



HPC SDK Release Notes

Release 26.11ng-beta

NVIDIA Corporation

Nov 19, 2026

Contents

1 Porting to Beta NVFORTRAN **3**

2 Release Component Versions **5**

3 Supported Platforms **9**
3.1 Platform Requirements for the HPC SDK 9
3.2 Supported CUDA Toolchain Versions 10

4 Known Limitations and Recommendations **11**

5 Deprecations and Changes **13**

NVIDIA HPC SDK Release Notes

Welcome to version 26.11ng-beta of the NVIDIA HPC SDK, a beta release of compilers and libraries enabling developers to program the entire HPC platform, from the GPU foundation to the CPU and out through the interconnect.

The 26.11ng-beta release of the HPC SDK includes component updates as well as important functionality and performance improvements.

The NVC and NVC++ compilers are not based on Clang. They utilize the same proprietary compiler technology found in production, non-beta versions of the HPC SDK to ensure stable, high-performance C and C++ optimization.

NVFORTRAN is based on the LLVM Flang framework, leveraging the LLVM infrastructure for its Fortran front-end and code generation.

- HPC SDK 26.11ng-beta supports CUDA 13.x and CUDA 12.9. The 26.11ng-beta release packages include components from CUDA 13.4U1 and 12.9U1.
- CUDA 13.4, driver R615, changes how GPU memory is managed by default on coherent systems i.e. GH200, GB200, GB300, and Vera Rubin.
  - These systems support two modes. In NUMA mode, Linux treats GPU memory as part of system memory and can move data into it automatically. In CDMM mode, data stays in CPU memory and the GPU accesses it over the NVLink-C2C link. Until R615, NUMA was the default; starting with R615, CDMM is the default. A system runs in one mode or the other, set per node. For more information on how to check and change the default mode for a system, see blog post Understanding Memory Management on Hardware-Coherent Platforms
  - HPC applications work correctly in both modes. Performance will remain unchanged for OpenACC, OpenMP, and Stdpar applications compiled for Separate or Managed Memory Mode. There may be a performance degradation when running Unified Memory HPC applications in CDMM mode. For further details see Coherent Driver-based Memory Management (CDMM)
  - We recommend that HPC systems continue to use NUMA mode.
- Programs generated by the HPC Compilers for x86_64 processors require a minimum of AVX2 instructions, which includes Haswell and newer CPUs from Intel, as well as Excavator and newer CPUs from AMD. Target processors without AVX2 are no longer supported.
- Including `openacc_lib.h` in Fortran now emits a deprecation warning. The header was removed from the OpenACC specification in version 2.5. Use the `openacc` module instead.

Chapter 1. Porting to Beta NVFORTRAN

Beta NVFORTRAN is a new compiler built on LLVM Flang, and it differs from classic NVFORTRAN in its command-line options, language features, directives, preprocessor behavior, runtime conventions, and messages. Before you build existing applications with Beta NVFORTRAN, read the Porting Guide in the NVIDIA HPC Fortran Compiler User's Guide. It collects what the NVIDIA HPC applications team learned moving its applications, tests, and benchmarks to the new compiler, and explains how to send feedback.

Chapter 2. Release Component Versions

The NVIDIA HPC SDK 26.11ng-beta release contains the following versions of each component:

Table 1: HPC SDK Release Components

	Linux_x86_64		**Linux_aarch64**	
	CUDA 12.9U1	CUDA 13.4U1	CUDA 12.9U1	CUDA 13.4U1
nvc++	26.11ng-beta		26.11ng-beta	
nvc	26.11ng-beta		26.11ng-beta	
nvfortran	26.11ng-beta		26.11ng-beta	
nvcc	12.9.79	13.4.92	12.9.79	13.4.92
NCCL	2.32.3	2.32.3	2.32.3	2.32.3
NVSHMEM	3.8.0.0	3.8.0.0	3.8.0.0	3.8.0.0
cuBLAS	12.9.2.10	13.8.046	12.9.2.10	13.8.046
cuBLASMp	0.10.0	0.10.0(cc80+)	0.10.0	0.10.0(cc80+)
cuFFT	11.4.1.4	12.4.0.43	11.4.1.4	12.4.0.43
cuFFTMp*	11.4.0	12.1.3	11.4.0	12.1.3
cuRAND	10.3.10.19	10.4.4.72	10.3.10.19	10.4.4.72
cuSOLVER	11.7.5.82	12.3.5.3	11.7.5.82	12.3.5.3
cuSOLVERMp*	0.9.1.0	0.9.1.0 cc80+	0.9.1.0	0.9.1.0 cc80+
cuSPARSE	12.5.10.65	12.8.6.72	12.5.10.65	12.8.6.72
cuTENSOR	2.2.0 (cc70) 2.8.1 (>cc70)	2.8.1	2.2.0 (<cc70) 2.8.1 (>=cc70)	2.8.1
Nsight Compute	2025.2.1	2026.2.1	2025.2.1	2026.3.1
Nsight Systems	2025.3 (cc70) 2026.4.1	2026.4.1	2025.3 (cc70) 2026.5.1	2026.5.1
HPC-X	2.20 (12.0-12.2) 2.50 (12.3+)	2.50	2.20 (12.0-12.2) 2.50 (12.3+)	2.50
OpenBLAS	0.3.34		0.3.34	
ScaLAPACK	2.2.0		2.2.0	
Thrust	3.4.2	3.4.2	3.4.2	3.4.3
CUB	3.4.2	3.4.2	3.4.2	3.4.3
libcu++	2.8.2	3.4.2	2.8.2	3.4.3
NVPL	N/A		26.10	

* product in beta

Chapter 3. Supported Platforms

3.1. Platform Requirements for the HPC SDK

Table 2: HPC SDK Platform Requirements

Architecture	Linux Distributions	Minimum gcc/glibc Toolchain	Minimum CUDA Driver
x86_64	RHEL/CentOS/Rocky 8.0 - 8.10 RHEL/Rocky 9.2 - 9.8, 10.0 - 10.2 OpenSUSE Leap 15.6 - 16.0 SLES 15SP6, 15SP7, 16.0 Ubuntu 22.04, 24.04, 26.04 Debian 12, 13 Fedora 41 - 44 CentOS Stream 9, 10	Fortran, C, and up to C++17: 7.5 C++20: 10.1 C++23: 12.1	12.x: >=525.60.13 13.x: >=580.65.06
aarch64	RHEL/CentOS/Rocky 8.0 - 8.10 Rocky 9.2 - 9.6, 10.0 - 10.1 Ubuntu 22.04, 24.04, 26.04 SLES 15SP6, 15SP7, 16.0 Amazon Linux 2023	Fortran, C, and up to C++17: 7.5 C++20: 10.1 C++23: 12.1	12.x: >=525.60.13 13.x: >=580.65.06

Programs generated by the HPC Compilers for x86_64 processors require a minimum of AVX2 instructions, which includes Haswell and newer CPUs from Intel, as well as Excavator and newer CPUs from

AMD. The HPC SDK includes support for v8.1+ Server Class Arm CPUs that meet the requirements in appendix E specified in the SBSA 7.1 specification.

The HPC Compilers are compatible with gcc and g++ and use the GCC C and C++ libraries; the minimum compatible versions of GCC are listed in the table in Section 3.1. The minimum system requirements for CUDA and NVIDIA Math Library requirements are available in the NVIDIA CUDA Toolkit documentation.

3.2. Supported CUDA Toolchain Versions

The NVIDIA HPC SDK uses elements of the CUDA toolchain when building programs for execution with NVIDIA GPUs. Every HPC SDK installation package puts the required CUDA components into an installation directory called [install-prefix]/[arch]/[nvhpc-version]/cuda.

An NVIDIA CUDA GPU device driver must be installed on a system with a GPU before you can run a program compiled for the GPU on that system. The NVIDIA HPC SDK does not contain CUDA drivers. You must download and install the appropriate CUDA driver from NVIDIA , including the CUDA Compatibility Platform if that is required.

The `nvaccelinfo` command prints the CUDA Driver version in its output. You can use it to find out which version of the CUDA Driver is installed on your system.

The NVIDIA HPC SDK 26.11ng-beta includes the following CUDA toolchain versions:

- CUDA 12.9U1
- CUDA 13.4U1

The minimum required CUDA driver versions are listed in the table in Section 3.1.

Chapter 4. Known Limitations and Recommendations

The following are recommendations for more effectively using the HPC SDK and its components when unexpected behavior or suboptimal performance is encountered.

- Communication libraries (HPC-X MPI, OpenSHMEM, UCX, ...)
  - HPC SDK 26.11ng-beta defaults to using HPC-X version 2.50 which is incompatible with the CUDA 12.0 - 12.3 driver (R545). HPC-X 2.20 is available as a fallback for users requiring CUDA 12.0 - 12.3. HPC-X 2.20 can be selected by loading the `nvhpc-hpcx-2.20-cuda12` environment module.
  - Starting with Open MPI v5.0 in HPC-X 2.50, the default runtime environment has transitioned from the Open Run-Time Environment (ORTE) to the PMIx Reference RunTime Environment (PRRTE).
    - Parallel job launch, monitoring, and both processor and memory affinity management are now fully handled by PRRTE.
    - Along with this runtime change, the default process affinity mapping has shifted from map-by-socket to map-by-core under PRRTE.
    - For more information on processor and memory affinity, see Open MPI Affinity Guide
    - For more information on the role of PMIx and PRRTE, see: Open MPI v5.0.x Launching Applications Guide
  - Starting with version 2.24, HPC-X requires the installed GDRCopy driver to be at least of version 2.5. This is only required when the application needs to use the GDRCopy transport.
  - HPC-X MPI initialization time on systems with CUDA may be higher than on systems without CUDA installed. If your application does not use GPU communication, you may be able to reduce the initialization overhead by setting the MPI environment variables `OMPI_MCA_coll_ucc_enable=0` and `UCX_MODULES=^cuda`. Please be aware that disabling UCC may degrade performance in other areas of HPC-X MPI, so we recommend testing overall performance changes with these settings.
  - Both NVSHMEM and NCCL rely on GPUDirect RDMA for direct GPU-to-GPU communication within a node. To achieve the best performance on bare metal Linux platforms, the GPUDirect Storage Best Practice Guide recommends that system settings like PCIe Access Control Services (ACS) and Input-Output Memory Management Units (IOMMUs) be disabled or set to passthrough mode. The NCCL documentation also suggests that ACS and IOMMUs be disabled, citing that they could cause a significant performance reduction or even a hang.
  - Any program data specified in `acc declare create` (and related clauses such as `copyin`, `device_resident`) can cause an application crash if used in an HPC-X MPI transport.

- The MPI wrappers in `comm_libs/mpi/bin` automatically detect the CUDA driver and select the matching MPI library from `comm_libs/X.Y`. Applications that require a full MPI directory hierarchy (e.g., bin, include, lib) or are launched via srun should bypass the MPI wrappers by loading the `nvhpc-hpcx-cuda13` or the `nvhpc-hpcx-cuda12` environment module, depending on the installed CUDA driver version.
- To use HPC-X, please use the provided environment module files or take care to source the hpcx-init.sh script: `$ . ${NVHPCSDK_HOME}/comm_libs/X.Y/hpcx/latest/hpcx-init.sh` Then, run the hpcx_load function defined by this script: `hpcx_load`. These actions will set important environment variables that are needed when running HPC-X.
- The following warning from HPC-X may be encountered due to oversubscription, failure to detect proper topology, etc., while running an MPI job – "WARNING: Open MPI tried to bind a process but failed. This is a warning only; your job will continue, though performance may be degraded". This may be suppressed as follows: `export OMPI_MCA_hwloc_base_binding_policy=""`
- Starting with version 2.17.1, HPC-X does not have performance-optimal support for stream-ordered CUDA-allocated memory. In practical terms it means that IPC methods such as the MPI calls `MPI_Send` and `MPI_Recv` can have significantly degraded throughput when passed data allocated with the `cudaMallocAsync` function or its variants. This limitation will be removed in a future release.

- Math Libraries
  - Known issues related to NVPL are described in the NVPL documentation.

Chapter 5. Deprecations and Changes

Notices

Notice and Disclaimers

All information provided in this document is provided as-is, for your informational purposes only and is subject to change at any time without notice. Reproduction of information in this document is permissible only if approved in advance by NVIDIA in writing. To obtain the latest information, please contact your NVIDIA representative. Product or service performance varies by use, configuration and other factors. Your costs and results may vary. No product or component is absolutely secure. TO THE FULLEST EXTENT PERMITTED BY APPLICABLE LAW, NVIDIA DISCLAIMS ALL WARRANTIES AND REPRESENTATIONS OF ANY KIND, WHETHER EXPRESS, IMPLIED OR STATUTORY, RELATING TO OR ARISING UNDER THIS DOCUMENT, INCLUDING, WITHOUT LIMITATION, THE WARRANTIES OF TITLE, NONINFRINGEMENT, MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, USAGE OF TRADE AND COURSE OF DEALING. NVIDIA products are not intended or authorized for use as critical components in a system or application where the use of or failure of such system or application developed with products, technology, software or services provided by NVIDIA could result in injury, death or catastrophic damage.

Except for your permitted use of the information contained in this document, no license or right is granted by implication, estoppel or otherwise. If this document directly includes or links to third-party websites, products, services or information, please consult other sources to evaluate if and how to use that information since NVIDIA does not support, endorse or assume any responsibility for any third party offerings or its accuracy or usefulness.

TO THE FULLEST EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT WILL NVIDIA BE LIABLE FOR ANY (I) INDIRECT, PUNITIVE, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, OR (II) DAMAGES FOR THE (A) COST OF PROCURING SUBSTITUTE GOODS OR (B) LOSS OF PROFITS, REVENUES, USE, DATA OR GOODWILL ARISING OUT OF OR RELATED TO THIS DOCUMENT, WHETHER BASED ON BREACH OF CONTRACT, TORT (INCLUDING NEGLIGENCE), STRICT LIABILITY, OR OTHERWISE, AND EVEN IF NVIDIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES AND EVEN IF A PARTY'S REMEDIES FAIL THEIR ESSENTIAL PURPOSE. ADDITIONALLY, TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, NVIDIA'S TOTAL CUMULATIVE AGGREGATE LIABILITY FOR ANY AND ALL LIABILITIES, OBLIGATIONS OR CLAIMS ARISING OUT OF OR RELATED TO THIS DOCUMENT WILL NOT EXCEED FIVE U.S. DOLLARS (US$5).

Statements in this document that refer to future plans or expectations are forward-looking statements. These statements are based on currently available information, beliefs, assumptions and involve many risks and uncertainties that could cause actual results to differ materially from those expressed or implied in these statements. For more information on the factors that could cause actual results to differ materially, see our most recent earnings release and SEC filings at NVIDIA Corporation SEC Filings.

© NVIDIA Corporation. All rights reserved. NVIDIA, the NVIDIA logo, and other NVIDIA marks are trademarks of NVIDIA Corporation or its affiliates. Other names and brands may be claimed as the property of others.

Trademarks

NVIDIA, the NVIDIA logo, CUDA, CUDA-X, GPUDirect, HPC SDK, NGC, NVIDIA Volta, NVIDIA DGX, NVIDIA Nsight, NVLink, NVSwitch, and Tesla are trademarks and/or registered trademarks of NVIDIA Corporation in the U.S. and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright

©2022-2026, NVIDIA Corporation & affiliates. All rights reserved