NVIDIA CUDA Toolkit

Release Notes for CUDA 11.1 Update 1

Table of Contents

List of Tables

Chapter 1. CUDA Toolkit Major Components

This section provides an overview of the major components of the NVIDIA® CUDA® Toolkit and points to their locations after installation.

Compiler

The CUDA-C and CUDA-C++ compiler, `nvcc`, is found in the `bin/` directory. It is built on top of the NVVM optimizer, which is itself built on top of the LLVM compiler infrastructure. Developers who want to target NVVM directly can do so using the Compiler SDK, which is available in the `nvvm/` directory.

Please note that the following files are compiler-internal and subject to change without any prior notice.

▶ any file in `include/crt` and `bin/crt`

▶ `include/common_functions.h`, `include/device_double_functions.h`, `include/device_functions.h`, `include/host_config.h`, `include/host_defines.h`, and `include/math_functions.h`

▶ `nvvm/bin/cicc`

▶ `bin/cudafe++`, `bin/bin2c`, and `bin/fatbinary`

Tools

The following development tools are available in the `bin/` directory (except for Nsight Visual Studio Edition (VSE) which is installed as a plug-in to Microsoft Visual Studio, Nsight Compute and Nsight Systems are available in a separate directory).

▶ IDEs: `nsight` (Linux, Mac), Nsight VSE (Windows)

▶ Debuggers: `cuda-memcheck`, `cuda-gdb` (Linux), Nsight VSE (Windows)

▶ Profilers: Nsight Systems, Nsight Compute, `nvprof`, `nvvp`, `ncu`, Nsight VSE (Windows)

▶ Utilities: `cuobjdump`, `nvdisasm`

Libraries

The scientific and utility libraries listed below are available in the `lib64/` directory (DLLs on Windows are in `bin/`), and their interfaces are available in the `include/` directory.

▶ `cub` (High performance primitives for CUDA)

▶ `cublas` (BLAS)

- ▶ `cublas_device` (BLAS Kernel Interface)
- ▶ `cuda_occupancy` (Kernel Occupancy Calculation [header file implementation])
- ▶ `cudadevrt` (CUDA Device Runtime)
- ▶ `cudart` (CUDA Runtime)
- ▶ `cufft` (Fast Fourier Transform [FFT])
- ▶ `cupti` (CUDA Profiling Tools Interface)
- ▶ `curand` (Random Number Generation)
- ▶ `cusolver` (Dense and Sparse Direct Linear Solvers and Eigen Solvers)
- ▶ `cusparse` (Sparse Matrix)
- ▶ `libcu++` (CUDA Standard C++ Library)
- ▶ `nvJPEG` (JPEG encoding/decoding)
- ▶ `npp` (NVIDIA Performance Primitives [image and signal processing])
- ▶ `nvblas` ("Drop-in" BLAS)
- ▶ `nvcuvid` (CUDA Video Decoder [Windows, Linux])
- ▶ `nvml` (NVIDIA Management Library)
- ▶ `nvrtc` (CUDA Runtime Compilation)
- ▶ `nvtx` (NVIDIA Tools Extension)
- ▶ `thrust` (Parallel Algorithm Library [header file implementation])

CUDA Samples

Code samples that illustrate how to use various CUDA and library APIs are available in the `samples/` directory on Linux and Mac, and are installed to `C:\ProgramData\NVIDIA Corporation\CUDA Samples` on Windows. On Linux and Mac, the `samples/` directory is read-only and the samples must be copied to another location if they are to be modified. Further instructions can be found in the *Getting Started Guides* for Linux and Mac.

Documentation

The most current version of these release notes can be found online at http://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html. Also, the `version.txt` file in the root directory of the toolkit will contain the version and build number of the installed toolkit.

Documentation can be found in PDF form in the `doc/pdf/` directory, or in HTML form at `doc/html/index.html` and online at http://docs.nvidia.com/cuda/index.html.

CUDA-GDB Sources

CUDA-GDB sources are available as follows:

- ▶ For CUDA Toolkit 7.0 and newer, in the installation directory `extras/`. The directory is created by default during the toolkit installation unless the `.rpm` or `.deb` package installer is used. In this case, the `cuda-gdb-src` package must be manually installed.
- ▶ For CUDA Toolkit 6.5, 6.0, and 5.5, at https://github.com/NVIDIA/cuda-gdb.

▶ For CUDA Toolkit 5.0 and earlier, at ftp://download.nvidia.com/CUDAOpen64/.

▶ Upon request by sending an e-mail to mailto:oss-requests@nvidia.com.

Chapter 2.   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.

2.1.   CUDA Toolkit Major Component Versions

CUDA Components

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

For CUDA 11.1.1, the table below indicates the versions:

Table 1.   CUDA 11.1 Update 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	11.1.105	x86_64, POWER, Arm64
CUDA Demo Suite	11.1.74	x86_64
CUDA GDB	11.1.105	x86_64, POWER, Arm64
CUDA Memcheck	11.1.105	x86_64, POWER
CUDA NVCC	11.1.105	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	11.1.105	x86_64, POWER, Arm64
CUDA nvprune	11.1.74	x86_64, POWER, Arm64
CUDA NVRTC	11.1.105	x86_64, POWER, Arm64
CUDA NVTX	11.1.74	x86_64, POWER, Arm64
CUDA NVVP	11.1.105	x86_64, POWER
CUDA Samples	11.1.105	x86_64, POWER, Arm64
CUDA Compute Sanitizer API	11.1.105	x86_64, POWER, Arm64

Component Name	Version Information	Supported Architectures
CUDA cuBLAS	11.3.0.106	x86_64, POWER, Arm64
CUDA cuFFT	10.3.0.105	x86_64, POWER, Arm64
CUDA cuRAND	10.2.2.105	x86_64, POWER, Arm64
CUDA cuSOLVER	11.0.1.105	x86_64, POWER, Arm64
CUDA cuSPARSE	11.3.0.10	x86_64, POWER, Arm64
CUDA NPP	11.1.2.301	x86_64, POWER, Arm64
CUDA nvJPEG	11.3.0.105	x86_64, POWER, Arm64
Nsight Eclipse Plugins	11.1.74	x86_64, POWER
Nsight Compute	2020.2.1.8	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)	2020.2.0.20284	x86_64 (Windows)
NVIDIA Linux Driver	455.32.00	x86_64, POWER, Arm64
NVIDIA Windows Driver	456.81	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 11.1.1 Update 1	>=455.32	>=456.81
CUDA 11.1 GA	>=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 Toolkit	Linux x86_64 Driver Version	Windows x86_64 Driver Version
CUDA 10.0.130	>= 410.48	>= 411.31
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

2.2. What's New in CUDA 11.1 Update 1

This section summarizes the changes in CUDA 11.1 Update 1 since the 11.1 GA release.

New Features

▶ **General CUDA**

   ▶ CUDA 11.1 Update 1 is a minor update that is binary compatible with CUDA 11.1. This release will work with all versions of the R450 NVIDIA driver.

▶ **nvJPEG**

   ▶ Added error handling capabilities for nonstandard JPEG images.

▶ **cuBLAS**

   ▶ cuBLASLt Logging is officially stable and no longer experimental. cuBLASLt Logging APIs are still experimental and may change in future releases.

▶ **cuSPARSE**

▶ `cusparseSparseToDense`

　　▶ CSR, CSC, or COO conversion to dense representation

　　▶ Support row-major and column-major layouts

　　▶ Support all data types

　　▶ Support 32-bit and 64-bit indices

　　▶ Provide performance 3x higher than `cusparseXcsc2dense`, `cusparseXcsr2dense`

▶ `cusparseDenseToSparse`

　　▶ Dense representation to CSR, CSC, or COO

　　▶ Support row-major and column-major layouts

　　▶ Support all data types

　　▶ Support 32-bit and 64-bit indices

　　▶ Provide performance 3x higher than `cusparseXcsc2dense`, `cusparseXcsr2dense`

Known Issues

▶ This toolkit release contains security fixes. Please refer to the <u>Security Bulletin</u> for more information on the security fixes provided in this toolkit release

▶ cuSOLVER

　　▶ `cusolverDnIRSXgels` may return `CUSOLVER_STATUS_INTERNAL_ERROR`. when the precision is 'z' due to insufficient workspace which causes illegal memory access. The `cusolverDnIRSXgels_bufferSize()` does not report the correct size of workspace. To workaround the issue, the user has to add more workspace than what is reported by `cusolverDnIRSXgels_bufferSize()`.
For example, if x is the size of workspace returned by `cusolverDnIRSXgels_bufferSize()`, then the user has to allocate `(x + min(m,n)*sizeof(cuDoubleComplex))` bytes.

▶ cuSPARSE

　　▶ `cusparseXdense2csr` provides incorrect results for some matrix sizes.

Resolved Issues

▶ **cuBLAS**

　　▶ cublasLt Matmul fails on Volta architecture GPUs with `CUBLAS_STATUS_EXECUTION_FAILED` when n dimension > 262,137 and epilogue bias feature is being used. This issue exists in 11.0 and 11.1 releases but has been corrected in 11.1 Update 1.

▶ **cuSOLVER**

　　▶ `cusolverDnDDgels` reports `IRS_NOT_SUPPORTED` when m > n. The issue has been fixed in release 11.1 U1, so `cusolverDnDDgels` will support m > n.

▶ `cusolverMgDeviceSelect` can consume over 1GB device memory. The issue has been fixed in release 11.1 U1. The hidden memory allocation inside cusolverMG handle is about 30 MB per device.

Deprecations

▶ **cuSPARSE**

   ▶ Legacy conversion routines: `cusparseXcsc2dense`, `cusparseXcsr2dense`, `cusparseXdense2csc`, `cusparseXdense2csr`

2.3.   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`.

▶ 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.

2.4.   CUDA Tools

2.4.1.   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

2.4.2.   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.

2.5.   CUDA Libraries

2.5.1.   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).

2.5.2.   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`.

2.5.3.   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.

2.6.   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

▶ CUDA 11.1 is the last release to support Ubuntu distributions on IBM POWER (ppc64le) platforms. Starting with the next release of CUDA, only Red Hat Enterprise Linux (RHEL) will be supported on IBM POWER.

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`

2.7.   Resolved Issues

2.7.1.   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.

2.7.2.   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.

2.7.3.   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.

2.7.4.   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.

2.8.   Known Issues

2.8.1.   cuFFT Library

▶ cuFFT will always overwrite the input for out-of-place C2R transform.

▶ Single dimensional multi-GPU FFT plans ignore user input on the `whichGPUs` parameter of `cufftXtSetGPUs()` and assume that GPUs IDs are always numbered from 0 to N-1.

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