Library Routines

Initialization Routines

**START_PES (NPES)**
Initializes the OpenSHMEM library. This routine must be called before any library other routine is called.

Query Routines

**MY_PE ()**
Returns the virtual PE number of the calling PE.

**NUM_PES ()**
Returns the virtual PE number of the calling PE.

Data Transfer Routines

**SHMEM_[funcname]_GET(target, source, len, pe)**
Retrieve contiguous data from a remote PE.  
[funcname] can be anything in { INTEGER, DOUBLE, COMPLEX, LOGICAL, REAL, CHARACTER }

**SHMEM_GET[funcname](target, source, len, pe)**
Retrieve contiguous data from a remote PE.  
[funcname] can be anything in { 32, 64, 128, MEM }

**SHMEM_[funcname]_IGET(target, source, tst, sst, len, pe)**
Retrieve strided (target, source stride can be different) data from a remote PE.  
[funcname] can be anything in { INTEGER, DOUBLE, COMPLEX, LOGICAL, REAL }

**SHMEM_IGET[funcname](target, source, tst, sst, len, pe)**
Retrieve strided (target, source stride can be different) data from a remote PE.  
[funcname] can be anything in { 32, 64, 128, MEM }

**SHMEM_[funcname]_PUT(target, source, len, pe)**
Write contiguous data to a remote PE.  
[funcname] can be anything in { INTEGER, DOUBLE, COMPLEX, LOGICAL, REAL, CHARACTER }
Library Routines (Continued)

Data Transfer Routines (Continued)

SHMEM_PUT[funcname](target, source, len, pe)
Write contiguous data to a remote PE.
[funcname] can be any of { 32, 64, 128, MEM }

SHMEM_[funcname]_IPUT(target, source, tst, sst, len, pe)
Write strided (target, source stride can be different) data to a remote PE.
[funcname] can be anything in { INTEGER, DOUBLE, COMPLEX, LOGICAL, REAL }

SHMEM_IPUT_[funcname](target, source, tst, sst, len, pe)
Write strided (target, source stride can be different) data to a remote PE.
[funcname] can be anything in { 32, 64, 128, MEM }

Synchronization Routines

SHMEM_BARRIER_ALL()
Suspend execution on the calling PE, until all other PEs reach this point of execution path.

SHMEM_BARRIER(PE_start, logPE_stride, PE_size, pSync)
Suspend execution on the calling PE, until a subset of PEs, defined by PE_start, logPE_stride and PE_size, reaches this point of execution path.

SHMEM_FENCE()
Ensure ordering or remote put operations to a particular PE.

SHMEM_QUIET()
Ensure ordering or remote put operations to multiple PEs.

Symmetric Heap Routines

SHPALLOC(addr, length, errcode, abort)
Allocates a memory block in the symmetric heap.

SHPCLMOVE(addr, length, status, abort)
Adjust the size of a symmetric memory block.

SHDEPALLC(addr, errcode, abort)
Deallocates a symmetric memory block.

Remote Pointer Routines

SHMEM_PTR(target, pe)
Returns a pointer to a data object of a remote PE.
Library Routines (Continued)

Collect Routines

**SHMEM_FCOLLECT[bits](target, source, nlong, PE_start, logPE_stride, PE_size, pSync)**
Concatenate remote data objects and stores the result in a local data object. nlong must be the same on all PEs.
**[bits]** can be any of \{ 4, 8 \}

**SHMEM_COLLECT[bits](target, source, nlong, PE_start, logPE_stride, PE_size, pSync)**
Concatenate remote data objects and stores the result in a local data object. nlong can vary from PE to PE.
**[bits]** can be any of \{ 4, 8 \}

Broadcast Routines

**SHMEM_BROADCAST[bits](target, source, nlong, PE_start, logPE_stride, PE_size, pSync)**
Write data to a symmetric data object on all PEs of the active set.
**[bits]** can be any of \{ 4, 8 \}

Reduction Routines

**SHMEM_[funcname]_[opname]_TO_ALL(target, source, nlong, PE_start, logPE_stride, PE_size, pWrk, pSync)**
Perform a logical reduction operation on symmetric data objects of all PEs in the active set.
**[funcname]** can be any of \{ INT4, INT8 \}
**[opname]** can be any of \{ AND, OR, XOR \}

**SHMEM_[funcname]_[opname]_TO_ALL(target, source, nlong, PE_start, logPE_stride, PE_size, pWrk, pSync)**
Perform a reduction operation on symmetric data objects of all PEs in the active set.
**[funcname]** can be any of \{ INT4, INT8, REAL4, REAL8 \}
**[opname]** can be any of \{ SUM, PROD, MIN, MAX \}
Environment Variables

SGI Specific Environment Variables

**SMA_VERSION**
Print library version at library startup.

**SMA_INFO**
Print helpful text about all these environment variables.

**SMA_SYMMETRIC_SIZE**
Number of bytes to allocate for the symmetric heap.

**SMA_DEBUG**
Enable debugging messages.

Reference Implementation Specific Environment Variables

**SHMEM_LOG_LEVELS**
A comma, space, semi-colon separated list of logging/trace facilities to enable debugging messages. The facilities currently supported include the following case-sensitive names:

- FATAL, DEBUG, INFO, NOTICE, AUTH, INIT, MEMORY, CACHE, BARRIER, BROADCAST, COLLECT, REDUCE, SYMBOLS, LOCK, SERVICE, FENCE, QUIET

Please refer to the OpenSHMEM Reference Implementation design document for more information about the facilities mentioned above.

**SHMEM_LOG_FILE**
A filename to which to write log messages.

**SHMEM_SYMMETRIC_HEAP_SIZE**
The number of bytes to allocate for the symmetric heap area. Can scale units with “K”, “M” etc. modifiers. The default is 1M.

**SHMEM_BARRIER_ALGORITHM**
The version of the barrier to use. The default is “naive”. Designed to allow people to plug other variants in easily and test.

**SHMEM_BARRIER_ALGORITHM_ALL**
As for SHMEM_BARRIER_ALGORITHM, but separating these two allows us to optimize if e.g. hardware has special support for global barriers.

**SHMEM_PE_ACCESSIBLE_TIMEOUT**
The number of seconds to wait for PEs to reply to accessibility checks. The default is 1.0 (i.e. may be fractional).