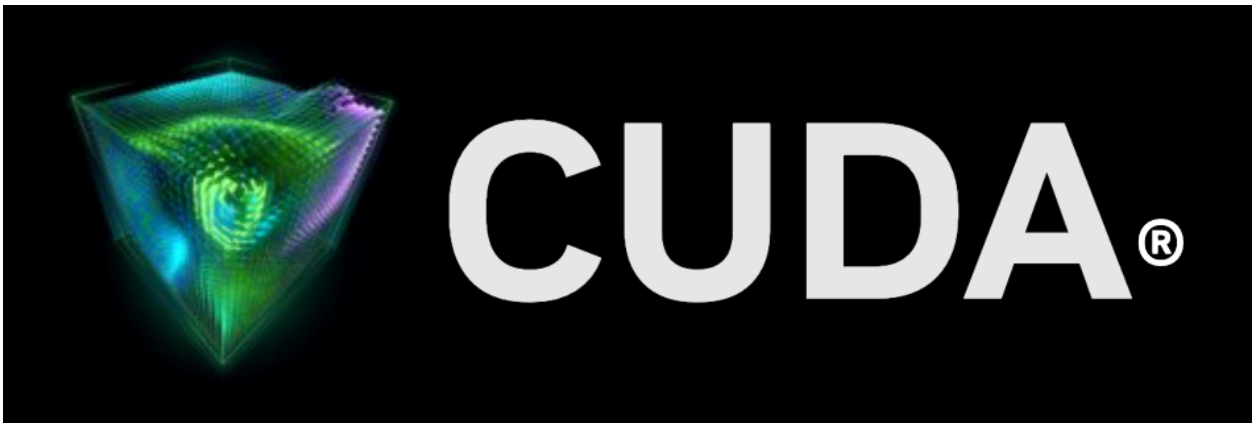


CUBLAS Release Notes

Release 13.4

NVIDIA Corporation

Oct 29, 2026

Contents

1 Overview **1**

2 cuBLAS: Patch Release 13.8.2 **3**

3 cuBLAS: Patch Release 13.8.1 **5**

4 cuBLAS: Patch Release 13.6.3 **7**

5 cuBLAS: Patch Release 13.6.2 **9**

6 cuBLAS: Patch Release 13.6.1 **11**

7 cuBLAS: Patch Release 13.4.3 **13**

8 cuBLAS: Patch Release 13.4.2 **15**

9 cuBLAS: Patch Release 13.4.1 **17**

10 cuBLAS: Patch Release 13.2.2 **19**

11 cuBLAS: Patch Release 13.1.2 **21**

12 cuBLAS: Patch Release 13.1.1 **23**

13 cuBLAS: Patch Release 12.9.3 **25**

14 cuBLAS: Patch Release 12.9.2 **27**

15 cuBLAS: Patch Release 12.8.5 **29**

16 Notices **31**
16.1 Notice 31
16.2 OpenCL 32
16.3 Trademarks 32

Chapter 1. Overview

cuBLAS Patches - Release Notes

This section lists cuBLAS patch releases that were issued separately from CUDA Toolkit major and minor releases. Each entry describes the resolved issues included in that patch release.

Chapter 2. cuBLAS: Patch Release 13.8.2

Applies to CUDA Toolkit 13.4 Update 1

- **Resolved Issues**
  - Fixed an issue where, under concurrent load on the GPU, GEMM kernels with `beta` not equal to `0` could produce incorrect results on Blackwell, Blackwell Ultra, and Rubin GPUs. This affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66` and a cluster of `8` or more CTAs. In testing, the issue was reproducible only for configurations returned by heuristics for NVFP4, but problematic algorithm configurations could be constructed manually for all precisions. Introduced in CUDA Toolkit 12.8 (cuBLAS 12.8.3). [*6854729*]
  - Fixed an issue with out-of-place matmul support for fixed-point emulation where the resulting D matrix could be incorrect if the leading dimensions of the C and D matrices differed or split-K was used. This affected Ampere and newer GPUs. Introduced in CUDA Toolkit 13.0 Update 2. [*CUB-10744*]
  - Fixed an issue where Ozaki-II double-complex BLAS3 routines could produce incorrect results in rare cases. This could happen for elements produced by dot products of nearly collinear vectors where the components of each vector are roughly the same magnitude. This affected Ampere and newer GPUs. Introduced in CUDA Toolkit 13.4. [*CUB-10730*]
  - Fixed an issue where Ozaki-I fixed-point emulation (FP64 and CFP64) could produce incorrect results or data corruption on B200 and GB200 Blackwell GPUs when run concurrently with third-party kernels that also allocate Tensor Memory. Introduced in CUDA Toolkit 13.0 Update 2. [*CUB-9633*]

Chapter 3. cuBLAS: Patch Release 13.8.1

Applies to CUDA Toolkit 13.4 Update 1

- **Resolved Issues**
  - Fixed an issue where cuBLASLt Grouped GEMM operations could produce incorrect results on Blackwell and Rubin GPUs. Introduced in CUDA Toolkit 13.4 (cuBLAS 13.7.0). [*6798616*]

Chapter 4. cuBLAS: Patch Release 13.6.3

Applies to CUDA Toolkit 13.3 Update 1

- **Resolved Issues**
  - Fixed an issue where, under concurrent load on the GPU, GEMM kernels with `beta` not equal to `0` could produce incorrect results on Blackwell, Blackwell Ultra, and Rubin GPUs. This affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66` and a cluster of 8 or more CTAs. In testing, the issue was reproducible only for configurations returned by heuristics for NVFP4, but problematic algorithm configurations could be constructed manually for all precisions. Introduced in CUDA Toolkit 12.8 (cuBLAS 12.8.3). [*6854729*]
  - Fixed an issue with out-of-place matmul support for fixed-point emulation where the resulting D matrix could be incorrect if the leading dimensions of the C and D matrices differed or split-K was used. This affected Ampere and newer GPUs. Introduced in CUDA Toolkit 13.0 Update 2. [*CUB-10744*]
  - Fixed an issue where Ozaki-I fixed-point emulation (FP64 and CFP64) could produce incorrect results or data corruption on B200 and GB200 Blackwell GPUs when run concurrently with third-party kernels that also allocate Tensor Memory. Introduced in CUDA Toolkit 13.0 Update 2. [*CUB-9633*]
  - Fixed an issue where Ozaki-I fixed-point emulation kernels could deadlock on B200 and GB200 Blackwell GPUs. Introduced in CUDA Toolkit 13.0 Update 2. [*CUB-10823*]
  - Fixed an issue where `cublasLtMatmul()` could return incorrect output when using split-K if the number of K splits (`CUBLASLT_ALGO_CONFIG_SPLITK_NUM`) did not evenly divide the K dimension and the floor division of K by the number of splits was an exact multiple of the stage size. This affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66` on GPUs with compute capability 9.0, 10.x, and 11.x. Introduced in CUDA Toolkit 12.6 Update 2 (cuBLAS 12.6.3). [*6580689*]
  - Improved performance for TF32 strided-batched problems on H20 GPUs. [*6432787*]

Chapter 5. cuBLAS: Patch Release 13.6.2

Applies to CUDA Toolkit 13.3 Update 1

- **Resolved Issues**
  - Fixed an issue where cuBLASLt Grouped GEMM operations with `CUBLAS_POINTER_MODE_HOST` could produce incorrect results on Hopper GPUs when the number of waves was larger than 2. Introduced in CUDA Toolkit 13.2 Update 1 (cuBLAS 13.4.0). [*6681084*]
  - Fixed an issue where `cublasLtMatmul()` could produce incorrect results on B300 and Rubin GPUs when the input matrices used the NVFP4 data type. This affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66` and `CUBLASLT_ALGO_CONFIG_STAGES_ID` equal to `CUBLASLT_MATMUL_STAGES_768xAUTO`. [*CUB-10573*]

Chapter 6. cuBLAS: Patch Release 13.6.1

Applies to CUDA Toolkit 13.3 Update 1

- **Resolved Issues**
  - Fixed an issue where the cuBLASLt heuristics for the Grouped GEMM API returned sub-optimal algorithms when the C and D matrices had ordering `CUBLASLT_ORDER_ROW`. Introduced in CUDA Toolkit 13.3 Update 1 (cuBLAS 13.6.0). [*6335555*]
  - Fixed an issue where executing multiple cuBLASLt Grouped GEMM operations that reused workspace on Hopper could lead to hangs or unspecified launch failures. Introduced in CUDA Toolkit 13.2 Update 1 (cuBLAS 13.4.0). [*6456362*]
  - Fixed an issue where using `cublasLtMatmulAlgoGetHeuristic()` while a non-default blocking stream was being captured resulted in `CUDNN_STATUS_INTERNAL_ERROR`. Introduced in CUDA Toolkit 13.3 (cuBLAS 13.5.1). [*6288786*]
  - Fixed an issue where GEMV-like operations (`cublas<t>gemv()`, `cublasLtMatmul()` with M or N equal to 1, and so on) could return `CUBLAS_STATUS_NOT_SUPPORTED` for certain shapes and workspace configurations. Introduced in CUDA Toolkit 13.3 (cuBLAS 13.5.1). [*6435705, 6418535*]
  - Fixed an issue where strided batched GEMM operations using broadcast operands could perform out-of-bounds memory reads on sm120 and sm121 GPUs. This occurred when an input matrix shared across batches (through zero or overlapping strides) was smaller than a few hundred KB. While numerical accuracy remained unaffected, these invalid accesses could trigger compute-sanitizer warnings or, in rare instances, result in illegal memory access errors. Introduced in CUDA Toolkit 13.0 Update 2 (cuBLAS 13.1.0). [*6040940, 5996751*]
  - Fixed an issue where cuBLAS GEMM kernels on sm10x and sm12x might access device alpha and beta pointers before calling `cudaGridDependencySynchronize()`. This could result in a WAR hazard if the preceding PDL kernel produced alpha and beta values on the device after calling `cudaTriggerProgrammaticLaunchCompletion()`. Introduced in CUDA Toolkit 13.0 Update 2 (cuBLAS 13.1.0). [*CUB-10409*]
  - Fixed an issue where GEMV-like operations (`cublas<t>gemv()`, `cublasLtMatmul()` with M or N equal to 1, and so on) could return incorrect results on sm120 and sm121 GPUs. This occurred when the matrix was transposed, its non-accumulation dimension was larger than 3145680, and `beta` was not 0. Introduced in CUDA Toolkit 13.3 Update 1 (cuBLAS 13.6.0). [*CUB-10445*]

Chapter 7. cuBLAS: Patch Release 13.4.3

Applies to CUDA Toolkit 13.2 Update 2

- **Resolved Issues**
  - Fixed an issue where, under concurrent load on the GPU, GEMM kernels with `beta` not equal to `0` could produce incorrect results on Blackwell, Blackwell Ultra, and Rubin GPUs. This affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66` and a cluster of 8 or more CTAs. In testing, the issue was reproducible only for configurations returned by heuristics for NVFP4, but problematic algorithm configurations could be constructed manually for all precisions. Introduced in CUDA Toolkit 12.8 (cuBLAS 12.8.3). [*6854729*]
  - Fixed an issue with out-of-place matmul support for fixed-point emulation where the resulting D matrix could be incorrect if the leading dimensions of the C and D matrices differed or split-K was used. This affected Ampere and newer GPUs. Introduced in CUDA Toolkit 13.0 Update 2. [*CUB-10744*]
  - Fixed an issue where Ozaki-I fixed-point emulation (FP64 and CFP64) could produce incorrect results or data corruption on B200 and GB200 Blackwell GPUs when run concurrently with third-party kernels that also allocate Tensor Memory. Introduced in CUDA Toolkit 13.0 Update 2. [*CUB-9633*]
  - Fixed an issue where Ozaki-I fixed-point emulation kernels could deadlock on B200 and GB200 Blackwell GPUs. Introduced in CUDA Toolkit 13.0 Update 2. [*CUB-10823*]
  - Fixed an issue where `cublasLtMatmul()` could return incorrect output when using split-K if the number of K splits (`CUBLASLT_ALGO_CONFIG_SPLITK_NUM`) did not evenly divide the K dimension and the floor division of K by the number of splits was an exact multiple of the stage size. This affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66` on GPUs with compute capability 9.0, 10.x, and 11.x. Introduced in CUDA Toolkit 12.6 Update 2 (cuBLAS 12.6.3). [*6580689*]
  - Improved performance for TF32 strided-batched problems on H20 GPUs. [*6432787*]

Chapter 8. cuBLAS: Patch Release 13.4.2

Applies to CUDA Toolkit 13.2 Update 2

- **Resolved Issues**
  - Fixed an issue where cuBLASLt Grouped GEMM operations with `CUBLAS_POINTER_MODE_HOST` could produce incorrect results on Hopper GPUs when the number of waves was larger than 2. Introduced in CUDA Toolkit 13.2 Update 1 (cuBLAS 13.4.0). [*6681084*]

Chapter 9. cuBLAS: Patch Release 13.4.1

Applies to CUDA Toolkit 13.2 Update 1

- **Resolved Issues**
  - Fixed an issue where `cublasLtMatmul()` ignored the tensor-wide scaling value specified by `CUBLASLT_MATMUL_DESC_D_SCALE_POINTER` for NVFP4 matrix multiplications with NVFP4 output, resulting in incorrect results. This affected algorithms with `CUBLASLT_ALGO_CONFIG_ID = 66` on GPUs with compute capability 10.x and 11.x. Introduced in CUDA Toolkit 13.2 Update 1 (cuBLAS 13.4.0). [6059292]

Chapter 10. cuBLAS: Patch Release 13.2.2

Applies to CUDA Toolkit 13.1 Update 1

- **Resolved Issues**
  - Fixed an issue in `cublasLtMatmul()` that could lead to incorrect results when it ran concurrently with another kernel that used Tensor Memory. This issue affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66` on GPUs with Compute Capability 10.x and 11.x, and had existed since cuBLAS 12.8. [*5807900, 5943783*]
  - Fixed an issue in `cublasLtMatmul()` that could lead to incorrect results or invalid memory access errors for large leading dimensions, specifically when the product of the data type size and the leading dimension of a matrix exceeded the bounds of a signed 32-bit integer. This issue affected GPUs with Compute Capability 9.0, 10.x, and 11.0, had existed since cuBLAS 12.6 Update 2, and affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66`. [*CUB-9572*]
  - Fixed an issue in the cuBLASLt Matmul API that caused FP8 kernels to hang on GPUs with Compute Capability 9.0 when `beta != 0` and `scale_C = 0`. This issue affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66`. [*CUB-9627*]
  - Fixed an issue where some `ztrmm` kernels produced incorrect results when `m = 1` and `side = r` on NVIDIA Ada and Blackwell GeForce-class GPUs. [*5452663*]

Chapter 11. cuBLAS: Patch Release 13.1.2

Applies to CUDA Toolkit 13.0 Update 3

- **Resolved Issues**
  - Fixed an issue where, under concurrent load on the GPU, GEMM kernels with `beta` not equal to `0` could produce incorrect results on Blackwell, Blackwell Ultra, and Rubin GPUs. This affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66` and a cluster of 8 or more CTAs. In testing, the issue was reproducible only for configurations returned by heuristics for NVFP4, but problematic algorithm configurations could be constructed manually for all precisions. Introduced in CUDA Toolkit 12.8 (cuBLAS 12.8.3). [*6854729*]
  - Fixed an issue where, under concurrent load on the GPU, GEMM kernels with `beta` not equal to `0` could produce incorrect results or fail with a launch error on Hopper GPUs. This affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66` and a cluster with more than one CTA in each dimension (for example, 2x2, 4x2, and 2x4). Such configurations can be returned by heuristics for FP8, FP16, BF16, and TF32. This affected cuBLAS versions from CUDA Toolkit 12.8 (cuBLAS 12.8.3) through CUDA Toolkit 13.2 (cuBLAS 13.3.0). [*CUB-9496*]
  - Fixed an issue with out-of-place matmul support for fixed-point emulation where the resulting D matrix could be incorrect if the leading dimensions of the C and D matrices differed or split-K was used. This affected Ampere and newer GPUs. Introduced in CUDA Toolkit 13.0 Update 2. [*CUB-10744*]
  - Fixed an issue where Ozaki-I fixed-point emulation (FP64 and CFP64) could produce incorrect results or data corruption on B200 and GB200 Blackwell GPUs when run concurrently with third-party kernels that also allocate Tensor Memory. Introduced in CUDA Toolkit 13.0 Update 2. [*CUB-9633*]
  - Fixed an issue where Ozaki-I fixed-point emulation kernels could deadlock on B200 and GB200 Blackwell GPUs. Introduced in CUDA Toolkit 13.0 Update 2. [*CUB-10823*]
  - Fixed an integer-overflow issue in complex emulated FP64 matrix multiplication. The affected routines include `cublasZgemm()`, `cublasZtrsm()`, `cublasGemmEx()`, and `cublasLtMatmul()`. The overflow occurred when `2*m*n+m` exceeded `UINT_MAX`, where `m` is the number of rows of op(A) and C, and `n` is the number of columns of op(B) and C. [*5720478*]
  - Fixed an issue where `cublasLtMatmul()` could return incorrect output when using split-K if the number of K splits (`CUBLASLT_ALGO_CONFIG_SPLITK_NUM`) did not evenly divide the K dimension and the floor division of K by the number of splits was an exact multiple of the stage size. This affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66` on GPUs with compute capability 9.0, 10.x, and 11.x. Introduced in CUDA Toolkit 12.6 Update 2 (cuBLAS 12.6.3). [*6580689*]

- Improved performance for TF32 strided-batched problems on H20 GPUs. [*6432787*]

Chapter 12. cuBLAS: Patch Release 13.1.1

Applies to CUDA Toolkit 13.0 Update 2

- **Resolved Issues**
  - Fixed an issue in `cublasLtMatmul()` that could lead to incorrect results when it ran concurrently with another kernel that used Tensor Memory. This issue affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66` on GPUs with Compute Capability 10.x and 11.x, and had existed since cuBLAS 12.8. [*5807900, 5943783*]
  - Fixed an issue in `cublasLtMatmul()` that could lead to incorrect results or invalid memory access errors for large leading dimensions, specifically when the product of the data type size and the leading dimension of a matrix exceeded the bounds of a signed 32-bit integer. This issue affected GPUs with Compute Capability 9.0, 10.x, and 11.0, had existed since cuBLAS 12.6 Update 2, and affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66`. [*CUB-9572*]
  - Fixed an issue in the cuBLASLt Matmul API that caused FP8 kernels to hang on GPUs with Compute Capability 9.0 when `beta != 0` and `scale_C = 0`. This issue affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66`. [*CUB-9627*]
  - Fixed an issue where some `ztrmm` kernels produced incorrect results when `m = 1` and `side = r` on NVIDIA Ada and Blackwell GeForce-class GPUs. [*5452663*]

Chapter 13. cuBLAS: Patch Release 12.9.3

Applies to CUDA Toolkit 12.9 Update 2

- **Resolved Issues**
  - Fixed an issue where, under concurrent load on the GPU, GEMM kernels with `beta` not equal to `0` could produce incorrect results on Blackwell, Blackwell Ultra, and Rubin GPUs. This affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66` and a cluster of `8` or more CTAs. In testing, the issue was reproducible only for configurations returned by heuristics for NVFP4, but problematic algorithm configurations could be constructed manually for all precisions. Introduced in CUDA Toolkit 12.8 (cuBLAS 12.8.3). [*6854729*]
  - Fixed an issue where, under concurrent load on the GPU, GEMM kernels with `beta` not equal to `0` could produce incorrect results or fail with a launch error on Hopper GPUs. This affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66` and a cluster with more than one CTA in each dimension (for example, 2x2, 4x2, and 2x4). Such configurations can be returned by heuristics for FP8, FP16, BF16, and TF32. This affected cuBLAS versions from CUDA Toolkit 12.8 (cuBLAS 12.8.3) through CUDA Toolkit 13.2 (cuBLAS 13.3.0). [*CUB-9496*]
  - Fixed an issue where `cublasLtMatmul()` could return incorrect output when using split-K if the number of K splits (`CUBLASLT_ALGO_CONFIG_SPLITK_NUM`) did not evenly divide the K dimension and the floor division of K by the number of splits was an exact multiple of the stage size. This affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66` on GPUs with compute capability 9.0, 10.x, and 11.x. Introduced in CUDA Toolkit 12.6 Update 2 (cuBLAS 12.6.3). [*6580689*]
  - Improved performance for TF32 strided-batched problems on H20 GPUs. [*6432787*]

Chapter 14. cuBLAS: Patch Release 12.9.2

Applies to CUDA Toolkit 12.9 Update 1

- **Resolved Issues**
  - Fixed an issue in `cublasLtMatmul()` that could lead to incorrect results when it ran concurrently with another kernel that used Tensor Memory. This issue affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66` on GPUs with Compute Capability 10.x and 11.x, and had existed since cuBLAS 12.8. [*5807900, 5943783*]
  - Fixed an issue in `cublasLtMatmul()` that could lead to incorrect results or invalid memory access errors for large leading dimensions, specifically when the product of the data type size and the leading dimension of a matrix exceeded the bounds of a signed 32-bit integer. This issue affected GPUs with Compute Capability 9.0, 10.x, and 11.0, had existed since cuBLAS 12.6 Update 2, and affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66`. [*CUB-9572*]
  - Fixed an issue in the cuBLASLt Matmul API that caused FP8 kernels to hang on GPUs with Compute Capability 9.0 when `beta != 0` and `scale_C = 0`. This issue affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66`. [*CUB-9627*]
  - Fixed an issue where some `ztrmm` kernels produced incorrect results when `m = 1` and `side = r` on NVIDIA Ada and Blackwell GeForce-class GPUs. [*5452663*]

Chapter 15. cuBLAS: Patch Release 12.8.5

Applies to CUDA Toolkit 12.8 Update 1

- **Resolved Issues**
  - Fixed an issue in `cublasLtMatmul()` that could lead to incorrect results when it ran concurrently with another kernel that used Tensor Memory. This issue affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66` on GPUs with Compute Capability 10.x and 11.x, and had existed since cuBLAS 12.8. [*5807900, 5943783*]
  - Fixed an issue in `cublasLtMatmul()` that could lead to incorrect results or invalid memory access errors for large leading dimensions, specifically when the product of the data type size and the leading dimension of a matrix exceeded the bounds of a signed 32-bit integer. This issue affected GPUs with Compute Capability 9.0, 10.x, and 11.0, had existed since cuBLAS 12.6 Update 2, and affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66`. [*CUB-9572*]
  - Fixed an issue in the cuBLASLt Matmul API that caused FP8 kernels to hang on GPUs with Compute Capability 9.0 when `beta != 0` and `scale_C = 0`. This issue affected only algorithms with `CUBLASLT_ALGO_CONFIG_ID` equal to `66`. [*CUB-9627*]
  - Fixed an issue where some `ztrmm` kernels produced incorrect results when `m = 1` and `side = r` on NVIDIA Ada and Blackwell GeForce-class GPUs. [*5452663*]

Chapter 16. Notices

16.1. 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.

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.

Reproduction of information in this document is permissible only if approved in advance by NVIDIA in writing, reproduced without alteration and in full compliance with all applicable export laws and regulations, and accompanied by all associated conditions, limitations, and notices.

THIS DOCUMENT AND ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL NVIDIA BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF NVIDIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Notwithstanding any damages that customer might incur for any reason whatsoever, NVIDIA's aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms of Sale for the product.

16.2. OpenCL

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

16.3. Trademarks

NVIDIA and the NVIDIA logo are trademarks 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

©2007-2026, NVIDIA Corporation & affiliates. All rights reserved