# NVIDIA NVSHMEM

## Installation Guide

DG-08911-001_v2.0.3 | January 2021

# Table of Contents

# Chapter 1. Overview

NVIDIA® NVSHMEM™ is a programming interface that implements a Partitioned Global Address Space (PGAS) model across a cluster of NVIDIA GPUs. NVSHMEM provides an easy-to-use interface to allocate memory that is symmetrically distributed across the GPUs. In addition to a CPU-side interface, NVSHMEM also provides a CUDA kernel-side interface that allows CUDA® threads to access any location in the symmetrically-distributed memory.

# Chapter 2. Hardware And Software Requirements

NVIDIA® NVSHMEM™ has the following hardware and software requirements.

## 2.1. Hardware Requirements

NVSHMEM requires the following hardware:

- The `x86_64` or `ppc64leCPU` architecture.
- NVIDIA® Data Center GPU of NVIDIA Volta™ GPU architecture or later.

  Refer to https://developer.nvidia.com/cuda-gpus for a complete list.
- All GPUs must be P2P-connected via NVLink/PCIe or via GPUDirect RDMA over InfiniBand/RoCE with a Mellanox adapter (CX-4 or later).

  Support for atomics requires a NVLink connection or a GPUDirect RDMA connection and GDRCopy. See Software Requirements for more information.

## 2.2. Software Requirements

NVSHMEM requires the following software:

- 64-bit Linux.

  For a complete compatibility matrix, see the NVIDIA CUDA Installation Guide for Linux .
- CUDA 10.1 or later.
- Mellanox OFED.
- `nv_peer_mem for GPUDirect RDMA`.
- PMI-1 (for example, Hydra), PMI-2 (for example, slurm), or a PMIx compatible launcher.
- **(Optional)**GDRCopy v2.0 or newer.

  This software is required for atomics support on non-NVLink connections.

# Chapter 3. Installation

## 3.1. Downloading NVSHMEM

### Procedure

To download NVSHMEM, go to NVSHMEM Downloads.

The extracted directory contains the following files and subdirectories:

| File or Directory | Description |
|---|---|
| `src/` | Contains NVSHMEM sources and headers. |
| `perftest/` | Contains tests showing use of NVSHMEM APIs with performance reporting. |
| `examples/` | Contains examples showing use of some common use cases of NVSHMEM. |
| `scripts/` | Contains helper scripts, for example, script to download, build and install Hydra. |
| `COPYRIGHT.txt` | Copyright information. |
| `NVSHMEM-SLA.txt` | NVSHMEM service level agreement (SLA). |

## 3.2. Building And Installing NVSHMEM

### Procedure

1. Set the `CUDA_HOME` environment variable to point to the CUDA Toolkit.
2. Set the `GDRCOPY_HOME` environment variable to point to the GDRCopy installation.

   To build without GDRCopy, set the environmental variable to `NVSHMEM_USE_GDRCOPY=0`.

   **Note:** Without GDRCopy, atomics are only supported across NVLink connections.

3. If MPI and/or SHMEM support is required, set `NVSHMEM_MPI_SUPPORT=1` and/or `NVSHMEM_SHMEM_SUPPORT=1`.

4. Set the `MPI_HOME` and `SHMEM_HOME` environment variables to point to the MPI and OpenSHMEM installations, respectively.
5. If NCCL will be used for host-initiated collectives, set `NVSHMEM_USE_NCCL=1` and `NCCL_HOME` to point to the NCCL installation.

   You can use any NCCL 2.x version, and NVSHMEM hast been tested with NCCL 2.8.3-1.
6. If the MPI library is neither OpenMPI nor its derivative, set `NVSHMEM_MPI_IS_OMPI=0`.

> **Note:** Here is some additional information:
>
> - When using OpenMPI and OSHMEM, the paths are the same.
>
>   To use OSHMEM, OpenMPI needs to be built with UCX support.
> - NVSHMEM has been tested with OpenMPI 4.0.1 and UCX 1.8.0.
> - Other MPI and OpenSHMEM installations should also work.
> - By default, MPI support is enabled, and OpenSHMEM support is disabled.

7. If PMIx support is required, set `NVSHMEM_PMIX_SUPPORT=1` and `PMIX_HOME` to point to the PMIx installation.

> **Note:**
>
> - PMI-1 and PMI-2 support is always included in the build, and PMI-1 is the default PMI.
>
>   To change the default `PMI NVSHMEM_DEFAULT_PMIX=1` (to select PMIx) or `NVSHMEM_DEFAULT_PMI2=1` (to select PMI-2) can be set. At runtime, `NVSHMEM_BOOTSTRAP_PMI` can be used to override the default. The possible values are PMIX, PMI-2, and PMI.
> - OpenMPI ships with its own copy of PMIx.
>
>   To avoid conflicting PMIx shared libraries, we recommend that you build a standalone PMIx and configure OpenMPI with `--with-pmix=external` to point to that installation.

8. To specify the location where NVSHMEM will be installed, set `NVSHMEM_PREFIX`.
9. To change the directory where NVSHMEM will be built, set `NVSHMEM_BUILDDIR`.

   The default is `NVSHMEM_DIR/build`.
10. To build and install the library, run `make -j8 install`.

# 3.3. Using NVSHMEM In Your Applications

## 3.3.1. Using NVSHMEM With Your C Or C++ Program

### Procedure

1. Include `nvshmem.h` and `nvshmemx.h` from `include/`.
2. Point to the `include/` and `lib/` paths.

3. NVSHMEM users: If your C or C++ program only uses NVSHMEM, install Hydra Process Manager using the bash script `install_hydra.sh` under the `scripts/` directory. Provide the download and install location as arguments, for example:

   ```
   ./install_hydra.sh <download_path> <install_path>
   ```

   Use `nvshmrun` launcher under `bin/` (of the Hydra install path) to run the NVSHMEM job.

## 3.3.2. Using NVSHMEM With Your MPI Program

### About this task

**Note:** The only MPI library currently tested is OpenMPI, however, derivatives of OpenMPI such as SpectrumMPI as well as MPICH derivatives are expected to work.

To run a Hybrid MPI + NVSHMEM program, use the `mpirun` launcher available in the MPI installation.

Similarly, NVSHMEM can be used from OpenSHMEM programs. In this case, you cannot use the launchers that are in the NVSHMEM package. The only OpenSHMEM version that has been tested is OSHMEM in OpenMPI. Other OpenSHMEM implementations such as Sandia OpenSHMEM (SOS) should also work. To run the hybrid OpenSHMEM/NVSHMEM job, use the `oshrun` launcher in the OpenMPI installation or follow the launcher specification of your OpenSHMEM library.

# 3.4. Running Performance Tests

Before you can run performance tests, you first must build them.

### Procedure

1. Set the `CUDA_HOME`, `NVSHMEM_HOME` and `MPI_HOME` (if the NVSHMEM library was built with `NVSHMEM_MPI_SUPPORT=1`) environment variables to build NVSHMEM performance tests:

   ```
   CUDA_HOME=<path to supported CUDA installation>
   NVSHMEM_HOME=<path to directory where NVSHMEM is installed>
   MPI_HOME=<path to MPI installation>
   ```

   Configuring OpenMPI using the `-with-ucx` option is required for OpenMPI/OSHMEM interoperability. If you have built NVSHMEM with MPI and OpenSHMEM support (`NVSHMEM_MPI_SUPPORT=1` and `NVSHMEM_SHMEM_SUPPORT=1`), building `perftest/` also requires MPI and OpenSHMEM support to be enabled.

   **Build without SHMEM interoperability:** To build NVSHMEM performance tests without SHMEM interoperability, set the environment variable `NVSHMEM_SHMEM_SUPPORT` to `0`. By default, performance tests are installed under `perftest/perftest_install`. To install to a different path, set `NVSHMEM_PERFTEST_INSTALL` to point to the desired path.
2. Update `LD_LIBRARY_PATH` to point to `$CUDA_HOME/lib64` and `$MPI_HOME/lib`.

3. Run performance tests as NVSHMEM jobs (assuming Hydra is installed under `HYDRA_HOME`), hybrid MPI+NVSHMEM jobs, or hybrid OpenSHMEM+NVSHMEM jobs with the following commands (using `perftest/device/pt-to-pt/put.cu` as an example):

   NVSHMEM job using Hydra (PM-1):

   ```
   $HYDRA_HOME/bin/nvshmrun -n <up to number of P2P or InfiniBand
   NIC accessible GPUs>
   $NVSHMEM_PERFTEST_INSTALL/device/pt-to-pt/shmem_put_bw
   ```

   NVSHMEM job using slurm:

   ```
   srun -n <up to number of P2P or InfiniBand NIC accessible GPUs>
   $NVSHMEM_PERFTEST_INSTALL/device/pt-to-pt/shmem_put_bw
   ```

   **Note:** When slurm was built with a PMI that does not match the default of NVSHMEM, for example, if slurm was built with PMIx support and `NVSHMEM_DEFAULT_PMIX=1` was not set when building NVSHMEM, `NVSHMEM_BOOTSTRAP_PMI` can be used to override the default. Possible values are PMIX, PMI-2, and PMI.

   Hybrid MPI/NVSHMEM job:

   ```
   $MPI_HOME/bin/mpirun -n <up to number of GPUs accessible by P2P
   or InfiniBand NIC> -x NVSHMEMTEST_USE_MPI_LAUNCHER=1
   $NVSHMEM_PERFTEST_INSTALL/device/pt-to-pt/shmem_put_bw
   ```

   Hybrid OpenSHMEM/NVSHMEM job:

   ```
   $MPI_HOME/bin/oshrun -n <up to number of GPUs accessible by P2P
   or InfiniBand NIC> -x USE_SHMEM_IN_TEST=1
   $NVSHMEM_PERFTEST_INSTALL/device/pt-to-pt/shmem_put_bw
   ```

## 3.5. "Hello World" Example

### Procedure

1. Save the following code as `nvshmemHelloWorld.cu`:

   ```
   #include <stdio.h>
   #include <cuda.h>
   #include <nvshmem.h>
   #include <nvshmemx.h>

   __global__ void simple_shift(int *destination) {
       int mype = nvshmem_my_pe();
       int npes = nvshmem_n_pes();
       int peer = (mype + 1) % npes;

       nvshmem_int_p(destination, mype, peer);
   }

   int main(void) {
       int mype_node, msg;
       cudaStream_t stream;

       nvshmem_init();
       mype_node = nvshmem_team_my_pe(NVSHMEMX_TEAM_NODE);
       cudaSetDevice(mype_node);
       cudaStreamCreate(&stream);

       int *destination = (int *) nvshmem_malloc(sizeof(int));

       simple_shift<<<1, 1, 0, stream>>>(destination);
   ```

```
    nvshmemx_barrier_all_on_stream(stream);
    cudaMemcpyAsync(&msg, destination, sizeof(int), cudaMemcpyDeviceToHost,
 stream);

    cudaStreamSynchronize(stream);
    printf("%d: received message %d\n", nvshmem_my_pe(), msg);

    nvshmem_free(destination);
    nvshmem_finalize();
    return 0;
}
```

2. Build `nvshmemHelloWorld.cu` with the following command:

```
nvcc -rdc=true -ccbin g++ -gencode=$NVCC_GENCODE -I $NVSHMEM_HOME/include
 nvshmemHelloWorld.cu -o nvshmemHelloWorld.out -L $NVSHMEM_HOME/lib -lnvshmem
-lcuda
```

   Where `arch=compute_70,code=sm_70` is the value of `NVCC_GENCODE` for V100 GPUs.

3. Run the `nvshmemHelloWorld` sample with one of the following commands:

   When running on a single host with 2 GPUs (connected by PCI-E, NVLink or Infiniband):

```
$HYDRA_HOME/bin/nvshmrun -n 2 -ppn 2 ./nvshmemHelloWorld.out
```

   When running on two hosts with 1 GPU per host (connected by InfiniBand):

```
$HYDRA_HOME/bin/nvshmrun -n 2 -ppn 1 --hosts hostname1,hostname2 ./
nvshmemHelloWorld.out
```

# Chapter 4. Support

Report bugs and submit feature requests using NVONLINE or by emailing nvshmem@nvidia.com.

## Notice

This document is provided for information purposes only and shall not be regarded as a warranty of a certain functionality, condition, or quality of a product. NVIDIA Corporation ("NVIDIA") makes no representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assumes no responsibility for any errors contained herein. NVIDIA shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This document is not a commitment to develop, release, or deliver any Material (defined below), code, or functionality.

This document is provided for information purposes only and shall not be regarded as a warranty of a certain functionality, condition, or quality of a product. NVIDIA Corporation ("NVIDIA") makes no representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assumes no responsibility for any errors contained herein. NVIDIA shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This document is not a commitment to develop, release, or deliver any Material (defined below), code, or functionality.

Customer should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

NVIDIA products are sold subject to the NVIDIA standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of NVIDIA and customer ("Terms of Sale"). NVIDIA hereby expressly objects to applying any customer general terms and conditions with regards to the purchase of the NVIDIA product referenced in this document. No contractual obligations are formed either directly or indirectly by this document.

NVIDIA products are not designed, authorized, or warranted to be suitable for use in medical, military, aircraft, space, or life support equipment, nor in applications where failure or malfunction of the NVIDIA product can reasonably be expected to result in personal injury, death, or property or environmental damage. NVIDIA accepts no liability for inclusion and/or use of NVIDIA products in such equipment or applications and therefore such inclusion and/or use is at customer's own risk.

NVIDIA makes no representation or warranty that products based on this document will be suitable for any specified use. Testing of all parameters of each product is not necessarily performed by NVIDIA. It is customer's sole responsibility to evaluate and determine the applicability of any information contained in this document, ensure the product is suitable and fit for the application planned by customer, and perform the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer's product designs may affect the quality and reliability of the NVIDIA product and may result in additional or different conditions and/or requirements beyond those contained in this document. NVIDIA accepts no liability related to any default, damage, costs, or problem which may be based on or attributable to: (i) the use of the NVIDIA product in any manner that is contrary to this document or (ii) customer product designs.

## VESA DisplayPort

DisplayPort and DisplayPort Compliance Logo, DisplayPort Compliance Logo for Dual-mode Sources, and DisplayPort Compliance Logo for Active Cables are trademarks owned by the Video Electronics Standards Association in the United States and other countries.

## HDMI

HDMI, the HDMI logo, and High-Definition Multimedia Interface are trademarks or registered trademarks of HDMI Licensing LLC.

## OpenCL

OpenCL is a trademark of Apple Inc. used under license to the Khronos Group Inc.

NVIDIA Corporation | 2788 San Tomas Expressway, Santa Clara, CA 95051
www.nvidia.com

### Trademarks



### Copyright



NVIDIA Corporation | 2788 San Tomas Expressway, Santa Clara, CA 95051
www.nvidia.com