NVIDIA VIDEO CODEC SDK - ENCODER

Application Note

vNVENC_DA-6209-001_v14 | October 2020

Table of Contents

Chapter 1. NVIDIA Hardware Video Encoder

1.1. Introduction

NVIDIA GPUs - beginning with the Kepler generation - contain a hardware-based encoder (referred to as NVENC in this document) which provides fully accelerated hardware-based video encoding and is independent of graphics/CUDA cores. With end-to-end encoding offloaded to NVENC, the graphics/CUDA cores and the CPU cores are free for other operations. For example, in a game recording scenario, offloading the encoding to NVENC makes the graphics engine fully available for game rendering. In the video transcoding use-case, video encoding/decoding can happen on NVENC/NVDEC in parallel with other video post-/pre-processing on CUDA cores.

The hardware capabilities available in NVENC are exposed through APIs referred to as NVENCODE APIs in the document. This document provides information about the capabilities of the hardware encoder and features exposed through NVENCODE APIs.

1.2. NVENC Capabilities

NVENC can perform end-to-end encoding for H.264, HEVC 8-bit and HEVC 10-bit. This includes motion estimation and mode decision, motion compensation and residual coding, and entropy coding. It can also be used to generate motion vectors between two frames, which are useful for applications such as depth estimation, frame interpolation or encoding using other codecs not supported by NVENC. These operations are hardware accelerated by a dedicated block on GPU silicon die. NVENCODE APIs provide the necessary knobs to utilize the hardware encoding capabilities.

Table 1 summarizes the capabilities of the NVENC hardware exposed through NVENCODE APIs and Table 2 lists the features exposed in Video Codec SDK 11.0.

Table 1. NVENC Hardware Capabilities

Feature	Description	Kepler GPUs	1st Gen Maxwell GPUs	2nd Gen Maxwell GPUs	Pascal GPUs	Volta and TU117 GPUs	Ampere and Turing GPUs (except TU117)
H.264 baseline, main and high profiles	Capability to encode YUV 4:2:0 sequence and generate a H.264-bit stream.	Y	Y	Y	Y	Y	Y
H.264 4:4:4 encoding (only CAVLC)	Capability to encode YUV 4:4:4 sequence and generate a H.264-bit stream.	N	Y	Y	Y	Y	Y
H.264 lossless encoding	Lossless encoding.	N	Y	Y	Y	Y	Y
H.264 motion estimation (ME) only mode	Capability to provide macro-block level motion vectors and intra/inter modes.	N	Y	Y	Y	Y	Y
H.264 field encoding	Capability to encode field content.	Y	Y	Y	Y	Y	N
H.264/HEVC weighted prediction	Support for weighted prediction.	N	N	N	Y	Y	Y
Encoding support for H.264 ARGB content	Capability to encode RGB input.	Y	Y	Y	Y	Y	Y
Multiple reference frames for H.264	Capability to use different reference frames	N	N	N	N	N	Y
HEVC main profile	Capability to encode YUV 4:2:0 sequence and generate a HEVC bit stream.	N	N	Y	Y	Y	Y
HEVC lossless encoding	Lossless encoding.	N	N	N	Y	Y	Y
HEVC main10 profile	Support for encoding 10-bit content generate a HEVC bit stream.	N	N	N	Y	Y	Y
HEVC 4:4:4 encoding	Capability to encode YUV 4:4:4 sequence and generate a HEVC bit stream.	N	N	N	Y	Y	Y
HEVC motion estimation (ME) only mode	Capability to provide CTB level motion vectors and intra/inter modes.	N	N	N	Y	Y	Y
HEVC 8K encoding	Support for encoding 8192 × 8192 Content.	N	N	N	Y*	Y	Y
HEVC sample adaptive offset (SAO)	Improves encoded video quality.	N	N	N	Y	Y	Y
HEVC B frame	Improves encoded quality	N	N	N	N	N	Y
Multiple reference frames for HEVC	Capability to use different reference frames	N	N	N	N	N	Y

- **Y**: Supported, **N**: Not supported
- *Supported in select Pascal generation GPUs

Table 2. What's new in Video Codec SDK 11.0

New Feature	Description
Alpha Layer Encoding in HEVC	Video Codec SDK 11.0 introduces encoding alpha as an auxiliary layer in HEVC thus enabling application to encode transparency information in the bitstream. With this feature, the encoded bitstream has two layers - a base layer contains the YUV data and an auxiliary layer contain the alpha data. NVENCODE API provides additional controls to split bits between base and alpha layers.
Temporal SVC in H.264	Video Code SDK 11.0 introduces temporal SVC encoding as defined in Annex G of the H.264 specification. In temporal SVC, a bitstream consists of multiple temporal layer which allows an application to scale the frame rate. Note that other types scalabilites are not support by NVENCODE API.

1.3. NVENC Licensing Policy

There is no change in licensing policy in the current SDK in comparison to the previous SDK. The licensing policy is as follows:

As far as NVENC hardware encoding is concerned, NVIDIA GPUs are classified into two categories: "qualified" and "non-qualified". On qualified GPUs, the number of concurrent encode sessions is limited by available system resources (encoder capacity, system memory, video memory etc.). On non-qualified GPUs, the number of concurrent encode sessions is limited to 3 per system. This limit of 3 concurrent sessions per system applies to the combined number of encoding sessions executed on all non-qualified cards present in the system.

For a complete list of qualified and non-qualified GPUs, refer to https://developer.nvidia.com/nvidia-video-codec-sdk..

For example, on a system with one Quadro RTX4000 card (which is a qualified GPU) and three GeForce cards (which are non-qualified GPUs), the application can run N simultaneous encode sessions on Quadro RTX4000 card (where N is defined by the encoder/memory/hardware limitations) and 3 sessions on all the GeForce cards combined. Thus, the limit on the number of simultaneous encode sessions for such a system is $N + 3$.

1.4. NVENC Performance

With every generation of NVIDIA GPUs (Kepler, Maxwell 1st/2nd gen, Pascal, Volta, Turing and Ampere), NVENC performance has increased steadily. Table 3 provides *indicative*[1] NVENC performance on Kepler, Maxwell, Pascal, Turing and Ampere GPUs for different presets and rate control modes (these two factors play a major role in determining the performance and quality). Note that performance numbers in Table 3 are measured on GeForce hardware with

[1] Encoder performance depends on many factors, including but not limited to: Encoder settings, GPU clocks, GPU type, video content type etc.

assumptions listed under the table. The performance varies across GPU classes (e.g. Quadro, Tesla), and scales (almost) linearly with the clock speeds for each hardware.

While Kepler and first-generation Maxwell GPUs had one NVENC engine per chip, certain variants of the second-generation Maxwell, Pascal and Volta GPUs have two/three NVENC engines per chip. This increases the aggregate encoder performance of the GPU. NVIDIA driver takes care of load balancing among multiple NVENC engines on the chip, so that applications don't require any special code to take advantage of multiple encoders and automatically benefit from higher encoder capacity on higher-end GPU hardware. The encode performance listed in Table 3 is given *per NVENC engine*. Thus, if the GPU has 2 NVENCs (e.g. GP104, GM204), multiply the corresponding number in Table 3 by the number of NVENCs per chip to get aggregate maximum performance (applicable only when running multiple simultaneous encode sessions). Note that performance with single encoding session cannot exceed performance per NVENC, regardless of the number of NVENCs present on the GPU.

NVENC hardware natively supports multiple hardware encoding contexts with negligible context-switching penalty. As a result, subject to the hardware performance limit and available memory, an application can encode multiple videos simultaneously. NVENCODE API exposes several presets, rate control modes and other parameters for programming the hardware. A combination of these parameters enables video encoding at varying quality and performance levels. In general, one can trade performance for quality and vice versa.

Table 3. NVENC encoding performance

Preset	RC Mode	Tuning Info	H.264			HEVC		
			Maxwell (M2000)	Pascal (P2000)	Turing (RTX8000	Maxwell (M2000)	Pascal (P2000)	Turing (RTX8000)
P1	CBR	LL	472	676	766	275	532	849
	VBR	HQ	479	694	749	239	499	837
P3	CBR	LL	454	658	550	228	436	421
	VBR	HQ	292	393	546	226	435	504
P5	CBR	LL	262	361	219	205	364	277
	VBR	HQ	216	323	216	204	363	304
P7	CBR	LL	219	319	194	189	338	277
	VBR	HQ	160	243	180	151	260	154

- Resolution/Input Format/Bit depth: 1920 × 1080/YUV 4:2:0/8-bit
- All the measurement is done on the highest video clocks as reported by nvidia-smi (i.e. 1129 MHz, 1683 MHz, 1755 MHz for M2000, P2000 and RTX8000 respectively). The performance should scale according to the video clocks as reported by nvidia-smi for other GPUs of every individual family. Information on nvidia-smi can be found at https://developer.nvidia.com/nvidia-system-management-interface.
- The encoding performance on Ampere GPUs scales up with the performance numbers on Turing GPUs in proportion to the highest video clocks as reported by nvidia-smi.
- The encoding performance on Volta GPUs scales up with the performance numbers on Pascal GPUs in proportion to the highest video clocks as reported by nvidia-smi.

- Software: Windows 10, Video Codec SDK 11.0, NVIDIA display driver: 456.71
- CBR: Constant bitrate rate control mode, VBR: Variable bitrate rate control mode, LL : Low latency tuning info, HQ: High quality tuning info

1.5. Programming NVENC

Video Codec SDK 11.0 is supported on R455 and above drivers on Windows and Linux. Refer to the SDK release notes for information regarding the required driver version.

Refer to the documents and the sample applications included in the SDK package for details on how to program NVENC.

1.6. FFmpeg Support

FFmpeg is the most popular multimedia transcoding tool used extensively for video and audio transcoding.

The video hardware accelerators in NVIDIA GPUs can be effectively used with FFmpeg to significantly speed up the video decoding, encoding and end-to-end transcoding at very high performance.

Note that FFmpeg is open-source project and its usage is governed by specific licenses and terms and conditions for FFmpeg.

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 acknowledgment, 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.

Trademarks

NVIDIA, the NVIDIA logo, and cuBLAS, CUDA, CUDA Toolkit, cuDNN, DALI, DIGITS, DGX, DGX-1, DGX-2, DGX Station, DLProf, GPU, Jetson, Kepler, Maxwell, NCCL, Nsight Compute, Nsight Systems, NVCaffe, NVIDIA Deep Learning SDK, NVIDIA Developer Program, NVIDIA GPU Cloud, NVLink, NVSHMEM, PerfWorks, Pascal, SDK Manager, Tegra, TensorRT, TensorRT Inference Server, Tesla, TF-TRT, Triton Inference Server, Turing, and Volta are trademarks and/or registered trademarks of NVIDIA Corporation in the United States and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright

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