NVIDIA TensorRT

Support Matrix | NVIDIA Docs

TRM-09025-001 _v8.0.0 Early Access (EA) | May 2021

Table of Contents

Chapter 1. Features For Platforms And Software

This section lists the supported TensorRT features based on which platform and software.

Table 1. List of supported features per platform.

	Linux x86-64		Windows x64	Linux ppc64le	Linux AArch64
	8.0.0 EA	8.0.x GA	8.0.x	8.0.x	8.0.x
Supported CUDA versions	11.3[1] 11.0 update 1 10.2	11.3 update 1[2] 11.2 update 2[2] 11.1 update 1[2] 11.0 update 1[2] 10.2	11.3 update 1 11.2 update 2 11.1 update 1 11.0 update 1 10.2	11.3 update 1	11.3 update 1 10.2
Supported cuBLAS versions	11.4.2.x 11.2.0.252 10.2.3.254	11.4.2.x 11.4.1.1043 11.3.0.106 11.2.0.252 10.2.3.254	11.4.2.x 11.4.1.1043 11.3.0.106 11.2.0.252 10.2.3.254	11.4.2.x	11.4.2.x 10.2.2.214
Supported cuDNN versions	cuDNN 8.2.0	cuDNN 8.2.0	cuDNN 8.2.0	cuDNN 8.2.0	cuDNN 8.2.0
TensorRT Python API	Yes	Yes	No	Yes	Yes

1 This build supports CUDA compute capability 8.6. It is compatible with CUDA 11.1 and CUDA 11.2. User-mode driver compatible with the runtime CUDA version is required and >= 465 is suggested for best performance.

2 These CUDA versions are supported using a single build, built with CUDA Toolkit 11.3 update 1. It is compatible with all CUDA 11.x (x <= 3) versions and only requires driver 450.x. For future CUDA 11.x (x > 3) versions, the corresponding 11.x driver is required which matches the CUDA Toolkit.

	Linux x86-64		Windows x64	Linux ppc64le	Linux AArch64
	8.0.0 EA	8.0.x GA	8.0.x	8.0.x	8.0.x
NvUffParser	Yes	Yes	Yes	Yes	Yes
NvOnnxParser	Yes	Yes	Yes	Yes	Yes
Loops	Yes	Yes	Yes	Yes	Yes

Note:

- Serialized engines are not portable across platforms or TensorRT versions.
- Refer to the minimum compatible driver versions in the CUDA Release Notes for specific NVIDIA Driver versions.

Chapter 2. Layers And Features

The section lists the supported TensorRT layers and each of the features.

About this task

Note:

- **Supports broadcast** indicates support for broadcast in this layer. This layer allows its two input tensors to be of dimensions [1, 5, 4, 3] and [1, 5, 1, 1], and its output is [1, 5, 4, 3]. The second input tensor has been broadcast in the innermost 2 dimensions.
- **Supports broadcast across batch** indicates support for broadcast across the batch dimension. "NA" in this column means it's not allowed in networks with an implicit batch dimension.

Table 2. List of supported features per TensorRT layer.

Layer	Dimensions of input tensor	Dimensions of output tensor	Does the operation apply to only the innermost 3 dimensions?	Supports broadcast	Supports broadcast across batch
IActivationLayer	0-7 dimensions	0-7 dimensions	No	No	No
IConcatenationLayer	1-7 dimensions	1-7 dimensions	No	No	No
IConstantLayer	has no inputs	0-7 dimensions	No	No	Always
IConvolutionLayer > 2D Convolution	3 or more dimensions	3 or more dimensions	Yes	No	No
IConvolutionLayer > 3D Convolution	4 or more dimensions	4 or more dimensions	No	No	No

Layer	Dimensions of input tensor	Dimensions of output tensor	Does the operation apply to only the innermost 3 dimensions?	Supports broadcast	Supports broadcast across batch
IDeconvolutionLayer > 2D Deconvolution	3 or more dimensions	3 or more dimensions	Yes	No	No
IDeconvolutionLayer > 3D Deconvolution	4 or more dimensions	4 or more dimensions	No	No	No
IDequantizeLayer	2 or more dimensions	2 or more dimensions	Yes	No	No
IElementWiseLayer	0-7 dimensions	0-7 dimensions	No	Yes	Yes
IFillLayer	1 dimension	0-7 dimensions	No	NA	NA
IFullyConnectedLayer	3 or more dimensions	3 or more dimensions	Yes	No	No
IGatherLayer	▶ Input1: 1-7 dimensions ▶ Input2: 0-7 dimensions	0-7 dimensions	No	No	Yes
IIdentityLayer	0-7 dimensions	0-7 dimensions	No	No	No
IIteratorLayer	1-7 dimensions	0-6 dimensions	No	No	NA
ILoopOutputLayer	0-7 dimensions	0-7 dimensions	No	No	NA
ILRNLayer	3 or more dimensions	3 or more dimensions	Yes	No	No
IMatrixMultiplyLayer	2 or more dimensions	2 or more dimensions	No	Yes	Yes
IPaddingLayer	3 or more dimensions	3 or more dimensions	Yes	No	No
IParametricReluLayer	1-7 dimensions	1-7 dimensions	No	No	No
IPluginV2Layer	User defined	User defined	User defined	User defined	User defined
IPoolingLayer > 2D Pooling	3 or more dimensions	3 or more dimensions	Yes	Yes	Yes
IPoolingLayer > 3D Pooling	4 or more dimensions	4 or more dimensions	No	Yes	Yes

Layer	Dimensions of input tensor	Dimensions of output tensor	Does the operation apply to only the innermost 3 dimensions?	Supports broadcast	Supports broadcast across batch
IQuantizeLayer	2 or more dimensions	2 or more dimensions	Yes	No	No
IRaggedSoftMaxLayer	▶ Input: 2 dimensions ▶ Bounds: 2 dimensions	2 or more dimensions	No	No	Yes
IRecurrenceLayer	0-7 dimensions	0-7 dimensions	No	No	NA
IReduceLayer	1-7 dimensions	0-7 dimensions	No	No	No
IResizeLayer	1-7 dimensions	1-7 dimensions	No	No	No
IRNNv2Layer	▶ Data/ Hidden/ Cell: 2 or more dimensions ▶ Seqlen: 0 or more dimensions	Data/Hidden/ Cell: 2 or more dimensions	No	No	No
IScaleLayer	3 or more dimensions	3 or more dimensions	Yes	No	No
ISelectLayer	0-7 dimensions	0-7 dimensions	No	Yes	NA
IShapeLayer	1 or more dimensions	1 dimension	No	No	NA
IShuffleLayer	0-7 dimensions	0-7 dimensions	No	No	No
ISliceLayer	1-7 dimensions	1-7 dimensions	No	No	Yes
ISoftMaxLayer	1-7 dimensions	1-7 dimensions	No	No	Yes
ITopKLayer	1-7 dimensions	▶ Output1: 1-7 dimensions ▶ Output2: 1-7 dimensions	Yes	No	Yes

Layer	Dimensions of input tensor	Dimensions of output tensor	Does the operation apply to only the innermost 3 dimensions?	Supports broadcast	Supports broadcast across batch
ITripLimitLayer	0 dimensions	has no outputs	No	No	NA
IUnaryLayer	1-7 dimensions	1-7 dimensions	No	No	No

For more information about each of the TensorRT layers, see TensorRT Layers.

Chapter 3. Layers And Precision

The section lists the TensorRT layers and the precision modes that each layer supports. It also lists the ability of the layer to run on Deep Learning Accelerator (DLA).
For more information about additional constraints, see DLA Supported Layers.

For more information about each of the TensorRT layers, see TensorRT Layers. To view a list of the specific attributes that are supported by each layer, refer to the TensorRT API documentation.

Table 3. List of supported precision modes per TensorRT layer.

Layer	FP32	FP16	INT8	INT32	DLA FP16	DLA INT8
IActivationLayer	Yes	Yes	Yes	No	Yes[3]	Yes[4]
IConcatenationLayer	Yes	Yes	Yes	Yes	Yes[5]	Yes[5]
IConstantLayer	Yes	Yes	Yes	Yes	No	No
IConvolutionLayer > 2D Convolution	Yes	Yes	Yes	No	Yes	Yes
IConvolutionLayer > 3D Convolution	Yes	Yes	Yes	No	No	No
IDeconvolutionLayer > 2D Deconvolution	Yes	Yes	Yes	No	Yes	Yes[6]
IDeconvolutionLayer > 3D Deconvolution	Yes	Yes	No	No	No	No
IDequantizeLayer	No	No	Yes	No	No	No

[3] Partial support. *Yes* for `ReLU`, `Clipped ReLU`, `Leaky ReLU`, `Sigmoid` and `TanH` activation types only.
[4] Partial support. *Yes* for `ReLU`, `Clipped ReLU`, `Leaky ReLU`, `Sigmoid` and `TanH` activation type only.
[5] Partial support. *Yes* for concatenation across `C` dimension only.
[6] Partial support. *Yes* for ungrouped deconvolutions and *No* for grouped.

Layer	FP32	FP16	INT8	INT32	DLA FP16	DLA INT8
IElementWiseLayer	Yes	Yes	No	Yes	Yes[7]	Yes[8]
IFillLayer	Yes	No	No	Yes	No	No
IFullyConnectedLayer	Yes	Yes	Yes	No	Yes	Yes
IGatherLayer	Yes	Yes	No	Yes	No	No
IIdentityLayer	Yes	Yes	Yes	Yes	No	No
IIteratorLayer	Yes	Yes	No	Yes	No	No
ILoopOutputLayer	Yes	Yes	No	Yes	No	No
IPluginV2Layer	Yes	Yes	Yes	No	No	No
ILRNLayer	Yes	Yes	Yes	No	Yes	No
IMatrixMultiplyLayer	Yes	Yes	No	No	No	No
IPaddingLayer	Yes	Yes	Yes	No	No	No
IParametricReLuLayer	Yes	Yes	Yes	No	No	No
IPoolingLayer > 2D Pooling	Yes	Yes	Yes	No	Yes[9]	Yes[9]
IPoolingLayer > 3D Pooling	Yes	Yes	No	No	No	No
IQuantizeLayer	Yes	No	No	No	No	No
IRaggedSoftMaxLayer	Yes	No	No	No	No	No
IRecurrenceLayer	Yes	Yes	No	Yes	No	No
IReduceLayer	Yes	Yes	No	No	No	No
IResizeLayer	Yes	Yes	No	No	No	No
IRNNv2Layer	Yes	Yes	No	No	No	No
IScaleLayer	Yes	Yes	Yes	No	Yes[10]	Yes[10]
ISelectLayer	Yes	Yes	No	Yes	No	No
IShapeLayer[11]	Yes	Yes	Yes	Yes	No	No
IShuffleLayer	Yes	Yes	Yes	Yes	No	No
ISliceLayer	Yes	Yes	No[12]	Yes	No	No
ISoftMaxLayer	Yes	Yes	No	No	No	No
ITopKLayer	Yes	Yes	No	No	No	No

[7] Partial support. *Yes* for `sum`, `sub`, `prod`, `min` and `max` elementwise operations only.
[8] Partial support. *Yes* for `sum` elementwise operation only.
[9] Partial support. *Yes* for `max` and average padding inclusive pooling type only.
[10] Partial support. DLA does not support power on scale layer.
[11] Output is always INT32.
[12] Partial support. *Yes* for unstrided Slice and *No* for strided.

Layer	FP32	FP16	INT8	INT32	DLA FP16	DLA INT8
ITripLimitLayer	Yes	Yes	No	Yes	No	No
IUnaryLayer	Yes	Yes	No	No	No	No

Note: DLA with FP16/INT8 precision with some restrictions on layer parameters.

Chapter 4. Hardware And Precision

The following table lists NVIDIA hardware and which precision modes each hardware supports. TensorRT supports all NVIDIA hardware with capability SM 5.0 or higher. It also lists the availability of Deep Learning Accelerator (DLA) on this hardware. Refer to the following tables for the specifics.

> **Note:** Support for CUDA Compute Capability version 3.0 has been removed. Support for CUDA Compute Capability versions below 5.0 may be removed in a future release and is now deprecated.

Table 4. Supported hardware

CUDA Compute Capability	Example Device	TF32	FP32	FP16	INT8	FP16 Tensor Cores	INT8 Tensor Cores	DLA
8.6	GeForce 3090	Yes	Yes	Yes	Yes	Yes	Yes	No
8.0	NVIDIA A100/ GA100 GPU	Yes	Yes	Yes	Yes	Yes	Yes	No
7.5	Tesla T4	No	Yes	Yes	Yes	Yes	Yes	No
7.2	Jetson AGX Xavier	No	Yes	Yes	Yes	Yes	Yes	Yes
7.0	Tesla V100	No	Yes	Yes	Yes	Yes	No	No
6.2	Jetson TX2	No	Yes	Yes	No	No	No	No
6.1	Tesla P4	No	Yes	No	Yes	No	No	No
6.0	Tesla P100	No	Yes	Yes	No	No	No	No

CUDA Compute Capability	Example Device	TF32	FP32	FP16	INT8	FP16 Tensor Cores	INT8 Tensor Cores	DLA
5.3	Jetson TX1	No	Yes	Yes	No	No	No	No
5.2	Tesla M4	No	Yes	No	No	No	No	No
5.0	Quadro K2200	No	Yes	No	No	No	No	No

Deprecated hardware

Table 5. List of supported precision mode per hardware.

CUDA Compute Capability	Example Device	FP32	FP16	INT8	FP16 Tensor Cores	INT8 Tensor Cores	DLA
3.7	Tesla K80	Yes	No	No	No	No	No
3.5	Tesla K40	Yes	No	No	No	No	No

Removed hardware

Table 6. List of supported precision mode per hardware.

CUDA Compute Capability	Example Device	FP32	FP16	INT8	FP16 Tensor Cores	INT8 Tensor Cores	DLA
3.0	Tesla K10	Yes	No	No	No	No	No

Chapter 5. Software Versions Per Platform

The section lists the supported software versions based on platform.

Table 7. List of supported platforms per software version.

	Compiler version	Python versions
Ubuntu 16.04 x86-64	gcc 8.3.1	3.5
Ubuntu 18.04 x86-64	gcc 8.3.1	3.6
Ubuntu 20.04 x86-64	gcc 8.3.1	3.8
CentOS 7.9 x86-64	gcc 8.3.1	3.6
CentOS 8.3 x86-64	gcc 8.3.1	3.8
SLES 15 x86-64	gcc 8.3.1	N/A
Windows 10 x64	MSVC 2017u5	N/A
CentOS 8.3 ppc64le	Clang 10.0.1	3.8
Ubuntu 20.04 SBSA	gcc 8.4.0	3.8
JetPack AArch64	gcc 7.5.0	3.6

Note: Python versions supported when using Debian or RPM packages. When using Python wheel files, versions 3.5, 3.6, 3.7, 3.8, and 3.9 are supported.

Chapter 6. Supported Ops

The section lists the operations that are supported in a Caffe or TensorFlow framework and in the ONNX TensorRT parser.

Caffe

These are the operations that are supported in a Caffe framework:

- `BatchNormalization`
- `BNLL`
- `Clip`[13]
- `Concatenation`
- `Convolution`
- `Crop`
- `Deconvolution`
- `Dropout`
- `ElementWise`
- `ELU`
- `InnerProduct`
- `Input`
- `LeakyReLU`
- `LRN`
- `Permute`
- `Pooling`
- `Power`
- `Reduction`
- `ReLU`, `TanH`, and `Sigmoid`
- `Reshape`
- `SoftMax`
- `Scale`

[13] When using the `Clip` operation, Caffe users must serialize their layers using `ditcaffe.pb.h` instead of `caffe.pb.h` in order to import the layer into TensorRT.

TensorFlow

These are the operations that are supported in a TensorFlow framework:

- `Add`, `AddV2`, `AddN`, `Sub`, `Mul`, `Div`, `FloorDiv`, `RealDiv`, `Minimum`, `Maximum`
- `AvgPool`, `AvgPool3D`[14]
- `ArgMin`
- `AvgPool`
- `BiasAdd`
- `Cast`[14]
- `Clip`
- `CombinedNonMaxSuppression`
- `ConcatV2`
- `Const`
- `Conv2D`, `Conv3D`
- `Conv2DBackpropInput`, `Conv3DBackpropInputV2`
- `ConvTranspose2D`
- `DepthToSpace`
- `DepthwiseConv2dNative`
- `Elu`
- `ExpandDims`
- `FusedBatchNorm`, `FusedBatchNormV2`, `FusedBatchNormV3`
- `FusedConv2DBiasActivation`
- `GatherV2`
- `Identity`
- `LeakyReLU`
- `MatMul`, `BatchMatMul`, `BatchMatMulV2`
- `MaxPool`, `MaxPool3D`[14]
- `Mean`
- `Negative`, `Abs`, `Sqrt`, `Recip`, `Rsqrt`, `Pow`, `Exp`, `Log`, `Square`
- `Pad` is supported if followed by one of these TensorFlow layers: `Conv2D`, `DepthwiseConv2dNative`, `MaxPool`, and `AvgPool`.
- `Pack`, `Unpack`
- `ReLU`, `TanH`, `Sigmoid`
- `Relu6`
- `Reshape`

[14] Supported only in TensorFlow 2.0.

- `ResizeBilinear`, `ResizeNearestNeighbor`
- `Sin`, `Cos`, `Tan`, `Asin`, `Acos`, `Atan`, `Sinh`, `Cosh`, `Asinh`, `Acosh`, `Atanh`, `Ceil`, `Floor`
- `Selu`
- `Shape`[14]
- `Slice`, `StridedSlice`
- `SoftMax`

> **Note:** If the input to a TensorFlow `SoftMax` op is not `NHWC`, TensorFlow will automatically insert a transpose layer with a non-constant permutation, causing the UFF converter to fail. It is therefore advisable to manually transpose `SoftMax` inputs to `NHWC` using a constant permutation.

- `Softplus`
- `Softsign`
- `SpaceToDepth`
- `Split`
- `SquaredDifference`
- `Squeeze`
- `TopKV2`
- `Transpose`

For the list of ops supported in UFF, see UFF Operators.

ONNX

Since the ONNX parser is an open source project, the most up-to-date information regarding the supported operations can be found here.

These are the ONNX operators that are supported by TensorRT:

- `Abs`
- `Acos`
- `Acosh`
- `And`
- `Asin`
- `Asinh`
- `Atan`
- `Atanh`
- `Add`
- `ArgMax`
- `ArgMin`

- `AveragePool`
- `BatchNormalization`
- `Cast`
- `Ceil`
- `Celu`
- `Clip`
- `Concat`
- `Constant`
- `ConstantOfShape`
- `Conv`
- `ConvTranspose`
- `Cos`
- `Cosh`
- `CumSum`
- `DepthToSpace`
- `DequantizeLinear`
- `Div`
- `Dropout`
- `Elu`
- `Equal`
- `Erf`
- `Exp`
- `Expand`
- `EyeLike`
- `Flatten`
- `Floor`
- `Gather`
- `GatherElements`
- `Gemm`
- `GlobalAveragePool`
- `GlobalLpPool`
- `GlobalMaxPool`
- `Greater`
- `GreaterOrEqual`
- `GRU`
- `HardSigmoid`

- `Identity`
- `ImageScaler`
- `InstanceNormalization`
- `LeakyRelU`
- `Less`
- `LessOrEqual`
- `Log`
- `LogSoftmax`
- `Loop`
- `LpNormalization`
- `LpPool`
- `LRN`
- `LSTM`
- `MatMul`
- `Max`
- `MaxPool`
- `Mean`
- `Min`
- `Mul`
- `Neg`
- `Not`
- `Or`
- `Pad`
- `ParametricSoftplus`
- `Pow`
- `PRelu`
- `QuantizeLinear`
- `RandomUniform`
- `RandomUniformLike`
- `Range`
- `Reciprocal`
- `ReduceL1`
- `ReduceL2`
- `ReduceLogSum`
- `ReduceLogSumExp`
- `ReduceMax`

- `ReduceMean`
- `ReduceMin`
- `ReduceProd`
- `ReduceSum`
- `ReduceSumSquare`
- `Relu`
- `Reshape`
- `Resize`
- `ReverseSequence`
- `RNN`
- `ScaledTanh`
- `Scan`
- `Selu`
- `Shape`
- `Sigmoid`
- `Sin`
- `Sinh`
- `Size`
- `Slice`
- `Softmax`
- `SoftmaxCrossEntropyLoss`
- `Softplus`
- `Softsign`
- `SpaceToDepth`
- `Split`
- `Sqrt`
- `Squeeze`
- `Sub`
- `Sum`
- `Tan`
- `Tanh`
- `ThresholdedRelu`
- `Tile`
- `TopK`
- `Transpose`
- `Unsqueeze`

- `Upsample`
- `Where`

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.

VESA DisplayPort

DisplayPort and DisplayPort Compliance Logo, DisplayPort Compliance Logo for Dual-mode Sources, and DisplayPort Compliance Logo for Active Cables are trademarks owned by the Video Electronics Standards Association in the United States and other countries.

HDMI

HDMI, the HDMI logo, and High-Definition Multimedia Interface are trademarks or registered trademarks of HDMI Licensing LLC.

ARM

ARM, AMBA and ARM Powered are registered trademarks of ARM Limited. Cortex, MPCore and Mali are trademarks of ARM Limited. All other brands or product names are the property of their respective holders. "ARM" is used to represent ARM Holdings plc; its operating company ARM Limited; and the regional subsidiaries ARM Inc.; ARM KK; ARM Korea Limited.; ARM Taiwan Limited; ARM France SAS; ARM Consulting (Shanghai) Co. Ltd.; ARM Germany GmbH; ARM Embedded Technologies Pvt. Ltd.; ARM Norway, AS and ARM Sweden AB.

OpenCL

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

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

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