DATATYPE, STATUS(MPI_STATUS_SIZE), IERROR                46
    INTEGER(KIND=MPI_OFFSET_KIND) OFFSET                                        47

```

```
1 MPI_FILE_READ_AT_ALL_BEGIN(FH, OFFSET, BUF, COUNT, DATATYPE, IERROR)
2   <type> BUF(*)
3   INTEGER FH, COUNT, DATATYPE, IERROR
4   INTEGER(KIND=MPI_OFFSET_KIND) OFFSET
5
6 MPI_FILE_READ_AT_ALL_END(FH, BUF, STATUS, IERROR)
7   <type> BUF(*)
8   INTEGER FH, STATUS(MPI_STATUS_SIZE), IERROR
9
10 MPI_FILE_READ_ORDERED(FH, BUF, COUNT, DATATYPE, STATUS, IERROR)
11   <type> BUF(*)
12   INTEGER FH, COUNT, DATATYPE, STATUS(MPI_STATUS_SIZE), IERROR
13
14 MPI_FILE_READ_ORDERED_BEGIN(FH, BUF, COUNT, DATATYPE, IERROR)
15   <type> BUF(*)
16   INTEGER FH, COUNT, DATATYPE, IERROR
17
18 MPI_FILE_READ_ORDERED_END(FH, BUF, STATUS, IERROR)
19   <type> BUF(*)
20   INTEGER FH, STATUS(MPI_STATUS_SIZE), IERROR
21
22 MPI_FILE_READ_SHARED(FH, BUF, COUNT, DATATYPE, STATUS, IERROR)
23   <type> BUF(*)
24   INTEGER FH, COUNT, DATATYPE, STATUS(MPI_STATUS_SIZE), IERROR
25
26 MPI_FILE_SEEK(FH, OFFSET, WHENCE, IERROR)
27   INTEGER FH, WHENCE, IERROR
28   INTEGER(KIND=MPI_OFFSET_KIND) OFFSET
29
30 MPI_FILE_SEEK_SHARED(FH, OFFSET, WHENCE, IERROR)
31   INTEGER FH, WHENCE, IERROR
32   INTEGER(KIND=MPI_OFFSET_KIND) OFFSET
33
34 MPI_FILE_SET_ATOMICITY(FH, FLAG, IERROR)
35   INTEGER FH, IERROR
36   LOGICAL FLAG
37
38 MPI_FILE_SET_INFO(FH, INFO, IERROR)
39   INTEGER FH, INFO, IERROR
40
41 MPI_FILE_SET_SIZE(FH, SIZE, IERROR)
42   INTEGER FH, IERROR
43   INTEGER(KIND=MPI_OFFSET_KIND) SIZE
44
45 MPI_FILE_SET_VIEW(FH, DISP, ETYPE, FILETYPE, DATAREP, INFO, IERROR)
46   INTEGER FH, ETYPE, FILETYPE, INFO, IERROR
47   CHARACTER*(*) DATAREP
48   INTEGER(KIND=MPI_OFFSET_KIND) DISP
49
50 MPI_FILE_SYNC(FH, IERROR)
51   INTEGER FH, IERROR
52
53 MPI_FILE_WRITE(FH, BUF, COUNT, DATATYPE, STATUS, IERROR)
54   <type> BUF(*)
```



```

    INTEGER FH, COUNT, DATATYPE, STATUS(MPI_STATUS_SIZE), IERROR          1
MPI_FILE_WRITE_ALL(FH, BUF, COUNT, DATATYPE, STATUS, IERROR)           2
    <type> BUF(*)                                                         3
    INTEGER FH, COUNT, DATATYPE, STATUS(MPI_STATUS_SIZE), IERROR          4
MPI_FILE_WRITE_ALL_BEGIN(FH, BUF, COUNT, DATATYPE, IERROR)             6
    <type> BUF(*)                                                         7
    INTEGER FH, COUNT, DATATYPE, IERROR                                    8
MPI_FILE_WRITE_ALL_END(FH, BUF, STATUS, IERROR)                         10
    <type> BUF(*)                                                         11
    INTEGER FH, STATUS(MPI_STATUS_SIZE), IERROR                          12
MPI_FILE_WRITE_AT(FH, OFFSET, BUF, COUNT, DATATYPE, STATUS, IERROR)    13
    <type> BUF(*)                                                         14
    INTEGER FH, COUNT, DATATYPE, STATUS(MPI_STATUS_SIZE), IERROR        15
    INTEGER(KIND=MPI_OFFSET_KIND) OFFSET                                 16
MPI_FILE_WRITE_AT_ALL(FH, OFFSET, BUF, COUNT, DATATYPE, STATUS, IERROR) 18
    <type> BUF(*)                                                         19
    INTEGER FH, COUNT, DATATYPE, STATUS(MPI_STATUS_SIZE), IERROR        20
    INTEGER(KIND=MPI_OFFSET_KIND) OFFSET                                 21
MPI_FILE_WRITE_AT_ALL_BEGIN(FH, OFFSET, BUF, COUNT, DATATYPE, IERROR)  22
    <type> BUF(*)                                                         23
    INTEGER FH, COUNT, DATATYPE, IERROR                                    24
    INTEGER(KIND=MPI_OFFSET_KIND) OFFSET                                 25
MPI_FILE_WRITE_AT_ALL_END(FH, BUF, STATUS, IERROR)                     27
    <type> BUF(*)                                                         28
    INTEGER FH, STATUS(MPI_STATUS_SIZE), IERROR                          29
MPI_FILE_WRITE_ORDERED(FH, BUF, COUNT, DATATYPE, STATUS, IERROR)       30
    <type> BUF(*)                                                         31
    INTEGER FH, COUNT, DATATYPE, STATUS(MPI_STATUS_SIZE), IERROR        32
MPI_FILE_WRITE_ORDERED_BEGIN(FH, BUF, COUNT, DATATYPE, IERROR)         34
    <type> BUF(*)                                                         35
    INTEGER FH, COUNT, DATATYPE, IERROR                                    36
MPI_FILE_WRITE_ORDERED_END(FH, BUF, STATUS, IERROR)                     37
    <type> BUF(*)                                                         38
    INTEGER FH, STATUS(MPI_STATUS_SIZE), IERROR                          39
MPI_FILE_WRITE_SHARED(FH, BUF, COUNT, DATATYPE, STATUS, IERROR)        41
    <type> BUF(*)                                                         42
    INTEGER FH, COUNT, DATATYPE, STATUS(MPI_STATUS_SIZE), IERROR        43
MPI_REGISTER_DATAREP(DATAREP, READ_CONVERSION_FN, WRITE_CONVERSION_FN,  44
    DTYPE_FILE_EXTENT_FN, EXTRA_STATE, IERROR)                          45
    CHARACTER*(*) DATAREP                                               46
    EXTERNAL READ_CONVERSION_FN, WRITE_CONVERSION_FN, DTYPE_FILE_EXTENT_FN 47

```

```

1     INTEGER(KIND=MPI_ADDRESS_KIND) EXTRA_STATE
2     INTEGER IERROR
3
4

```

A.4.12 Language Bindings Fortran Bindings

```

6     MPI_F_SYNC_REG(buf)
7         <type> buf(*)
8
9     MPI_SIZEOF(X, SIZE, IERROR)
10        <type> X
11        INTEGER SIZE, IERROR
12
13    MPI_STATUS_F082F(F08_STATUS, F_STATUS, IERROR)
14        TYPE(MPI_Status) :: F08_STATUS
15        INTEGER :: F_STATUS(MPI_STATUS_SIZE)
16        INTEGER IERROR
17
18    MPI_STATUS_F2F08(F_STATUS, F08_STATUS, IERROR)
19        INTEGER :: F_STATUS(MPI_STATUS_SIZE)
20        TYPE(MPI_Status) :: F08_STATUS
21        INTEGER IERROR
22
23    MPI_TYPE_CREATE_F90_COMPLEX(P, R, NEWTYPE, IERROR)
24        INTEGER P, R, NEWTYPE, IERROR
25
26    MPI_TYPE_CREATE_F90_INTEGER(R, NEWTYPE, IERROR)
27        INTEGER R, NEWTYPE, IERROR
28
29    MPI_TYPE_CREATE_F90_REAL(P, R, NEWTYPE, IERROR)
30        INTEGER P, R, NEWTYPE, IERROR
31
32
33
34
35
36
37
38
39    MPI_TYPE_MATCH_SIZE(TYPECLASS, SIZE, DATATYPE, IERROR)
40        INTEGER TYPECLASS, SIZE, DATATYPE, IERROR
41
42
43
44
45
46
47
48

```

A.4.13 Tools / Profiling Interface Fortran Bindings

```

34    MPI_PCONTROL(LEVEL)
35        INTEGER LEVEL
36
37

```

A.4.14 Deprecated Fortran Bindings

```

39    MPI_ATTR_DELETE(COMM, KEYVAL, IERROR)
40        INTEGER COMM, KEYVAL, IERROR
41
42    MPI_ATTR_GET(COMM, KEYVAL, ATTRIBUTE_VAL, FLAG, IERROR)
43        INTEGER COMM, KEYVAL, ATTRIBUTE_VAL, IERROR
44        LOGICAL FLAG
45
46    MPI_ATTR_PUT(COMM, KEYVAL, ATTRIBUTE_VAL, IERROR)
47        INTEGER COMM, KEYVAL, ATTRIBUTE_VAL, IERROR
48

```

```
MPI_DUP_FN(OLDCOMM, KEYVAL, EXTRA_STATE, ATTRIBUTE_VAL_IN,      1
           ATTRIBUTE_VAL_OUT, FLAG, IERR)                        2
    INTEGER OLDCOMM, KEYVAL, EXTRA_STATE, ATTRIBUTE_VAL_IN,    3
    ATTRIBUTE_VAL_OUT, IERR                                     4
    LOGICAL FLAG                                              5
                                                                6
MPI_KEYVAL_CREATE(COPY_FN, DELETE_FN, KEYVAL, EXTRA_STATE, IERROR) 7
    EXTERNAL COPY_FN, DELETE_FN                                8
    INTEGER KEYVAL, EXTRA_STATE, IERROR                       9
                                                                10
MPI_KEYVAL_FREE(KEYVAL, IERROR)                                11
    INTEGER KEYVAL, IERROR                                    12
                                                                13
MPI_NULL_COPY_FN(OLDCOMM, KEYVAL, EXTRA_STATE, ATTRIBUTE_VAL_IN, 14
                ATTRIBUTE_VAL_OUT, FLAG, IERR)                15
    INTEGER OLDCOMM, KEYVAL, EXTRA_STATE, ATTRIBUTE_VAL_IN, 16
    ATTRIBUTE_VAL_OUT, IERR                                   17
    LOGICAL FLAG                                             18
                                                                19
MPI_NULL_DELETE_FN(COMM, KEYVAL, ATTRIBUTE_VAL, EXTRA_STATE, IERROR) 20
    INTEGER COMM, KEYVAL, ATTRIBUTE_VAL, EXTRA_STATE, IERROR 21
                                                                22
SUBROUTINE COPY_FUNCTION(OLDCOMM, KEYVAL, EXTRA_STATE, ATTRIBUTE_VAL_IN, 23
                        ATTRIBUTE_VAL_OUT, FLAG, IERR)         24
    INTEGER OLDCOMM, KEYVAL, EXTRA_STATE, ATTRIBUTE_VAL_IN, 25
    ATTRIBUTE_VAL_OUT, IERR                                   26
    LOGICAL FLAG                                             27
                                                                28
SUBROUTINE DELETE_FUNCTION(COMM, KEYVAL, ATTRIBUTE_VAL, EXTRA_STATE, IERR) 29
    INTEGER COMM, KEYVAL, ATTRIBUTE_VAL, EXTRA_STATE, IERR 30
                                                                31
                                                                32
                                                                33
                                                                34
                                                                35
                                                                36
                                                                37
                                                                38
                                                                39
                                                                40
                                                                41
                                                                42
                                                                43
                                                                44
                                                                45
                                                                46
                                                                47
                                                                48
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48