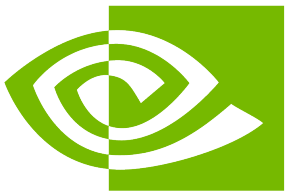

GDS Support for NVME Drives Using Linux PCI P2PDMA Support

Release 615

NVIDIA Corporation

Sep 09, 2026

Contents

Chapter 1. Intro

This document describes how to configure Linux to enable peer-to-peer DMA support on PCI for NVMe drives on an x86_64 platform.

Note

Arm64-based GraceHopper platforms, RAID0 over NVMe and NVMEoF are not supported in this release.

Chapter 2. References

https://docs.nvidia.com/gpudirect-storage/index.html

Chapter 3. Pre-installation Steps

3.1. NVIDIA GPUs

Check if you have a system with NVIDIA GPU newer than or equal to NVIDIA Ampere GPU architecture.

A100, A40, L4, L40S, and H100 are supported.

3.2. NVME

Check to make sure your system has standard Linux kernel NVME driver installed and you have NVME drives:

```
$ nvme list -v
$ lsblk
$ lspci -nnvv -d ::0108
```

3.3. Linux Kernel Version

Check that the Linux kernel version is newer than 6.2 or above on Ubuntu distributions. For other distributions please check the PCI P2PDMA feature is compiled. If the PCI P2PDMA is not enabled, GDS-specific NVMe patches can be installed from MLNX_OFED to support GDS with `nvidia-fs.ko`.

```
$ cat /proc/kallsyms | grep -i p2pdma_pgmap_ops
0000000000000000 d p2pdma_pgmap_ops
```

3.4. Installing Open RM Driver 565 or Above

https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html#driver-installation

Install OpenRM version 565.47 or above for PCI_P2PDMA support.

Note

OpenRM is default from version 560 and above. To check the installation, run:

```
$ cat /proc/driver/nvidia/version  | grep -i open
NVRM version: NVIDIA UNIX Open Kernel Module for x86_64  565.xx
```

Chapter 4. Setting the Driver Registries for Enabling PCI P2PDMA

Disable multipathing support

NVMe Multipathing is currently not enabled with PCI P2PDMA in the upstream driver and will not work without a specialized patch.

```
$ cat /etc/modprobe.d/nvme.conf
options nvme_core multipath=N

#RH/SLES:

     $ sudo dracut -f

#ubuntu:

     $ sudo update-initramfs -u -k `uname -r`

$ sudo reboot

$ cat /sys/module/nvme_core/parameters/multipath
N
```

pci_p2pdma support (Enable static BAR1 and force disable write combine)

Set these params in driver modprobe conf settings to persist beyond reboots:

```
$ cat /etc/modprobe.d/nvidia-temp.conf

options nvidia NVreg_RegistryDwords="RMForceStaticBar1=1;
↪RmForceDisableIomapWC=1;"
```

Note

For pre-Hopper GPUs (L4, L40, A100, A40), the following additional setting of `ForceP2P=0` should be applied:

```
$ cat /etc/modprobe.d/nvidia-p2pdma.conf

options nvidia NVreg_RegistryDwords="RMForceStaticBar1=1;ForceP2P=0;
```

(continues on next page)

(continued from previous page)

```
↪RmForceDisableIomapWC=1;"

#RH/SLES:

   $ sudo dracut -f

#ubuntu :

   $ sudo update-initramfs -u -k `uname -r`

# reboot

   $ sudo reboot

# check the settings

   $ cat /proc/driver/nvidia/params  | grep -i static
   RegistryDwords: "RMForceStaticBar1=1;RmForceDisableIomapWC=1;"
```

Chapter 5. Installing CUDA Toolkit

1. For detailed steps for installing CUDA Toolkit on Ubuntu, refer to https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html#ubuntu
2. For detailed steps for installing CUDA Toolkit on Redhat, refer to https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html#rhel-rocky
3. Install CUDA Toolkit 12.7 and make sure `libcufile-12-7` and `gds-tools-12-7` are installed.

Note

Neither `nvidia-gds-12-7` or `nvidia-fs-dkms` are needed.

```
sudo apt-get install cuda-toolkit
```

Chapter 6. Setting cufile.json Configuration

Add the following config parameter into the `/etc/cufile.json` or App-specific JSON file:

```
cufile.json
{
   "properties": {
      "use_pci_p2pdma": true
   }
}
```

6.1. Before You Test GDS

To install GDS on a non-DGX platform, complete the following steps:

1. Run the following command to check the current status of IOMMU.

```
$ dmesg | grep -i iommu
```

On x86_64 based platforms, if IOMMU is enabled, complete step 3 to disable it; otherwise continue to step 4.

1. Disable IOMMU
    1. Run the following command:

    ```
    $ sudo vi /etc/default/grub
    ```

    2. Add one of the following options to the `GRUB_CMDLINE_LINUX_DEFAULT` option:
        - If you have an AMD CPU, add `amd_iommu=off`.
        - If you have an Intel CPU, add `intel_iommu=off`.
    3. If there are already other options, enter a space to separate the options, for example, `GRUB_CMDLINE_LINUX_DEFAULT="console=tty0 amd_iommu=off"`
    4. In our experience, `iommu=off` works the best in terms of functionality and performance. On certain platforms such as DGX H100, DGX A100, `iommu=pt` is supported. `iommu=on` is not guaranteed to be either functional or performant.

```
 $ /usr/local/cuda/gds/tools/gdscheck -p

 GDS release version: 1.12.0.25
 libcufile version: 2.12
 Platform: x86_64
 ============
 ENVIRONMENT:
 ============
 =====================
 DRIVER CONFIGURATION:
 =====================
 NVMe P2PDMA        : Supported
 NVMe               : Unsupported
 NVMeOF             : Unsupported
 SCSI               : Unsupported
 ScaleFlux CSD      : Unsupported
 NVMesh             : Unsupported
 DDN EXAScaler      : Unsupported
 IBM Spectrum Scale : Unsupported
 NFS                : Unsupported
 BeeGFS             : Unsupported
 YanRong YRCloudFile: Unsupported
 WekaFS             : Unsupported
 Userspace RDMA     : Unsupported
 --Mellanox PeerDirect : Disabled
 --rdma library        : Not Loaded (libcufile_rdma.so)
 --rdma devices        : Not configured
 --rdma_device_status  : Up: 0 Down: 0
 =====================
 CUFILE CONFIGURATION:
 =====================
 properties.use_pci_p2pdma : true

 . . .
 =========
 GPU INFO:
 =========
 GPU index 0 NVIDIA L4 bar:1 bar size (MiB):32768 supports GDS, IOMMU State:
↪Disabled
==============
 PLATFORM INFO:
 ==============
 IOMMU: disabled
 Nvidia Driver Info Status: Supported(Nvidia Open Driver Installed)
 Cuda Driver Version Installed:  12070
 Platform: ProLiant DL360 Gen10, Arch: x86_64(Linux 6.2.0-31-generic)
 Platform verification succeeded
```

Note

Make sure ACS is disabled and IOMMU is `pt` or `off`.

6.2. Create Local Filesystem on NVMe

```
$ sudo mkfs.ext4 /dev/nvme1n1
```

6.3. Mount the NVME drive

```
$ mkdir -p  /mnt/nvme
$ sudo mount -o data=ordered /dev/nvme1n1 /mnt/nvme1
$ sudo chmod -R 777 /mnt/nvme
```

Chapter 7. Test GDS

Run a `gdsio` test with four threads writing and reading to four files in a directory `gdstest` under `/mnt/nvme` with GDS mode enabled.

```
$ sudo mkdir -p /mnt/nvme1n1/gdstest

NVME->HOST_MEMORY (using host memory similar to FIO)
# /usr/local/cuda/gds/tools/gdsio -D /mnt/nvme1n1/gdstest -d 0 -w 4 -x 1 -I 1
 ↪-s 1g -i 1m -V
IoType: WRITE XferType: CPUONLY Threads: 4 DataSetSize: 4156416/4194304(KiB)
 ↪IOSize: 1024(KiB) Throughput: 1.049312 GiB/sec, Avg_Latency: 3721.307952
 ↪usecs ops: 4059 total_time 3.777586 secs
Verifying data
IoType: READ XferType: CPUONLY Threads: 4 DataSetSize: 4069376/4194304(KiB)
 ↪IOSize: 1024(KiB) Throughput: 2.698050 GiB/sec, Avg_Latency: 1443.841342
 ↪usecs ops: 3974 total_time 1.438394 secs


# /usr/local/cuda/gds/tools/gdsio -D /mnt/nvme1n1/gdstest -d 0 -w 4 -x 1 -I 0
 ↪-s 1g -i 1m
IoType: READ XferType: CPUONLY Threads: 4 DataSetSize: 4187136/4194304(KiB)
 ↪IOSize: 1024(KiB) Throughput: 3.227660 GiB/sec, Avg_Latency: 1209.149901
 ↪usecs ops: 4089 total_time 1.237170 secs


NVME -> SYS MEM  -> HBM  (using host bounce buffer)

# /usr/local/cuda/gds/tools/gdsio -D /mnt/nvme1n1/gdstest -d 0 -w 4 -x 2 -I 1
 ↪-s 1g -i 1m -V
IoType: WRITE XferType: CPU_GPU Threads: 4 DataSetSize: 4191232/4194304(KiB)
 ↪IOSize: 1024(KiB) Throughput: 1.054915 GiB/sec, Avg_Latency: 3702.944992
 ↪usecs ops: 4093 total_time 3.788999 secs
Verifying data
IoType: READ XferType: CPU_GPU Threads: 4 DataSetSize: 4178944/4194304(KiB)
 ↪IOSize: 1024(KiB) Throughput: 3.135021 GiB/sec, Avg_Latency: 1244.475969
 ↪usecs ops: 4081 total_time 1.271236 secs

# /usr/local/cuda/gds/tools/gdsio -D /mnt/nvme1n1/gdstest -d 0 -w 4 -x 2 -I 0
 ↪-s 1g -i 1m
IoType: READ XferType: CPU_GPU Threads: 4 DataSetSize: 4188160/4194304(KiB)
 ↪IOSize: 1024(KiB) Throughput: 3.191077 GiB/sec, Avg_Latency: 1223.118288
 ↪usecs ops: 4090 total_time
```

(continues on next page)

(continued from previous page)

```
1.251659 secs


NVME -> HBM (GDS)
# /usr/local/cuda/gds/tools/gdsio -D /mnt/nvme1n1/gdstest -d 0 -w 4 -x 0 -I 1
 ↪-s 1g -i 1m -V
IoType: WRITE XferType: GPUD Threads: 4 DataSetSize: 4184064/4194304(KiB)
 ↪IOSize: 1024(KiB) Throughput: 1.048155 GiB/sec, Avg_Latency: 3726.404280
 ↪usecs ops: 4086 total_time 3.806913 secs
Verifying data
IoType: READ XferType: GPUD Threads: 4 DataSetSize: 4179968/4194304(KiB)
 ↪IOSize: 1024(KiB) Throughput: 3.184993 GiB/sec, Avg_Latency: 1224.346146
 ↪usecs ops: 4082 total_time 1.251597 secs

# /usr/local/cuda/gds/tools/gdsio -D /mnt/nvme1n1/gdstest -d 0 -w 4 -x 0 -I 0
 ↪-s 1g -i 1m
IoType: READ XferType: GPUD Threads: 4 DataSetSize: 4190208/4194304(KiB)
 ↪IOSize: 1024(KiB) Throughput: 3.229036 GiB/sec, Avg_Latency: 1208.627088
 ↪usecs ops: 4092 total_time 1.237550 secs
```

Chapter 8. Notices

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.

NVIDIA reserves the right to make corrections, modifications, enhancements, improvements, and any other changes to this document, at any time without notice.

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.

No license, either expressed or implied, is granted under any NVIDIA patent right, copyright, or other NVIDIA intellectual property right under this document. Information published by NVIDIA regarding third-party products or services does not constitute a license from NVIDIA to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property rights of the third party, or a license from NVIDIA under the patents or other intellectual property rights of NVIDIA.

8.1. Trademarks

Copyright