Mellanox Scalable Hierarchical Aggregation and Reduction Protocol (SHARP)

API Guide

Rev 1.1

NOTE:
THIS HARDWARE, SOFTWARE OR TEST SUITE PRODUCT ("PRODUCT(S)") AND ITS RELATED DOCUMENTATION ARE PROVIDED BY MELLANOX TECHNOLOGIES "AS-IS" WITH ALL FAULTS OF ANY KIND AND SOLELY FOR THE PURPOSE OF AIDING THE CUSTOMER IN TESTING APPLICATIONS THAT USE THE PRODUCTS IN DESIGNATED SOLUTIONS. THE CUSTOMER'S MANUFACTURING TEST ENVIRONMENT HAS NOT MET THE STANDARDS SET BY MELLANOX TECHNOLOGIES TO FULLY QUALIFY THE PRODUCT(S) AND/OR THE SYSTEM USING IT. THEREFORE, MELLANOX TECHNOLOGIES CANNOT AND DOES NOT GUARANTEE OR WARRANT THAT THE PRODUCTS WILL OPERATE WITH THE HIGHEST QUALITY. ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT ARE DISCLAIMED. IN NO EVENT SHALL MELLANOX BE LIABLE TO CUSTOMER OR ANY THIRD PARTIES FOR ANY DIRECT, INDIRECT, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES OF ANY KIND (INCLUDING, BUT NOT LIMITED TO, PAYMENT FOR PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY FROM THE USE OF THE PRODUCT(S) AND RELATED DOCUMENTATION EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Mellanox Technologies
350 Oakmead Parkway Suite 100
Sunnyvale, CA 94085
U.S.A.
www.mellanox.com
Tel: (408) 970-3400
Fax: (408) 970-3403

Table of Contents

Document Revision History

Table 1: Document Revision History

Revision	Date	Description
1.1	Mar. 4, 2018	Updated Mellanox Scalable Hierarchical Aggregation and Reduction Protocol (SHARP)™ trademark. Updated template.
1.0		Initial release.

Copyright

1 Introduction

Mellanox Scalable Hierarchical Aggregation and Reduction Protocol (SHARP)™ technology improves the performance of MPI operation, by offloading collective operations from the CPU to the switch network, and eliminating the need to send data multiple times between endpoints. This innovative approach decreases the amount of data traversing the network as aggregation nodes are reached, and dramatically reduces the MPI operations time. Implementing collective communication algorithms in the network also has additional benefits, such as freeing up valuable CPU resources for computation rather than using them to process communication.

2 Class Index

2.1 SHARP Accelerators

Here are the classes, structs, unions and interfaces with brief descriptions:

3 File Index

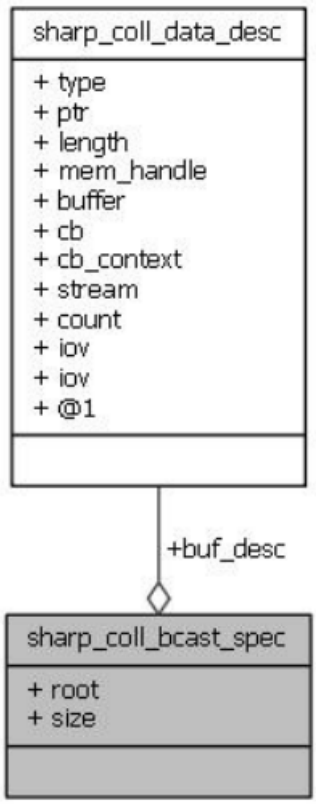
3.1 File List

Here is a list of all files with brief descriptions:

4 Class Documentation

4.1 sharp_coll_bcast_spec Struct Reference

Collaboration diagram for sharp_coll_bcast_spec:



4.1.1 Public Attributes

- int root
- struct sharp_coll_data_desc buf_desc
- int size

4.1.2 Member Data Documentation

4.1.2.1 int sharp_coll_bcast_spec::root

[in] root rank number

4.1.2.2 struct sharp_coll_data_desc sharp_coll_bcast_spec::buf_desc

[in,out] buffer desc to send/recv bcast data

4.1.2.3 int sharp_coll_bcast_spec::size

[in] bcast size

4.1.2.4 The documentation for this struct was generated from the following file:

- sharp_coll.h

4.2 sharp_coll_comm_init_spec Struct Reference

Collaboration diagram for sharp_coll_comm_init_spec:

4.2.1 Public Attributes

- int rank
- int size
- int is_comm_world
- void * oob_ctx

4.2.2 Member Data Documentation

4.2.2.1 int sharp_coll_comm_init_spec::rank

Uniq process rank in the group.

4.2.2.2 int sharp_coll_comm_init_spec::size

Size of the SHARP group.

4.2.2.3 int sharp_coll_comm_init_spec::is_comm_world

Is universal group (MPI_COMM_WORLD).

4.2.2.4 void* sharp_coll_comm_init_spec::oob_ctx

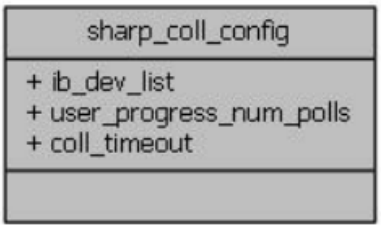
External group context for OOB functions.

4.2.2.5 The documentation for this struct was generated from the following file:

- sharp_coll.h

4.3 sharp_coll_config Struct Reference

Collaboration diagram for sharp_coll_config:



4.3.1 Public Attributes

- char * ib_dev_list
- int user_progress_num_polls
- int coll_timeout

4.3.2 Member Data Documentation

4.3.2.1 char* sharp_coll_config::ib_dev_list

IB device name, port list.

4.3.2.2 int sharp_coll_config::user_progress_num_polls

Number of polls to do before calling user progress.

4.3.2.3 int sharp_coll_config::coll_timeout

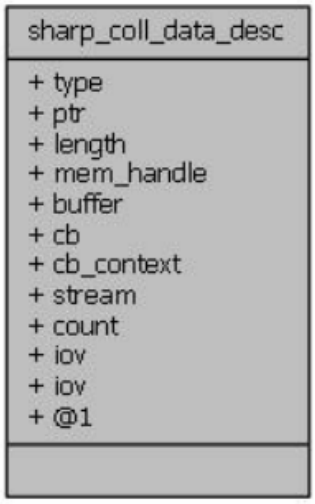
Timeout (msec) for collective operation, -1 - infinite

4.3.2.4 The documentation for this struct was generated from the following file:

- sharp_coll.h

4.4 sharp_coll_data_desc Struct Reference

Collaboration diagram for sharp_coll_data_desc:



4.4.1 Public Attributes

- enum sharp_data_buffer_type type
- union {
  - struct {
    - void * ptr
    - size_t length
    - void * mem_handle
  - } buffer
  - struct {
    - sharp_stream_cb cb
    - void * cb_context
    - size_t length
  - } stream
  - struct {
    - unsigned count
    - struct iovec * iov
  - } iov
- };

4.4.2 Member Data Documentation

4.4.2.1 enum sharp_data_buffer_type sharp_coll_data_desc::type

4.4.2.2 void* sharp_coll_data_desc::ptr

Contiguous data buffer.

4.4.2.3 size_t sharp_coll_data_desc::length

Buffer len.

Total data length.

4.4.2.4 void* sharp_coll_data_desc::mem_handle

Memory handle returned from sharp_coll_reg_mr

4.4.2.5 struct { ... } sharp_coll_data_desc::buffer

4.4.2.6 sharp_stream_cb sharp_coll_data_desc::cb

Streaming callback.

4.4.2.7 void* sharp_coll_data_desc::cb_context

Context for stream callback.

4.4.2.8 struct { ... } sharp_coll_data_desc::stream

4.4.2.9 unsigned sharp_coll_data_desc::count

Number of IOV entries.

4.4.2.10 struct iovec* sharp_coll_data_desc::iov

IOV entries.

4.4.2.11 struct { ... } sharp_coll_data_desc::iov

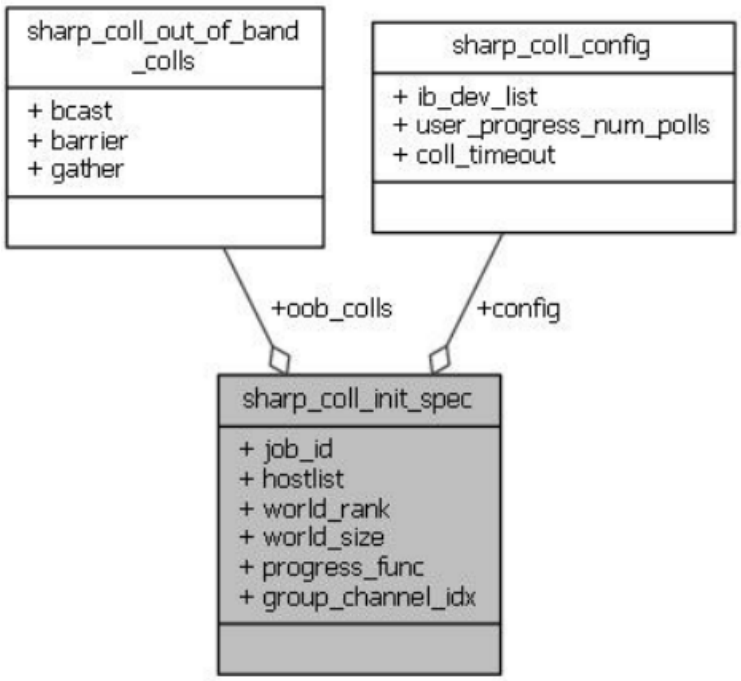
4.4.2.12 union { ... }

4.4.2.13 The documentation for this struct was generated from the following file:

- sharp_coll.h

4.5 sharp_coll_init_spec Struct Reference

Collaboration diagram for sharp_coll_init_spec:



4.5.1 Public Attributes

- uint64_t job_id
- char * hostlist
- int world_rank
- int world_size
- int(* progress_func)(void)
- int group_channel_idx
- struct sharp_coll_config config
- struct sharp_coll_out_of_band_colls oob_colls

4.5.2 Member Data Documentation

4.5.2.1 uint64_t sharp_coll_init_spec::job_id

Job unique ID

4.5.2.2 char* sharp_coll_init_spec::hostlist

Comma separated hostlist.

4.5.2.3 int sharp_coll_init_spec::world_rank

Global unique process number.

4.5.2.4 int sharp_coll_init_spec::world_size

Num of processes in the job.

4.5.2.5 int(* sharp_coll_init_spec::progress_func) (void)

External progress function.

4.5.2.6 int sharp_coll_init_spec::group_channel_idx

local group channel index(0 .. (max - 1))

4.5.2.7 struct sharp_coll_config sharp_coll_init_spec::config

SHARP COLL Configuration.

4.5.2.8 struct sharp_coll_out_of_band_colls sharp_coll_init_spec::oob_colls

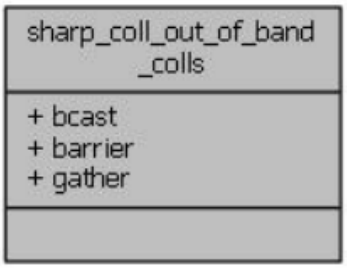
List of OOB collectives.

4.5.2.9 The documentation for this struct was generated from the following file:

- sharp_coll.h

4.6 sharp_coll_out_of_band_colls Struct Reference

Collaboration diagram for sharp_coll_out_of_band_colls:



4.6.1 Public Attributes

- int(* bcast)(void *context, void *buffer, int len, int root)
  out-of-band broadcast
- int(* barrier)(void *context)
  out-of-band barrier
- int(* gather)(void *context, int root, void *sbuf, void *rbuf, int len)
  out-of-band gather

4.6.2 Member Data Documentation

4.6.2.1 int(* sharp_coll_out_of_band_colls::bcast) (void *context, void *buffer, int len, int root)

Out-of-band broadcast

The pointer refers to application defined out-of-band bcast collective.

4.6.2.1.1 Parameters:

in	context	User-defined context or NULL.
in	buffer	Buffer to send/recv.
in	len	Size of the buffer.
in	root	Root of the the broadcast.

4.6.2.2 int(* sharp_coll_out_of_band_colls::barrier) (void *context)

Out-of-band barrier

The pointer refers to application defined out-of-band barrier collective.

4.6.2.2.1 Parameters:

in	context	User-defined context or NULL.

4.6.2.3 int(* sharp_coll_out_of_band_colls::gather) (void *context, int root, void *sbuf, void *rbuf, int len)

Out-of-band gather

The pointer refers to an application defined out-of-band gather collective.

4.6.2.3.1 Parameters:

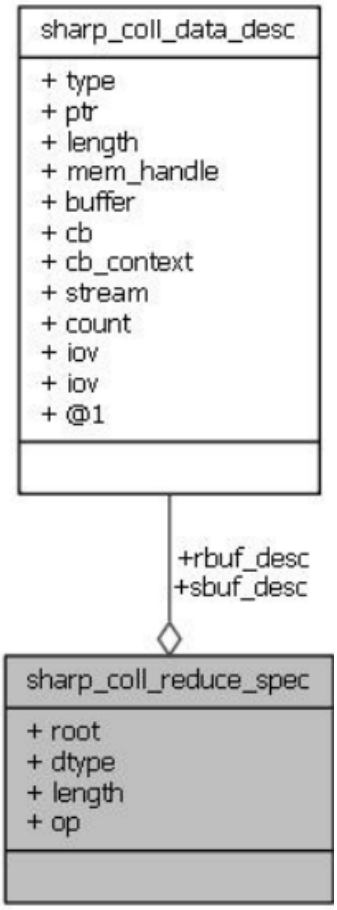
in	context	User-defined context or NULL.
in	root	Root of the broadcast.
in	sbuf	Buffer to send.
in	rbuf	Buffer to recvcle.
in	len	Size of the buffer.

4.6.2.4 The documentation for this struct was generated from the following file:

- sharp_coll.h

4.7 sharp_coll_reduce_spec Struct Reference

Collaboration diagram for sharp_coll_reduce_spec:



4.7.1 Public Attributes

- int root
- struct sharp_coll_data_desc sbuf_desc
- struct sharp_coll_data_desc rbuf_desc
- enum sharp_datatype dtype
- int length
- enum sharp_reduce_op op

4.7.2 Member Data Documentation

4.7.2.1 int sharp_coll_reduce_spec::root

[in] root rank number (ignored for allreduce)

4.7.2.2 struct sharp_coll_data_desc sharp_coll_reduce_spec::sbuf_desc

[in] source data buffer desc

4.7.2.3 struct sharp_coll_data_desc sharp_coll_reduce_spec::rbuf_desc

[out] destination data buffer desc

4.7.2.4 enum sharp_datatype sharp_coll_reduce_spec::dtype

[in] data type sharp_datatype

4.7.2.5 int sharp_coll_reduce_spec::length

[in] reduce operation size

4.7.2.6 enum sharp_reduce_op sharp_coll_reduce_spec::op

[in] reduce operator sharp_reduce_op

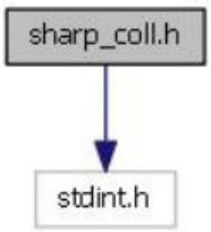
4.7.2.7 The documentation for this struct was generated from the following file:

- sharp_coll.h

5 File Documentation

5.1 sharp_coll.h File Reference

Include dependency graph for sharp_coll.h:



5.1.1 Classes

- struct sharp_coll_config
- struct sharp_coll_out_of_band_colls
- struct sharp_coll_comm_init_spec
- struct sharp_coll_init_spec
- struct sharp_coll_data_desc
- struct sharp_coll_reduce_spec
- struct sharp_coll_bcast_spec

5.1.2 Typedefs

- typedef size_t(* sharp_stream_cb) (void *buffer, size_t length, size_t offset, void *context)

  stream callback

5.1.3 Enumerations

- enum sharp_datatype { SHARP_DTYPE_UNSIGNED, SHARP_DTYPE_INT, SHARP_DTYPE_UNSIGNED_LONG, SHARP_DTYPE_LONG, SHARP_DTYPE_FLOAT, SHARP_DTYPE_DOUBLE, SHARP_DTYPE_NULL }SHARP coll supported data types.
- enum sharp_reduce_op { SHARP_OP_MAX, SHARP_OP_MIN, SHARP_OP_SUM, SHARP_OP_PROD, SHARP_OP_LAND, SHARP_OP_BAND, SHARP_OP_LOR, SHARP_OP_BOR, SHARP_OP_LXOR, SHARP_OP_BXOR, SHARP_OP_MAXLOC, SHARP_OP_MINLOC, SHARP_OP_NULL }SHARP coll supported aggregation operations.
- enum sharp_error_no { SHARP_COLL_SUCCESS = 0, SHARP_COLL_ERROR = -1, SHARP_COLL_ENOT_SUPP = -2, SHARP_COLL_ENOMEM = -3, SHARP_COLL_EGROUP_ALLOC = -4, SHARP_COLL_ECONN_TREE = -5, SHARP_COLL_EGROUP_JOIN = -6, SHARP_COLL_EQUOTA = -7, SHARP_COLL_ESESS_INIT = -8, SHARP_COLL_EDEV = -9, SHARP_COLL_EINVAL = -10, SHARP_COLL_EJOB_CREATE = -11, SHARP_COLL_ETREE_INFO = -12, SHARP_COLL_ENOTREE = -13,

SHARP_COLL_EGROUP_ID = -14, SHARP_COLL_EOOB = -15,
SHARP_COLL_EGROUP_MCAST = -16 }SHARP coll status code.

- enum sharp_data_buffer_type { SHARP_DATA_BUFFER, SHARP_DATA_STREAM, SHARP_DATA_IOV }SHARP coll buffer types.
- enum config_print_flags { SHARP_COLL_CONFIG_PRINT_CONFIG = 1,
SHARP_COLL_CONFIG_PRINT_HEADER = 2,
SHARP_COLL_CONFIG_PRINT_DOC = 4,
SHARP_COLL_CONFIG_PRINT_HIDDEN = 8 }

5.1.4 Functions

- int sharp_coll_init (struct sharp_coll_init_spec *sharp_coll_spec, struct sharp_coll_context **sharp_coll_context)
- int sharp_coll_finalize (struct sharp_coll_context *context)
- int sharp_coll_progress (struct sharp_coll_context *context)
- int sharp_coll_comm_init (struct sharp_coll_context *context, struct sharp_coll_comm_init_spec *spec, struct sharp_coll_comm **sharp_coll_comm)
- int sharp_coll_comm_destroy (struct sharp_coll_comm *comm)
- int sharp_coll_do_barrier (struct sharp_coll_comm *comm)
- int sharp_coll_do_barrier_nb (struct sharp_coll_comm *comm, struct sharp_coll_request **handle)
- int sharp_coll_do_allreduce (struct sharp_coll_comm *comm, struct sharp_coll_reduce_spec *spec)
- int sharp_coll_do_allreduce_nb (struct sharp_coll_comm *comm, struct sharp_coll_reduce_spec *spec, struct sharp_coll_request **req)
- int sharp_coll_do_reduce (struct sharp_coll_comm *comm, struct sharp_coll_reduce_spec *spec)
- int sharp_coll_do_reduce_nb (struct sharp_coll_comm *comm, struct sharp_coll_reduce_spec *spec, struct sharp_coll_request **handle)
- int sharp_coll_do_bcast (struct sharp_coll_comm *comm, struct sharp_coll_bcast_spec *spec)
- int sharp_coll_do_bcast_nb (struct sharp_coll_comm *comm, struct sharp_coll_bcast_spec *spec, struct sharp_coll_request **handle)
- int sharp_coll_req_test (struct sharp_coll_request *req)
- int sharp_coll_req_wait (struct sharp_coll_request *req)
- int sharp_coll_req_free (struct sharp_coll_request *req)
- int sharp_translate_mpi_op (const char *mpi_op_str)
- int sharp_translate_mpi_dtype (const char *mpi_dtype_str)
- int sharp_coll_reg_mr (struct sharp_coll_context *context, void *buf, int size, void **mr)
- int sharp_coll_dereg_mr (struct sharp_coll_context *context, void *mr)

- int sharp_coll_print_config (FILE *stream, enum config_print_flags print_flags)
- const char * sharp_coll_strerror (int error)
- int sharp_coll_dump_stats (struct sharp_coll_context *context)

5.1.5 Variables

- const struct sharp_coll_config sharp_coll_default_config

5.1.6 Typedef Documentation

5.1.6.1 typedef size_t(* sharp_stream_cb) (void *buffer, size_t length, size_t offset, void *context)

Stream callback

Streaming writer callback function.

5.1.6.1.1 Parameters:

buffer	Buffer to read or write the data
length	Max. number of bytes to copy.
offset	Offset in the data stream. First time called with offset = 0, and it's incremented by the return value.
context	User-define context, from stream.cb_context;

5.1.6.1.2 Returns:

Number of bytes processed from the buffer. For writers - if it's smaller than `length', it means to end the fragment prematurely, and the callback will be called again until total length is processed. For readers, the callback must process the entire buffer, if it returns a smaller value it means the message is truncated.

5.1.6.1.3 Note:

It's guaranteed that (offset + length) is not larger than the declared stream size (stream.length).

5.1.7 Enumeration Type Documentation

5.1.7.1 enum sharp_datatype

SHARP coll supported data types.

Copyright (C) Mellanox Technologies Ltd. 2015-2018. ALL RIGHTS RESERVED.

See file LICENSE for terms. The enumeration list describes the datatypes supported by SHARP coll

Enumerator

SHARP_DTYPE_UNSIGNED 32-bit unsigned integer.

SHARP_DTYPE_INT 32-bit integer.

SHARP_DTYPE_UNSIGNED_LONG 64-bit unsigned integer.

SHARP_DTYPE_LONG 64-bit integer.

SHARP_DTYPE_FLOAT 32-bit long floating point number

SHARP_DTYPE_DOUBLE 64-bit long floating point number.

SHARP_DTYPE_NULL NULL data type

5.1.7.2 enum sharp_reduce_op

SHARP coll supported aggregation operations.

The enumeration list describes the aggregation operations supported by SHARP coll

Enumerator

SHARP_OP_MAX maximum.

SHARP_OP_MIN minimum.

SHARP_OP_SUM sum.

SHARP_OP_PROD product.

SHARP_OP_LAND logical and.

SHARP_OP_BAND bit-wise and.

SHARP_OP_LOR logical or.

SHARP_OP_BOR bit-wise or.

SHARP_OP_LXOR logical xor.

SHARP_OP_BXOR bit-wise xor.

SHARP_OP_MAXLOC max value and location.

SHARP_OP_MINLOC min value and location.

SHARP_OP_NULL

5.1.7.3 enum sharp_error_no

SHARP coll status code.

The enumeration list describes the error codes returned by SHARP coll

Enumerator

SHARP_COLL_SUCCESS Success.

SHARP_COLL_ERROR Error.

SHARP_COLL_ENOT_SUPP Collective operation not supported.

SHARP_COLL_ENOMEM No memory.

SHARP_COLL_EGROUP_ALLOC SHARP Group alloc error.

SHARP_COLL_ECONN_TREE No connection to SHARP tree.

SHARP_COLL_EGROUP_JOIN Not able to join SHARP group.

SHARP_COLL_EQUOTA SHARP resource quota error.

SHARP_COLL_ESESS_INIT Cannot connect to SHARP.

SHARP_COLL_EDEV SHARP device error.

SHARP_COLL_EINVAL Invalid value.

SHARP_COLL_EJOB_CREATE Cannot create SHARP job.

SHARP_COLL_ETREE_INFO SHARP tree info not found.

SHARP_COLL_ENOTREE No available SHARP trees.

SHARP_COLL_EGROUP_ID Wrong SHARP group ID.

SHARP_COLL_EOOB Out-Of-Band collective error.

SHARP_COLL_EGROUP_MCAST Multicast target error.

5.1.7.4 enum sharp_data_buffer_type

SHARP coll buffer types.

The enumeration list describes the buffer types supported in collective calls.

5.1.7.4.1 Note:

Only SHARP_DATA_BUFFER is implemented.

Enumerator

SHARP_DATA_BUFFER Contiguous buffer.

SHARP_DATA_STREAM Data stream.

SHARP_DATA_IOV Vector input.

5.1.7.5 enum config_print_flags

brief SHARP coll print config flags

The enumeration list describes bit masks for different options to print config flags.

Enumerator

SHARP_COLL_CONFIG_PRINT_CONFIG basic configuration.

SHARP_COLL_CONFIG_PRINT_HEADER Print header.

SHARP_COLL_CONFIG_PRINT_DOC full description.

SHARP_COLL_CONFIG_PRINT_HIDDEN hidden options.

5.1.8 Function Documentation

5.1.8.1 int sharp_coll_init (struct sharp_coll_init_spec * sharp_coll_spec, struct sharp_coll_context ** sharp_coll_context)

brief SHARP coll context initialization

This routine is initialize SHARP coll library and create ref sharp_coll_context "SHARP coll context". This is a collective, called from all processes of the job.

warning An application cannot call any SHARP coll routine before sharp_coll_init

param [in] sharp_coll_spec SHARP coll specification descriptor. *param [out]* sharp_coll_context Initialized ref sharp_coll_context "SHArP coll context".

return Error code as defined by ref sharp_error_no

5.1.8.2 int sharp_coll_finalize (struct sharp_coll_context * context)

brief SHARP coll context finalize

This routine finalizes and releases the resources associated with ref sharp_coll_context "SHArP coll context". typically done once, just before the process ends.

warning An application cannot call any SHARP coll routine after sharp_coll_finalize

param [in] context SHARP coll context to cleanup.

return Error code as defined by ref sharp_error_no

5.1.8.3 int sharp_coll_progress (struct sharp_coll_context * context)

brief Progress SHARP coll communication operations.

This routine explicitly progresses all SHARP communication operation. This typically called from MPI progress context for MPI case.

param [in] context SHARP coll context to progress.

return Error code as defined by ref sharp_error_no

5.1.8.4 int sharp_coll_comm_init (struct sharp_coll_context * context, struct sharp_coll_comm_init_spec * spec, struct sharp_coll_comm ** sharp_coll_comm)

brief SHARP coll communicator(group) initialization

This routine creates ref sharp_coll_comm "SHARP coll group". This is a collective, called from all processes of the SHARP group.

param [in] context Handle to SHARP coll context. param [in] spec Input ref sharp_coll_comm_init_spec "SHArP coll group specification". *param [out]* sharp_coll_comm Handle to SHARP coll communicator(group)

return Error code as defined by ref sharp_error_no

5.1.8.5 int sharp_coll_comm_destroy (struct sharp_coll_comm * comm)

brief SHARP coll communicator cleanup

This routine cleanup SHARP coll communicator handle returned from ref sharp_coll_comm_init.

param [in] comm SHARP coll cCommunicator to destroy.

return Error code as defined by ref sharp_error_no

5.1.8.6 int sharp_coll_do_barrier (struct sharp_coll_comm * comm)

brief SHARP coll barrier collective

This routine is collective operation blocks until all processes call this routine.

param [in] comm SHARP coll communicator to run the barrier.

return Error code as defined by ref sharp_error_no

5.1.8.7 int sharp_coll_do_barrier_nb (struct sharp_coll_comm * comm, struct sharp_coll_request ** handle)

brief SHARP coll non-blocking barrier collective

This routine is non-blocking version of ref sharp_coll_do_barrier "SHArP coll barrier". The progress of this operation is tracked with return request handle

param [in] comm SHARP coll communicator to run the barrier. *param [out]* handle Handle representing the communication operation.

return Error code as defined by ref sharp_error_no

note Not implemented.

5.1.8.8 int sharp_coll_do_allreduce (struct sharp_coll_comm * comm, struct sharp_coll_reduce_spec * spec)

brief SHARP coll allreduce collective

This routine aggreate the data from all processes of the group and distributes the result back to all processes.

param [in] comm SHARP coll communicator to run the allreduce collective. *param [in]* spec Allreduce operation specification.

return Error code as defined by ref sharp_error_no

5.1.8.9 int sharp_coll_do_allreduce_nb (struct sharp_coll_comm * comm, struct sharp_coll_reduce_spec * spec, struct sharp_coll_request ** req)

brief SHARP coll non-blocking allreduce collective

This routine is non-blocking version of ref sharp_coll_do_allreduce "SHArP coll allreduce". The progress of this operation is tracked with return request handle

param [in] comm SHARP coll communicator to run the allreduce collective. *param [in]* spec Allreduce operation specification. *param [out]* handle Handle representing the communication operation.

eturn Error code as defined by ref sharp_error_no

note Not implemented.

5.1.8.10 int sharp_coll_do_reduce (struct sharp_coll_comm * comm, struct sharp_coll_reduce_spec * spec)

brief SHARP coll reduce collective

This routine aggregate the data from all processes of the group to a specific process

param [in] comm SHARP coll communicator to run the reduce collective. *param [in]* spec Reduce operation specification.

return Error code as defined by ref sharp_error_no

note Not implemented.

5.1.8.11 int sharp_coll_do_reduce_nb (struct sharp_coll_comm * comm, struct sharp_coll_reduce_spec * spec, struct sharp_coll_request ** handle)

brief SHARP coll non-blocking reduce collective

This routine is non-blocking version of ref sharp_coll_do_reduce "SHArP coll Reduce". The progress of this operation is tracked with return request handle

param [in] comm SHARP coll communicator to run the reduce collective. *param [in]* spec Allreduce operation specification. *param [out]* handle Handle representing the communication operation.

return Error code as defined by ref sharp_error_no

note Not implemented.

5.1.8.12 int sharp_coll_do_bcast (struct sharp_coll_comm * comm, struct sharp_coll_bcast_spec * spec)

brief SHARP coll broadcast collective

This routine broadcast data from single process to all processes of the group

param [in] comm SHARP coll communicator to run the bcast collective. *param [in]* spec Bcast operation specification.

return Error code as defined by ref sharp_error_no

note Not implemented.

5.1.8.13 int sharp_coll_do_bcast_nb (struct sharp_coll_comm * comm, struct sharp_coll_bcast_spec * spec, struct sharp_coll_request ** handle)

brief SHARP coll non-blocking broadcast collective

This routine is non-blocking version of ref sharp_coll_do_bcast "SHArP coll Bcast". The progress of this operation is tracked with return request handle

param [in] comm SHARP coll communicator to run the bcast collective. *param [in]* spec Bcast operation specification. *param [out]* handle Handle representing the communication operation.

return Error code as defined by ref sharp_error_no

note Not implemented.

5.1.8.14 int sharp_coll_req_test (struct sharp_coll_request * req)

brief SHARP coll request test

This routine tests for the completion of a specific non-blocking coll operation

param [in] req SHARP coll request handle

return Error code as defined by ref sharp_error_no

5.1.8.15 int sharp_coll_req_wait (struct sharp_coll_request * req)

brief SHARP coll request wait

This routine returns when the operation identified by non-blocking collective request is complete. The request object is deallocated by the call to sharp_coll_req_wait and the request handle is set to NULL.

param [in] req SHARP coll request handle

return Error code as defined by ref sharp_error_no

5.1.8.16 int sharp_coll_req_free (struct sharp_coll_request * req)

brief SHARP coll request deallocate

This routine deallocates request handle

param [in] req SHARP coll request handle

return Error code as defined by ref sharp_error_no

5.1.8.17 int sharp_translate_mpi_op (const char * mpi_op_str)

brief SHARP coll translate MPI OP to SHARP coll OP

This routine returns SHARP coll reduce operation ID for given MPI OP string if it is supported.

param [in] mpi_op_str MPI Operation name.

return SHARP reduce Op ID as defined by ref sharp_reduce_op on success, < 0 on failure

5.1.8.18 int sharp_translate_mpi_dtype (const char * mpi_dtype_str)

brief SHARP coll translate MPI datatype string to SHARP coll data type

This routine returns SHARP coll data type for given MPI data type string if it is supported.

param [in] mpi_dtype_str MPI Datatype string.

return SHARP coll datatype as defined by ref sharp_datatype on success, < 0 on failure.

5.1.8.19 int sharp_coll_reg_mr (struct sharp_coll_context * context, void * buf, int size, void ** mr)

brief SHARP coll memory registration.

This routine registers external mem buffer

param [in] context SHARP coll context to progress. *param [in]* buf Buffer to register *param [in]* size length of the buffer in bytes param [out] mr memory registration handle.

return Error code as defined by ref sharp_error_no

note Only one outstanding registration supported. No registration cache.

5.1.8.20 int sharp_coll_dereg_mr (struct sharp_coll_context * context, void * mr)

brief SHARP coll memory de-registration.

This routine de-registers the MR.

param [in] context SHARP coll context to progress. *param [in]* mr memory registration handle.

return Error code as defined by ref sharp_error_no

5.1.8.21 int sharp_coll_print_config (FILE * stream, enum config_print_flags print_flags)

brief SHARP coll print configuration

This routine prints SHARP coll configuration to a stream.

param [in] stream Output stream to print to. *param [in]* print_flags Controls how the configuration is printed.

return Error code as defined by ref sharp_error_no

5.1.8.22 const char* sharp_coll_strerror (int error)

brief SHARP coll print error string

This routine returns error string for a given ref sharp_error_no "SHArP coll error code".

param [in] error SHARP coll error code.

return Error string

5.1.8.23 int sharp_coll_dump_stats (struct sharp_coll_context * context)

brief SHARP coll print statistics

This routine dumps SHARP coll usage statistics

param [in] context SHARP coll context to progress.

return Error code as defined by ref sharp_error_no

note It is expected that Out-Of_Band collectives are operational valid to get accumulated stats dumps (SHARP_COLL_STATS_DUMP_MODE=2) during the finalize process.

5.1.9 Variable Documentation

5.1.9.1 const struct sharp_coll_config sharp_coll_default_config

Default SHARP COLL configuration.