

[NCCL]([../index.html](#))

[2.25](<https://docs.nvidia.com/deeplearning/sdk/nccl-archived/index.html>)

- * [Overview of NCCL]([../overview.html](#))

- * [Setup]([../setup.html](#))

- * [Using NCCL]([../usage.html](#))

- * [Creating a Communicator]([communicators.html](#))

- * [Creating a communicator with options]([communicators.html#creating-a-communicator-with-options](#))

- * [Creating a communicator using multiple ncclUniqueIds]([communicators.html#creating-a-communicator-using-multiple-nccluniqueids](#))

- * [Creating more communicators]([communicators.html#creating-more-communicators](#))

- * [Using multiple NCCL communicators concurrently]([communicators.html#using-multiple-nccl-communicators-concurrently](#))

- * [Finalizing a communicator]([communicators.html#finalizing-a-communicator](#))

- * [Destroying a communicator]([communicators.html#destroying-a-communicator](#))

- * [Error handling and communicator abort]([communicators.html#error-handling-and-communicator-abort](#))

- * [Asynchronous errors and error handling]([communicators.html#asynchronous-errors-and-error-handling](#))

- * [Fault Tolerance]([communicators.html#fault-tolerance](#))

- * [Collective Operations]([collectives.html](#))

- * [AllReduce]([collectives.html#allreduce](#))

- * [Broadcast]([collectives.html#broadcast](#))

- * [Reduce]([collectives.html#reduce](#))

- * [AllGather]([collectives.html#allgather](#))

- * [\[ReduceScatter\]\(collectives.html#reducescatter\)](#)
- * [\[Data Pointers\]\(data.html\)](#)
- * [\[CUDA Stream Semantics\]\(streams.html\)](#)
 - * [\[Mixing Multiple Streams within the same ncclGroupStart/End\(\) group\]\(streams.html#mixing-multiple-streams-within-the-same-ncclgroupstart-end-group\)](#)
- * [\[Group Calls\]\(groups.html\)](#)
 - * [\[Management Of Multiple GPUs From One Thread\]\(groups.html#management-of-multiple-gpus-from-one-thread\)](#)
 - * [\[Aggregated Operations \(2.2 and later\)\]\(groups.html#aggregated-operations-2-2-and-later\)](#)
 - * [\[Nonblocking Group Operation\]\(groups.html#nonblocking-group-operation\)](#)
- * [\[Point-to-point communication\]\(p2p.html\)](#)
 - * [\[Sendrecv\]\(p2p.html#sendrecv\)](#)
 - * [\[One-to-all \(scatter\)\]\(p2p.html#one-to-all-scatter\)](#)
 - * [\[All-to-one \(gather\)\]\(p2p.html#all-to-one-gather\)](#)
 - * [\[All-to-all\]\(p2p.html#all-to-all\)](#)
 - * [\[Neighbor exchange\]\(p2p.html#neighbor-exchange\)](#)
- * [\[Thread Safety\]\(threadsafety.html\)](#)
- * [\[In-place Operations\]\(inplace.html\)](#)
- * [\[Using NCCL with CUDA Graphs\]\(cudagraph.html\)](#)
- * [User Buffer Registration](#)
 - * [NVLink Sharp Buffer Registration](#)
 - * [IB Sharp Buffer Registration](#)
 - * [General Buffer Registration](#)
 - * [Memory Allocator](#)
- * [\[NCCL API\]\(../api.html\)](#)
 - * [\[Communicator Creation and Management Functions\]\(../api/comms.html\)](#)
 - * [\[ncclGetLastError\]\(../api/comms.html#ncclgetlasterror\)](#)

- * [\[ncclGetErrorString\]\(../api/comms.html#ncclgeterrorstring\)](#)
- * [\[ncclGetVersion\]\(../api/comms.html#ncclgetversion\)](#)
- * [\[ncclGetUniqueId\]\(../api/comms.html#ncclgetuniqueid\)](#)
- * [\[ncclCommInitRank\]\(../api/comms.html#ncclcomminitrank\)](#)
- * [\[ncclCommInitAll\]\(../api/comms.html#ncclcomminitall\)](#)
- * [\[ncclCommInitRankConfig\]\(../api/comms.html#ncclcomminitrankconfig\)](#)
- * [\[ncclCommInitRankScalable\]\(../api/comms.html#ncclcomminitrankscalable\)](#)
- * [\[ncclCommSplit\]\(../api/comms.html#ncclcommsplit\)](#)
- * [\[ncclCommFinalize\]\(../api/comms.html#ncclcommfinalize\)](#)
- * [\[ncclCommDestroy\]\(../api/comms.html#ncclcommdestroy\)](#)
- * [\[ncclCommAbort\]\(../api/comms.html#ncclcommabort\)](#)
- * [\[ncclCommGetAsyncError\]\(../api/comms.html#ncclcommgetasyncerror\)](#)
- * [\[ncclCommCount\]\(../api/comms.html#ncclcommcount\)](#)
- * [\[ncclCommCuDevice\]\(../api/comms.html#ncclcommcudevice\)](#)
- * [\[ncclCommUserRank\]\(../api/comms.html#ncclcommuserrank\)](#)
- * [\[ncclCommRegister\]\(../api/comms.html#ncclcommregister\)](#)
- * [\[ncclCommDeregister\]\(../api/comms.html#ncclcommderegister\)](#)
- * [\[ncclMemAlloc\]\(../api/comms.html#ncclmemalloc\)](#)
- * [\[ncclMemFree\]\(../api/comms.html#ncclmemfree\)](#)
- * [\[Collective Communication Functions\]\(../api/colls.html\)](#)
 - * [\[ncclAllReduce\]\(../api/colls.html#ncclallreduce\)](#)
 - * [\[ncclBroadcast\]\(../api/colls.html#ncclbroadcast\)](#)
 - * [\[ncclReduce\]\(../api/colls.html#ncclreduce\)](#)
 - * [\[ncclAllGather\]\(../api/colls.html#ncclallgather\)](#)
 - * [\[ncclReduceScatter\]\(../api/colls.html#ncclreducescatter\)](#)
- * [\[Group Calls\]\(../api/group.html\)](#)
 - * [\[ncclGroupStart\]\(../api/group.html#ncclgroupstart\)](#)

- * [\[ncclGroupEnd\]\(../api/group.html#ncclgroupend\)](#)
- * [\[ncclGroupSimulateEnd\]\(../api/group.html#ncclgroupsimulateend\)](#)
- * [\[Point To Point Communication Functions\]\(../api/p2p.html\)](#)
- * [\[ncclSend\]\(../api/p2p.html#ncclsend\)](#)
- * [\[ncclRecv\]\(../api/p2p.html#ncclrecv\)](#)
- * [\[Types\]\(../api/types.html\)](#)
- * [\[ncclComm_t\]\(../api/types.html#ncclcomm-t\)](#)
- * [\[ncclResult_t\]\(../api/types.html#ncclresult-t\)](#)
- * [\[ncclDataType_t\]\(../api/types.html#nccldatatype-t\)](#)
- * [\[ncclRedOp_t\]\(../api/types.html#ncclredop-t\)](#)
- * [\[ncclScalarResidence_t\]\(../api/types.html#ncclscalarresidence-t\)](#)
- * [\[ncclConfig_t\]\(../api/types.html#ncclconfig-t\)](#)
- * [\[ncclSimInfo_t\]\(../api/types.html#ncclsiminfo-t\)](#)
- * [\[User Defined Reduction Operators\]\(../api/ops.html\)](#)
- * [\[ncclRedOpCreatePreMulSum\]\(../api/ops.html#ncclredopcreatepremulsum\)](#)
- * [\[ncclRedOpDestroy\]\(../api/ops.html#ncclredopdestroy\)](#)
- * [\[Migrating from NCCL 1 to NCCL 2\]\(../nccl1.html\)](#)
- * [\[Initialization\]\(../nccl1.html#initialization\)](#)
- * [\[Communication\]\(../nccl1.html#communication\)](#)
- * [\[Counts\]\(../nccl1.html#counts\)](#)
- * [\[In-place usage for AllGather and ReduceScatter\]\(../nccl1.html#in-place-usage-for-allgather-and-reducescatter\)](#)
- * [\[AllGather arguments order\]\(../nccl1.html#allgather-arguments-order\)](#)
- * [\[Datatypes\]\(../nccl1.html#datatypes\)](#)
- * [\[Error codes\]\(../nccl1.html#error-codes\)](#)
- * [\[Examples\]\(../examples.html\)](#)
- * [\[Communicator Creation and Destruction\]](#)

Examples](../examples.html#communicator-creation-and-destruction-examples)

- * [Example 1: Single Process, Single Thread, Multiple Devices](../examples.html#example-1-single-process-single-thread-multiple-devices)

- * [Example 2: One Device per Process or Thread](../examples.html#example-2-one-device-per-process-or-thread)

- * [Example 3: Multiple Devices per Thread](../examples.html#example-3-multiple-devices-per-thread)

- * [Example 4: Multiple communicators per device](../examples.html#example-4-multiple-communicators-per-device)

- * [Communication Examples](../examples.html#communication-examples)

- * [Example 1: One Device per Process or Thread](../examples.html#example-1-one-device-per-process-or-thread)

- * [Example 2: Multiple Devices per Thread](../examples.html#example-2-multiple-devices-per-thread)

- * [NCCL and MPI](../mpi.html)

- * [API](../mpi.html#api)

- * [Using multiple devices per process](../mpi.html#using-multiple-devices-per-process)

- * [ReduceScatter operation](../mpi.html#reducescatter-operation)

- * [Send and Receive counts](../mpi.html#send-and-receive-counts)

- * [Other collectives and point-to-point operations](../mpi.html#other-collectives-and-point-to-point-operations)

- * [In-place operations](../mpi.html#in-place-operations)

- * [Using NCCL within an MPI Program](../mpi.html#using-nccl-within-an-mpi-program)

- * [MPI Progress](../mpi.html#mpi-progress)

- * [Inter-GPU Communication with CUDA-aware MPI](../mpi.html#inter-gpu-communication-with-cuda-aware-mpi)

- * [Environment Variables](../env.html)

* [System configuration](../env.html#system-configuration)

* [NCCL_SOCKET_IFNAME](../env.html#nccl-socket-ifname)

* [Values accepted](../env.html#values-accepted)

* [NCCL_SOCKET_FAMILY](../env.html#nccl-socket-family)

* [Values accepted](../env.html#id2)

* [NCCL_SOCKET_RETRY_CNT](../env.html#nccl-socket-retry-cnt)

* [Values accepted](../env.html#id3)

* [NCCL_SOCKET_RETRY_SLEEP_MSEC](../env.html#nccl-socket-retry-sleep-msec)

* [Values accepted](../env.html#id4)

* [NCCL_SOCKET_NTHREADS](../env.html#nccl-socket-nthreads)

* [Values accepted](../env.html#id5)

* [NCCL_NSOCKS_PERTHREAD](../env.html#nccl-nsocks-perthread)

* [Values accepted](../env.html#id6)

* [NCCL_CROSS_NIC](../env.html#nccl-cross-nic)

* [Values accepted](../env.html#id7)

* [NCCL_IB_HCA](../env.html#nccl-ib-hca)

* [Values accepted](../env.html#id8)

* [NCCL_IB_TIMEOUT](../env.html#nccl-ib-timeout)

* [Values accepted](../env.html#id9)

* [NCCL_IB_RETRY_CNT](../env.html#nccl-ib-retry-cnt)

* [Values accepted](../env.html#id10)

* [NCCL_IB_GID_INDEX](../env.html#nccl-ib-gid-index)

* [Values accepted](../env.html#id11)

* [NCCL_IB_ADDR_FAMILY](../env.html#nccl-ib-addr-family)

* [Values accepted](../env.html#id12)

* [NCCL_IB_ADDR_RANGE](../env.html#nccl-ib-addr-range)

* [Values accepted](../env.html#id13)

* [NCCL_IB_ROCE_VERSION_NUM](../env.html#nccl-ib-roce-version-num)
* [Values accepted](../env.html#id14)

* [NCCL_IB_SL](../env.html#nccl-ib-sl)
* [Values accepted](../env.html#id15)

* [NCCL_IB_TC](../env.html#nccl-ib-tc)
* [Values accepted](../env.html#id16)

* [NCCL_IB_FIFO_TC](../env.html#nccl-ib-fifo-tc)
* [Values accepted](../env.html#id17)

* [NCCL_IB_RETURN_ASYNC_EVENTS](../env.html#nccl-ib-return-async-events)
* [Values accepted](../env.html#id18)

* [NCCL_OOB_NET_ENABLE](../env.html#nccl-oob-net-enable)
* [Values accepted](../env.html#id19)

* [NCCL_OOB_NET_IFNAME](../env.html#nccl-oob-net-ifname)
* [Values accepted](../env.html#id20)

* [NCCL_UID_STAGGER_THRESHOLD](../env.html#nccl-uid-stagger-threshold)
* [Values accepted](../env.html#id21)

* [NCCL_UID_STAGGER_RATE](../env.html#nccl-uid-stagger-rate)
* [Values accepted](../env.html#id22)

* [NCCL_NET](../env.html#nccl-net)
* [Values accepted](../env.html#id23)

* [NCCL_NET_PLUGIN](../env.html#nccl-net-plugin)
* [Values accepted](../env.html#id24)

* [NCCL_TUNER_PLUGIN](../env.html#nccl-tuner-plugin)
* [Values accepted](../env.html#id25)

* [NCCL_PROFILER_PLUGIN](../env.html#nccl-profiler-plugin)
* [Values accepted](../env.html#id26)

* [NCCL_IGNORE_CPU_AFFINITY](../env.html#nccl-ignore-cpu-affinity)

- * [Values accepted](../env.html#id27)
- * [NCCL_CONF_FILE](../env.html#nccl-conf-file)
- * [Values accepted](../env.html#id28)
- * [NCCL_DEBUG](../env.html#nccl-debug)
- * [Values accepted](../env.html#id30)
- * [NCCL_DEBUG_FILE](../env.html#nccl-debug-file)
- * [Values accepted](../env.html#id31)
- * [NCCL_DEBUG_SUBSYS](../env.html#nccl-debug-subsys)
- * [Values accepted](../env.html#id32)
- * [NCCL_COLLNET_ENABLE](../env.html#nccl-collnet-enable)
- * [Value accepted](../env.html#value-accepted)
- * [NCCL_COLLNET_NODE_THRESHOLD](../env.html#nccl-collnet-node-threshold)
- * [Value accepted](../env.html#id33)
- * [NCCL_TOPO_FILE](../env.html#nccl-topo-file)
- * [Value accepted](../env.html#id34)
- * [NCCL_TOPO_DUMP_FILE](../env.html#nccl-topo-dump-file)
- * [Value accepted](../env.html#id35)
- * [NCCL_SET_THREAD_NAME](../env.html#nccl-set-thread-name)
- * [Value accepted](../env.html#id36)
- * [Debugging](../env.html#debugging)
- * [NCCL_P2P_DISABLE](../env.html#nccl-p2p-disable)
- * [Values accepted](../env.html#id37)
- * [NCCL_P2P_LEVEL](../env.html#nccl-p2p-level)
- * [Values accepted](../env.html#id38)
- * [Integer Values (Legacy)](../env.html#integer-values-legacy)
- * [NCCL_P2P_DIRECT_DISABLE](../env.html#nccl-p2p-direct-disable)
- * [Values accepted](../env.html#id39)

* [NCCL_SHM_DISABLE](../env.html#nccl-shm-disable)
* [Values accepted](../env.html#id40)

* [NCCL_BUFFSIZE](../env.html#nccl-buffersize)
* [Values accepted](../env.html#id41)

* [NCCL_NTHREADS](../env.html#nccl-nthreads)
* [Values accepted](../env.html#id42)

* [NCCL_MAX_NCHANNELS](../env.html#nccl-max-nchannels)
* [Values accepted](../env.html#id43)

* [NCCL_MIN_NCHANNELS](../env.html#nccl-min-nchannels)
* [Values accepted](../env.html#id44)

* [NCCL_CHECKS_DISABLE](../env.html#nccl-checks-disable)
* [Values accepted](../env.html#id45)

* [NCCL_CHECK_POINTERS](../env.html#nccl-check-pointers)
* [Values accepted](../env.html#id46)

* [NCCL_LAUNCH_MODE](../env.html#nccl-launch-mode)
* [Values accepted](../env.html#id47)

* [NCCL_IB_DISABLE](../env.html#nccl-ib-disable)
* [Values accepted](../env.html#id48)

* [NCCL_IB_AR_THRESHOLD](../env.html#nccl-ib-ar-threshold)
* [Values accepted](../env.html#id49)

* [NCCL_IB_QPS_PER_CONNECTION](../env.html#nccl-ib-qps-per-connection)
* [Values accepted](../env.html#id50)

* [NCCL_IB_SPLIT_DATA_ON_QPS](../env.html#nccl-ib-split-data-on-qps)
* [Values accepted](../env.html#id51)

* [NCCL_IB_CUDA_SUPPORT](../env.html#nccl-ib-cuda-support)
* [Values accepted](../env.html#id52)

* [NCCL_IB_PCI_RELAXED_ORDERING](../env.html#nccl-ib-pci-relaxed-ordering)

* [Values accepted](../env.html#id53)

* [NCCL_IB_ADAPTIVE_ROUTING](../env.html#nccl-ib-adaptive-routing)

* [Values accepted](../env.html#id54)

* [NCCL_IB_ECE_ENABLE](../env.html#nccl-ib-ece-enable)

* [Values accepted](../env.html#id55)

* [NCCL_MEM_SYNC_DOMAIN](../env.html#nccl-mem-sync-domain)

* [Values accepted](../env.html#id56)

* [NCCL_CUMEM_ENABLE](../env.html#nccl-cumem-enable)

* [Values accepted](../env.html#id57)

* [NCCL_CUMEM_HOST_ENABLE](../env.html#nccl-cumem-host-enable)

* [Values accepted](../env.html#id58)

* [NCCL_NET_GDR_LEVEL (formerly

NCCL_IB_GDR_LEVEL)](../env.html#nccl-net-gdr-level-formerly-nccl-ib-gdr-level)

* [Values accepted](../env.html#id59)

* [Integer Values (Legacy)](../env.html#id60)

* [NCCL_NET_GDR_READ](../env.html#nccl-net-gdr-read)

* [Values accepted](../env.html#id61)

* [NCCL_NET_SHARED_BUFFERS](../env.html#nccl-net-shared-buffers)

* [Value accepted](../env.html#id62)

* [NCCL_NET_SHARED_COMMS](../env.html#nccl-net-shared-comms)

* [Value accepted](../env.html#id63)

* [NCCL_SINGLE_RING_THRESHOLD](../env.html#nccl-single-ring-threshold)

* [Values accepted](../env.html#id64)

* [NCCL_LL_THRESHOLD](../env.html#nccl-ll-threshold)

* [Values accepted](../env.html#id65)

* [NCCL_TREE_THRESHOLD](../env.html#nccl-tree-threshold)

* [Values accepted](../env.html#id66)

* [NCCL_ALGO](../env.html#nccl-algo)

* [Values accepted](../env.html#id67)

* [NCCL_PROTO](../env.html#nccl-proto)

* [Values accepted](../env.html#id68)

* [NCCL_NVB_DISABLE](../env.html#nccl-nvb-disable)

* [Value accepted](../env.html#id69)

* [NCCL_PXN_DISABLE](../env.html#nccl-pxn-disable)

* [Value accepted](../env.html#id70)

* [NCCL_P2P_PXN_LEVEL](../env.html#nccl-p2p-pxn-level)

* [Value accepted](../env.html#id71)

* [NCCL_RUNTIME_CONNECT](../env.html#nccl-runtime-connect)

* [Value accepted](../env.html#id72)

* [NCCL_GRAPH_REGISTER](../env.html#nccl-graph-register)

* [Value accepted](../env.html#id74)

* [NCCL_LOCAL_REGISTER](../env.html#nccl-local-register)

* [Value accepted](../env.html#id75)

* [NCCL_LEGACY_CUDA_REGISTER](../env.html#nccl-legacy-cuda-register)

* [Value accepted](../env.html#id76)

* [NCCL_SET_STACK_SIZE](../env.html#nccl-set-stack-size)

* [Value accepted](../env.html#id77)

* [NCCL_GRAPH_MIXING_SUPPORT](../env.html#nccl-graph-mixing-support)

* [Value accepted](../env.html#id79)

* [NCCL_DMABUF_ENABLE](../env.html#nccl-dmabuf-enable)

* [Value accepted](../env.html#id80)

* [NCCL_P2P_NET_CHUNKSIZE](../env.html#nccl-p2p-net-chunksize)

* [Values accepted](../env.html#id81)

* [NCCL_P2P_LL_THRESHOLD](../env.html#nccl-p2p-ll-threshold)

- * [Values accepted](../env.html#id82)
- * [NCCL_ALLOC_P2P_NET_LL_BUFFERS](../env.html#nccl-alloc-p2p-net-ll-buffers)
 - * [Values accepted](../env.html#id83)
- * [NCCL_COMM_BLOCKING](../env.html#nccl-comm-blocking)
 - * [Values accepted](../env.html#id84)
- * [NCCL_CGA_CLUSTER_SIZE](../env.html#nccl-cga-cluster-size)
 - * [Values accepted](../env.html#id85)
- * [NCCL_MAX_CTAS](../env.html#nccl-max-ctas)
 - * [Values accepted](../env.html#id86)
- * [NCCL_MIN_CTAS](../env.html#nccl-min-ctas)
 - * [Values accepted](../env.html#id87)
- * [NCCL_NVLS_ENABLE](../env.html#nccl-nvls-enable)
 - * [Values accepted](../env.html#id88)
- * [NCCL_IB_MERGE_NICS](../env.html#nccl-ib-merge-nics)
 - * [Values accepted](../env.html#id89)
- * [NCCL_MNNVL_ENABLE](../env.html#nccl-mnnvl-enable)
 - * [Values accepted](../env.html#id90)
- * [NCCL_RAS_ENABLE](../env.html#nccl-ras-enable)
 - * [Values accepted](../env.html#id91)
- * [NCCL_RAS_ADDR](../env.html#nccl-ras-addr)
 - * [Values accepted](../env.html#id92)
- * [NCCL_RAS_TIMEOUT_FACTOR](../env.html#nccl-ras-timeout-factor)
 - * [Values accepted](../env.html#id93)
- * [Troubleshooting](../troubleshooting.html)
 - * [Errors](../troubleshooting.html#errors)
 - * [RAS](../troubleshooting.html#ras)
 - * [RAS](../troubleshooting/ras.html)

- * [\[Principle of Operation\]\(../troubleshooting/ras.html#principle-of-operation\)](#)
- * [\[RAS Queries\]\(../troubleshooting/ras.html#ras-queries\)](#)
- * [\[Sample Output\]\(../troubleshooting/ras.html#sample-output\)](#)
- * [\[GPU Direct\]\(../troubleshooting.html#gpu-direct\)](#)
- * [\[GPU-to-GPU communication\]\(../troubleshooting.html#gpu-to-gpu-communication\)](#)
- * [\[GPU-to-NIC communication\]\(../troubleshooting.html#gpu-to-nic-communication\)](#)
- * [\[PCI Access Control Services \(ACS\)\]\(../troubleshooting.html#pci-access-control-services-ac\)](#)
- * [\[Topology detection\]\(../troubleshooting.html#topology-detection\)](#)
- * [\[Shared memory\]\(../troubleshooting.html#shared-memory\)](#)
- * [\[Docker\]\(../troubleshooting.html#docker\)](#)
- * [\[Systemd\]\(../troubleshooting.html#systemd\)](#)
- * [\[Networking issues\]\(../troubleshooting.html#networking-issues\)](#)
- * [\[IP Network Interfaces\]\(../troubleshooting.html#ip-network-interfaces\)](#)
- * [\[IP Ports\]\(../troubleshooting.html#ip-ports\)](#)
- * [\[InfiniBand\]\(../troubleshooting.html#infiniband\)](#)
- * [\[RDMA over Converged Ethernet \(RoCE\)\]\(../troubleshooting.html#rdma-over-converged-ethernet-roce\)](#)

[__\[NCCL\]\(../index.html\)](#)

- * [\[Docs\]\(../index.html\)](#) »
- * [\[Using NCCL\]\(../usage.html\)](#) »
- * [User Buffer Registration](#)
- * [\[View page source\]\(../_sources/usage/bufferreg.rst.txt\)](#)

* * *

User Buffer Registration¶

User Buffer Registration is a feature that allows NCCL to directly send/receive/operate data through the user buffer without extra internal copy (zero-copy). It can accelerate collectives and greatly reduce the resource usage (e.g. #channel usage). NCCL provides two ways to register user buffers; one is `_CUDA Graph_` registration, and the other is `_Local_` registration. NCCL requires that for all NCCL communication function calls (e.g., `allreduce`, `sendrecv`, and so on), if any rank in a communicator passes registered buffers to a NCCL communication function, all other ranks in the same communicator must pass their registered buffers; otherwise, mixing registered and non-registered buffers can result in undefined behavior.

NVLink Sharp Buffer Registration¶

Since 2.19.x, NCCL supports user buffer registration for NVLink Sharp (NVLS); any NCCL collectives (e.g., `allreduce`) that support NVLS algorithm can utilize this feature.

To enable the `_CUDA Graph_` based buffer registration for NVLS, users have to comply with several requirements:

- > * The buffer is allocated through
 - > [`ncclMemAlloc()`](../api/comms.html#c.ncclMemAlloc "ncclMemAlloc") or a
 - > qualified allocator (see Memory Allocator).
- > * The NCCL operation is launched on a stream captured by a CUDA graph for
- > each rank.

> * Offset to the head address of the buffer is the same in collectives for
> each rank.
>

Registered buffers will be deregistered when the CUDA graph is destroyed. Here
is a CUDA graph based buffer registration example:

```
void* sendbuff;

void* recvbuff;

size_t count = 1 << 25;

CHECK(ncclMemAlloc(&sendbuff, count * sizeof(float)));
CHECK(ncclMemAlloc(&recvbuff, count * sizeof(float)));

cudaGraph_t graph;

CHECK(cudaStreamBeginCapture(stream, cudaStreamCaptureModeThreadLocal));
CHECK(ncclAllReduce(sendbuff, recvbuff, 1024, ncclFloat, ncclSum, comm, stream));
// Same offset to the sendbuff and recvbuff head address for each rank
CHECK(ncclAllReduce((void*)((float*)sendbuff + 1024), (void*)((float*)recvbuff + 2048), 1024,
ncclFloat, ncclSum, comm, stream));

CHECK(cudaStreamEndCapture(stream, &graph));

cudaGraphExec_t instance;

CHECK(cudaGraphInstantiate(&instance, graph, NULL, NULL, 0));

CHECK(cudaGraphLaunch(instance, stream));

CHECK(cudaStreamSynchronize(stream));
```

```
CHECK(cudaGraphExecDestroy(instance));
```

```
CHECK(cudaGraphDestroy(graph));
```

```
CHECK(ncclMemFree(sendbuff));
```

```
CHECK(ncclMemFree(recvbuff));
```

On the other hand, to enable the `_Local_` based buffer registration for NVLS, users have to comply with the following requirements:

- > * The buffer is allocated through
 - > [`ncclMemAlloc()`](../api/comms.html#c.ncclMemAlloc "ncclMemAlloc") or a
 - > qualified allocator (see Memory Allocator).
- > * Register buffer with
 - > [`ncclCommRegister()`](../api/comms.html#c.ncclCommRegister
 - > "ncclCommRegister") before calling collectives for each rank.
- > * Call NCCL collectives as usual but similarly keep the offset to the head
- > address of the buffer the same for each rank.
- >

Registered buffers will be deregistered when users explicitly call

```
[ncclCommDeregister()](../api/comms.html#c.ncclCommDeregister
```

```
"ncclCommDeregister"). Here is a local based buffer registration example:
```

```
void* sendbuff;
```



```

void* recvbuff;

size_t count = 1 << 25;

void* sendRegHandle;

void* recvRegHandle;

CHECK(ncclMemAlloc(&sendbuff, count * sizeof(float)));

CHECK(ncclMemAlloc(&recvbuff, count * sizeof(float)));


CHECK(ncclCommRegister(comm, sendbuff, count * sizeof(float), &sendRegHandle));

CHECK(ncclCommRegister(comm, recvbuff, count * sizeof(float), &recvRegHandle));


CHECK(ncclAllReduce(sendbuff, recvbuff, 1024, ncclFloat, ncclSum, comm, stream));

    CHECK(ncclAllReduce((void*)((float*)sendbuff + 1024), (void*)((float*)recvbuff + 2048), 1024,
ncclFloat, ncclSum, comm, stream));

CHECK(cudaStreamSynchronize(stream));


CHECK(ncclCommDeregister(comm, sendRegHandle));

CHECK(ncclCommDeregister(comm, recvRegHandle));


CHECK(ncclMemFree(sendbuff));

CHECK(ncclMemFree(recvbuff));

```

For local based registration, users can register the buffer once at the beginning of the program and reuse the buffer multiple times to utilize registration benefits.

To save the memory, it is also valid to allocate a large chunk of buffer and

register it once. sendbuff and recvbuff can be further allocated through the big chunk for zero-copy NCCL operations as long as sendbuff and recvbuff satisfy the offset requirements. The following example shows a use case:

```
void* buffer;

void* handle;

void* sendbuff;

void* recvbuff;

size_t size = 1 << 29;

CHECK(ncclMemAlloc(&buffer, size));

CHECK(ncclCommRegister(comm, buffer, size, &handle));

// assign buffer chunk to sendbuff and recvbuff

sendbuff = buffer;

recvbuff = (void*)((uint8_t*)buffer + (1 << 20));

CHECK(ncclAllReduce(sendbuff, recvbuff, 1024, ncclFloat, ncclSum, comm, stream));

CHECK(cudaStreamSynchronize(stream));

CHECK(ncclCommDeregister(comm, handle));

CHECK(ncclMemFree(sendbuff));
```

IB Sharp Buffer Registration¶

NCCL 2.21.x supports IB Sharp buffer registration, any NCCL collectives that support IB Sharp algorithm can benefit from the feature such as allreduce, reducescatter, and allgather. Currently, NCCL only supports IB Sharp buffer registration for the communicators which contain 1 rank per node, and the registration can reduce the number of NCCL SM usage down to 1.

To enable IB Sharp buffer registration by CUDA graph:

- > * Allocate send and recv buffer with any CUDA allocator (e.g.,
> cudaMalloc/nccclMemAlloc)
- > * Launch NCCL collectives with CUDA graph
- >

To enable IB Sharp buffer registration by local registration:

- > * Allocate send and recv buffer with any CUDA allocator (e.g.,
> cudaMalloc/nccclMemAlloc)
- > * Register send and recv buffer for each rank in the communicator with
> ncclCommRegister
- > * Launch NCCL collectives
- >

General Buffer Registration¶

Since 2.23.x, NCCL supports intra-node buffer registration, which targets all

peer-to-peer intra-node communications and brings less memory access, fewer SM usage and performance improvement. Either registering buffers by `ncclCommRegister` in the beginning or applying CUDA graph can enable intra-node buffer registration for NCCL collectives and `sendrecv`. The registered buffers can be allocated through legacy cuda API (e.g., `cudaMalloc`) as well as VMM API (e.g., `cuMem*` or `ncclMemAlloc`). However, VMM-allocated buffers are highly recommended since it is safer than legacy buffers during failure and abort.

Memory Allocator¶

For convenience, NCCL provides `ncclMemAlloc` function to help users to allocate buffers through VMM API, which can be used for NCCL registration later. It is only designed for NCCL so that it is not recommended to use `ncclMemAlloc` allocated buffers everywhere in the applications. For advanced users, if you want to create your own memory allocator for NVLS buffer registration, the allocator needs to satisfy the following requirements:

- > * Allocate buffer with shared flag
- > `CU_MEM_HANDLE_TYPE_POSIX_FILE_DESCRIPTOR` and also `CU_MEM_HANDLE_TYPE_FABRIC`
- > on GPUs where itâ€™s supported.
- > * Buffer size is multiple of multicast recommended granularity (i.e. `cuMulticastGetGranularity(â€¦, CU_MULTICAST_GRANULARITY_RECOMMENDED)`)
- > * Buffer head address is at least aligned to multicast minimal granularity (i.e. `cuMulticastGetGranularity(â€¦, CU_MULTICAST_GRANULARITY_MINIMUM)`)
- >

[\[Next \]\(../api.html "NCCL API"\)](#) [\[Previous\]\(cudagraph.html "Using NCCL with
CUDA Graphs"\)](#)

* * *

(C) Copyright 2020, NVIDIA Corporation

Built with [\[Sphinx\]](http://sphinx-doc.org/) using a

[\[theme\]\(https://github.com/rtd/sphinx_rtd_theme\)](https://github.com/rtd/sphinx_rtd_theme) provided by [\[Read the
Docs\]\(https://readthedocs.org\)](https://readthedocs.org).