NVIDIA CUDA Toolkit

Release Notes for CUDA 11.1

RN-06722-001 _v11.1 | September 2020

Table of Contents

List of Tables

Chapter 1. CUDA 11.1 Release Notes

The release notes for the CUDA Toolkit can be found online at http://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html.

1.1. CUDA Toolkit Major Component Versions

CUDA Components

Starting with CUDA 11, the various components in the toolkit are versioned independently.

For CUDA 11.1, the table below indicates the versions:

Table 1. CUDA 11.1 Component Versions

Component Name	Version Information	Supported Architectures
CUDA Runtime (cudart)	11.1.74	x86_64, POWER, Arm64
cuobjdump	11.1.74	x86_64, POWER, Arm64
CUPTI	11.1.69	x86_64, POWER, Arm64
CUDA Demo Suite	11.1.74	x86_64
CUDA GDB	11.1.69	x86_64, POWER, Arm64
CUDA Memcheck	11.1.69	x86_64, POWER
CUDA NVCC	11.1.74	x86_64, POWER, Arm64
CUDA nvdisasm	11.1.74	x86_64, POWER, Arm64
CUDA NVML Headers	11.1.74	x86_64, POWER, Arm64
CUDA nvprof	11.1.69	x86_64, POWER, Arm64
CUDA nvprune	11.1.74	x86_64, POWER, Arm64
CUDA NVRTC	11.1.74	x86_64, POWER, Arm64
CUDA NVTX	11.1.74	x86_64, POWER, Arm64
CUDA NVVP	11.1.74	x86_64, POWER
CUDA Samples	11.1.74	x86_64, POWER, Arm64
CUDA Compute Sanitizer API	11.1.49	x86_64, POWER, Arm64

Component Name	Version Information	Supported Architectures
CUDA cuBLAS	11.2.1.74	x86_64, POWER, Arm64
CUDA cuFFT	10.3.0.74	x86_64, POWER, Arm64
CUDA cuRAND	10.2.2.74	x86_64, POWER, Arm64
CUDA cuSOLVER	11.0.0.74	x86_64, POWER, Arm64
CUDA cuSPARSE	11.2.0.275	x86_64, POWER, Arm64
CUDA NPP	11.1.1.269	x86_64, POWER, Arm64
CUDA nvJPEG	11.2.0.74	x86_64, POWER, Arm64
Nsight Eclipse Plugins	11.1.74	x86_64, POWER
Nsight Compute	2020.2.0.18	x86_64, POWER, Arm64
Nsight Windows NVTX	1.21018621	x86_64, POWER, Arm64
Nsight Systems	2020.3.4.32	x86_64, POWER, Arm64
Nsight Visual Studio Edition (VSE)	2020.2.0.20225	x86_64 (Windows)
NVIDIA Linux Driver	455.23	x86_64, POWER, Arm64
NVIDIA Windows Driver	456.38	x86_64 (Windows)

CUDA Driver

Running a CUDA application requires the system with at least one CUDA capable GPU and a driver that is compatible with the CUDA Toolkit. See Table 2. For more information various GPU products that are CUDA capable, visit https://developer.nvidia.com/cuda-gpus.

Each release of the CUDA Toolkit requires a minimum version of the CUDA driver. The CUDA driver is backward compatible, meaning that applications compiled against a particular version of the CUDA will continue to work on subsequent (later) driver releases.

More information on compatibility can be found at https://docs.nvidia.com/cuda/cuda-c-best-practices-guide/index.html#cuda-runtime-and-driver-api-version.

Note: Starting with CUDA 11.0, the toolkit components are individually versioned, and the toolkit itself is versioned as shown in the table below.

Table 2. CUDA Toolkit and Compatible Driver Versions

CUDA Toolkit	Linux x86_64 Driver Version	Windows x86_64 Driver Version
CUDA 11.1	>=455.23	>=456.38
CUDA 11.0.3 Update 1	>= 450.51.06	>= 451.82
CUDA 11.0.2 GA	>= 450.51.05	>= 451.48
CUDA 11.0.1 RC	>= 450.36.06	>= 451.22
CUDA 10.2.89	>= 440.33	>= 441.22
CUDA 10.1 (10.1.105 general release, and updates)	>= 418.39	>= 418.96
CUDA 10.0.130	>= 410.48	>= 411.31

CUDA Toolkit	Linux x86_64 Driver Version	Windows x86_64 Driver Version
CUDA 9.2 (9.2.148 Update 1)	>= 396.37	>= 398.26
CUDA 9.2 (9.2.88)	>= 396.26	>= 397.44
CUDA 9.1 (9.1.85)	>= 390.46	>= 391.29
CUDA 9.0 (9.0.76)	>= 384.81	>= 385.54
CUDA 8.0 (8.0.61 GA2)	>= 375.26	>= 376.51
CUDA 8.0 (8.0.44)	>= 367.48	>= 369.30
CUDA 7.5 (7.5.16)	>= 352.31	>= 353.66
CUDA 7.0 (7.0.28)	>= 346.46	>= 347.62

For convenience, the NVIDIA driver is installed as part of the CUDA Toolkit installation. Note that this driver is for development purposes and is not recommended for use in production with Tesla GPUs.

For running CUDA applications in production with Tesla GPUs, it is recommended to download the latest driver for Tesla GPUs from the NVIDIA driver downloads site at http://www.nvidia.com/drivers.

During the installation of the CUDA Toolkit, the installation of the NVIDIA driver may be skipped on Windows (when using the interactive or silent installation) or on Linux (by using meta packages).

For more information on customizing the install process on Windows, see http://docs.nvidia.com/cuda/cuda-installation-guide-microsoft-windows/index.html#install-cuda-software.

For meta packages on Linux, see https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html#package-manager-metas

1.2. General CUDA

- Added support for NVIDIA Ampere GPU architecture based GA10x GPUs GPUs (compute capability 8.6), including the GeForce RTX-30 series.
- Enhanced CUDA compatibility across minor releases of CUDA will enable CUDA applications to be compatible with all versions of a particular CUDA major release.
- CUDA 11.1 adds a new PTX Compiler static library that allows compilation of PTX programs using set of APIs provided by the library. See https://docs.nvidia.com/cuda/ptx-compiler-api/index.html for details.
- Added the 7.1 version of the Parallel Thread Execution instruction set architecture (ISA). For more details on new (sm_86 target, mma.sp) and deprecated instructions, see https://docs.nvidia.com/cuda/parallel-thread-execution/index.html#ptx-isa-version-7-1 in the PTX documentation.
- Added support for Fedora 32 and Debian 10.3 Buster on x86_64 platforms.
- Unified programming model for:

  - async-copy
  - async-pipeline
  - async-barrier (cuda::barrier)
- Added hardware accelerated sparse texture support.
- Added support for read-only mapping for `cudaHostRegister`.
- Multi-threaded launch to different CUDA streams is supported.
- CUDA Graphs enhancements:
  - improved graphExec update
  - external dependencies
  - extended memcopy APIs
  - presubmit
- Introduced new system level interface using /dev based capabilities for cgroups style isolation with MIG.
- Improved MPS error handling when using multi-GPUs.
- A fatal GPU exception generated by a Volta+ MPS client will be contained within the devices affected by it and other clients using those devices. Clients running on the other devices managed by the same MPS server can continue running as normal.
- Users can now configure and query the per-context time slice duration for a GPU via nvidia-smi. Configuring the time slice will require administrator privileges and the allowed settings are default, short, medium and long. The time slice will only be applicable to CUDA applications that are executed after the configuration is applied.
- Improved detection and reporting of unsupported configurations.

1.3. CUDA Tools

1.3.1. CUDA Compilers

- PTX Compiler is provided as a redistributable library.
- The following compilers are supported as host compilers in nvcc:
  - GCC 10.0
  - Clang 10.0

1.3.2. CUDA Developer Tools

- For new features, improvements, and bug fixes in CUPTI, see the changelog.
- For new features, improvements, and bug fixes in Nsight Compute, see the changelog.
- Application replay for metric collection.

1.4. CUDA Libraries

1.4.1. cuFFT Library

- cuFFT is now L2-cache aware and uses L2 cache for GPUs with more than 4.5MB of L2 cache. Performance may improve in certain single-GPU 3D C2C FFT cases.
- After successfully creating a plan, cuFFT now enforces a lock on the cufftHandle. Subsequent calls to any planning function with the same cufftHandle will fail.
- Added support for very large sizes (3k cube) to multi-GPU cuFFT on DGX-2.
- Improved performance on multi-gpu cuFFT for certain sizes (1k cube).

1.4.2. cuSOLVER Library

- Added new 64-bit APIs:
  - `cusolverDnXpotrf_bufferSize`
  - `cusolverDnXpotrf`
  - `cusolverDnXpotrs`
  - `cusolverDnXgeqrf_bufferSize`
  - `cusolverDnXgeqrf`
  - `cusolverDnXgetrf_bufferSize`
  - `cusolverDnXgetrf`
  - `cusolverDnXgetrs`
  - `cusolverDnXsyevd_bufferSize`
  - `cusolverDnXsyevd`
  - `cusolverDnXsyevdx_bufferSize`
  - `cusolverDnXsyevdx`
  - `cusolverDnXgesvd_bufferSize`
  - `cusolverDnXgesvd`
- Added a new SVD algorithm based on polar decomposition, called GESVDP which uses the new 64-bit API, including `cusolverDnXgesvdp_bufferSize` and `cusolverDnXgesvdp`.

1.4.3. CUDA Math Library

- Added host support for half and nv_bfloat16 converts to/from integer types.
- Added `__hcmadd()` device only API for fast half2 and nv_bfloat162 based complex multiply-accumulate.

1.5. Deprecated Features

The following features are deprecated in the current release of the CUDA software. The features still work in the current release, but their documentation may have been removed, and they will become officially unsupported in a future release. We recommend that developers employ alternative solutions to these features in their software.

General CUDA

- Support for Ubuntu in IBM's ppc64le platforms is deprecated in this release and will be dropped in a future CUDA release.

CUDA Tools

- Support for VS2015 is deprecated. Older Visual Studio versions including VS2012 and VS2013 are also deprecated and support may be dropped in a future release of CUDA.

CUDA Libraries

- The following cuSOLVER 64-bit APIs are deprecated:
  - `cusolverDnPotrf_bufferSize`
  - `cusolverDnPotrf`
  - `cusolverDnPotrs`
  - `cusolverDnGeqrf_bufferSize`
  - `cusolverDnGeqrf`
  - `cusolverDnGetrf_bufferSize`
  - `cusolverDnGetrf`
  - `cusolverDnGetrs`
  - `cusolverDnSyevd_bufferSize`
  - `cusolverDnSyevd`
  - `cusolverDnSyevdx_bufferSize`
  - `cusolverDnSyevdx`
  - `cusolverDnGesvd_bufferSize`
  - `cusolverDnGesvd`

1.6. Resolved Issues

1.6.1. General CUDA

- Fixed an issue that caused `cuD3D11GetDevices()` to return a misleading error code.

- Fixed an issue that caused `cuda_ipc_open` to fail with CUDA_ERROR_INVALID_HANDLE. (
- Fixed an issue that caused the nvidia-ml library to be installed in a different location from the one specified in pkg-config.
- Fixed an issue that caused some streaming apps to trigger CUDA safe detection.
- Fixed an issue that caused unexpectedly large host memory usage when loading cubin.
- Fixed an issue with the paths for .pc files in the CUDA SLES15 repo.
- Fixed an issue that caused warnings to be considered fatal when installing nvidia-drivers modules with kickstart.
- Resolved a memory issue when using `cudaGraphInstantiate`.
- Read-only OS_DESCRIPTOR allocations are now supported.
- Loading an application against the libcuda.so stub library now returns a helpful error message.
- The `cudaOccupancy`* API is now available even when `__CUDA_ACC__` is not defined.

1.6.2. CUDA Tools

- When tracing graphs, grid/block dimensions showed in nvvp and nsight-sys were not always correct. This has been resolved.
- Fixed an issue that prevented profiling with nvprof without setting LD_LIBRARY_PATH to the lib64 folder.
- The Visual Profiler "Varying Register Count" graph's x-axis has changed from 65536 to 255 and the device limit is now 255.
- Added nvswitch init error checking improvements for DMA, MSI, and SOE.
- Improved detection and reporting of unsupported configurations.

1.6.3. cuBLAS Library

- A performance regression in the `cublasCgetrfBatched` and `cublasCgetriBatched` routines has been fixed.
- The IMMA kernels do not support padding in matrix C and may corrupt the data when matrix C with padding is supplied to `cublasLtMatmul`. A suggested work around is to supply matrix C with leading dimension equal to 32 times the number of rows when targeting the IMMA kernels: computeType = CUDA_R_32I and CUBLASLT_ORDER_COL32 for matrices A,C,D, and CUBLASLT_ORDER_COL4_4R2_8C (on NVIDIA Ampere GPU architecture or Turing architecture) or CUBLASLT_ORDER_COL32_2R_4R4 (on NVIDIA Ampere GPU architecture) for matrix B. Matmul descriptor must specify CUBLAS_OP_T on matrix B and CUBLAS_OP_N (default) on matrix A and C. The data corruption behavior was fixed so that CUBLAS_STATUS_NOT_SUPPORTED is returned instead.
- Fixed an issue that caused an Address out of bounds error when calling `cublasSgemm()`.
- A performance regression in the `cublasCgetrfBatched` and `cublasCgetriBatched` routines has been fixed.

1.6.4. cuFFT Library

- Resolved an issue that caused cuFFT to crash when reusing a handle after clearing a callback.
- Fixed an error which produced incorrect results / NaN values when running a real-to-complex FFT in half precision.

1.7. Known Issues

1.7.1. cuFFT Library

- cuFFT will always overwrite the input for out-of-place C2R transform.
- For cuFFT's 1D multi-GPU transforms, cufftXtSetGPUs *whichGPUs* parameter needs to enumerate the first N GPUs, rather than arbitrary GPUs.