NVIDIA VIDEO CODEC SDK APPLICATION NOTE - ENCODER

NVENC_DA-6209-001_v13| Aug 2019

Application Note

DOCUMENT CHANGE HISTORY

NVENC_DA-6209-001_v13

Version	Date	Authors	Description of Change
01	Jan 30,2012	AP/CC	Initial release
02	Sept 24, 2012	AP	Update for NVENC SDK 2.0
03	April 10, 2013	AP	Update for Monterey SDK 2.0.0 update
04	Aug 4, 2013	AP	Update for NVENC SDK 3.0
05	June 17, 2014	SM/AP	Update for NVENC SDK 4.0
06	Nov 14, 2014	SM	Update for NVENC SDK 5.0
07	Oct 10, 2015	SM	Update for Video Codec SDK 6.0
08	June 10, 2016	SM	Update for Video Codec SDK 7.0
09	Nov 15, 2016	SM	Update for Video Codec SDK 7.1
10	Apr 11, 2017	SM/AP	Update for Video Codec SDK 8.0
11	Jan 10, 2018	SM	Update for Video Codec SDK 8.1
12	Jan 10, 2019	SM	Update for Video Codec SDK 9.0
13	Aug 10, 2019	SM	Update for Video Codec SDK 9.1

TABLE OF CONTENTS

NVIDIA Hardware Video Encoder 4

1. Introduction 4
2. NVENC Capabilities 4
3. NVENC Licensing Policy 8
4. NVENC Performance 9
5. Programming NVENC 11
6. FFmpeg and Libav Support 11

LIST OF TABLES

Table 1. NVENC hardware capabilities 5
Table 2. What's new in Video Codec SDK 9.0 7
Table 3. What's new in Video Codec SDK 9.1 8
Table 4. NVENC encoding performance 10

NVIDIA HARDWARE VIDEO ENCODER

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.

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.

Table 1. NVENC hardware capabilities

Feature	Description	Kepler GPUs	1st Gen Maxwell GPUs	2nd Gen Maxwell GPUs	Pascal GPUs	Volta and TU117 GPUs	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.	✓	✓	✓	✓	✓	✓
H.264 4:4:4 encoding (only CAVLC)	Capability to encode YUV 4:4:4 sequence and generate a H.264-bit stream.	×	✓	✓	✓	✓	✓
H.264 lossless encoding	Lossless encoding.	×	✓	✓	✓	✓	✓
H.264 motion estimation (ME) only mode	Capability to provide macro-block level motion vectors and intra/inter modes.	×	✓	✓	✓	✓	✓
H.264 field encoding	Capability to encode field content.	✓	✓	✓	✓	✓	×
H.264/HEVC weighted prediction	Support for weighted prediction.	×	×	×	✓	✓	✓
Encoding support for H.264 ARGB content	Capability to encode RGB input.	✓	✓	✓	✓	✓	✓
Multiple reference frames for H.264	Capability to use different reference frames	×	×	×	×	×	✓
HEVC main profile	Capability to encode YUV 4:2:0 sequence and generate a HEVC bit stream.	×	×	✓	✓	✓	✓

Feature	Description	Kepler GPUs	1st Gen Maxwell GPUs	2nd Gen Maxwell GPUs	Pascal GPUs	Volta and TU117 GPUs	Turing GPUs except TU117
HEVC lossless encoding	Lossless encoding.	×	×	×	✓	✓	✓
HEVC main10 profile	Support for encoding 10-bit content generate a HEVC bit stream.	×	×	×	✓	✓	✓
HEVC 4:4:4 encoding	Capability to encode YUV 4:4:4 sequence and generate a HEVC bit stream.	×	×	×	✓	✓	✓
HEVC motion estimation (ME) only mode	Capability to provide CTB level motion vectors and intra/inter modes.	×	×	×	✓	✓	✓
HEVC 8K encoding	Support for encoding 8192 × 8192 Content.	×	×	×	✓*	✓	✓
HEVC sample adaptive offset (SAO)	Improves encoded video quality.	×	×	×	✓	✓	✓
HEVC B frame	Improves encoded quality	×	×	×	×	×	✓
Multiple reference frames for HEVC	Capability to use different reference frames	×	×	×	×	×	✓

*: Supported in select Pascal generation GPUs

Table 2. What's new in Video Codec SDK 9.0

Feature	Description
Improved encoded quality for Turing GPUs	Turing hardware adds support for features like rate distortion optimization (RDO) and enable multiple frames to be used as reference. These features significantly improve the encoding quality for both H.264 and HEVC. These features are tied with the already existing presets. This ensures that existing applications can take advantage of these features without making changes to their source code.
HEVC B frame	The support for HEVC B frame is added in Turing GPUs. The SDK 9.0 adds HEVC B frame support for Turing GPUs.
Encoded bitstream in video memory	This feature enables the clients to have the NVENC output the encoded bitstream in video memory. The feature is supported for both HEVC and H.264. This avoids overhead of copying from system to video memory for date pipelines operating on video memory.
H.264 ME-only mode output in video memory.	This feature enables the clients to have the NVENC output the H.264 motion vectors (for H.264 ME-only mode) in video memory. This avoids overhead of copying from system to video memory for date pipelines operating on video memory.
Non-reference P frames	This provides client the capability to mark a P frame to be *not* used as reference. This can help prevent error propagation in noisy transmission channels.
Support for accepting CUArray as input	This feature enables to clients to send all the input formats supported by NVENCODE API as a CUArray.
Sample application demonstrating encoding of Vulkan surfaces.	A sample application has been added which illustrates encoding of a Vulkan surface using NVENCODE API on Linux.

Table 3. What's new in Video Codec SDK 9.1

Feature	Description
NVENCODE API for retrieving the last encountered error.	A new NVENCODE API has been added for error reporting. This API will be useful for debugging and trouble shooting.
Support for CUStream	NVENCODE API internally uses CUDA kernels for doing certain preprocessing and postprocessing. Support for CUStream has been added in NVENCODE API to enable execution of preprocessing and postprocessing CUDA kernels on separate client specified CUDA streams instead of default NULL stream. This results in better pipelining and improved throughput when NVENCODE API is used along with CUDA operations.
Filler NALU insertion	This feature enables clients to insert filler NALUs in the bitstream to meet the target bit rate in constant bit rate (CBR) rate control modes. This is useful in scenarios where it is mandatory to adhere to the specified bitrate and NVENC is generating a lower bitrate than target.
Multiple reference frames	Turing NVENC adds support for choosing the matching macroblock/CTB from multiple reference frames, which results to improvement to encoded quality. The numbers of reference frames are decided inside NVIDIA's display driver. The current SDK exposes control to the client for specifying the number of reference frames which will override the values set inside NVIDIA's display driver.
Fixes for H.264 MVC	Bug-fixes and API enhancement to support H.264 MVC encoding.

3. NVENC LICENSING POLICY

There is no change in licensing policy in the current SDK in comparison to the earlier SDK(s). 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 2 per system. This limit of 2 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 K4000 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 K4000 card (where *N* is defined by the encoder/memory/hardware limitations) and two sessions on all the three GeForce cards combined. Thus, the limit on the number of simultaneous encode sessions for such a system is *N* + 2.

4. NVENC PERFORMANCE

With every generation of NVIDIA GPUs (Kepler, Maxwell 1st/2nd gen, Pascal, Volta, and Turing), NVENC performance has increased steadily. Table 4 provides *indicative*[1] NVENC performance on Kepler, Maxwell, Pascal and Turing 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 4 are measured on GeForce hardware with 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 4 is given *per NVENC engine*. Thus, if the GPU has 2 NVENCs (e.g. GP104, GM204), multiply the corresponding number in Table 4 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

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

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 4. NVENC encoding performance

		H.264 (FPS)				HEVC (FPS)		
Preset	**RC Mode***	**Kepler (K2000)**	**2nd Gen Maxwell (M2000)**	**Pascal (P2000)**	**Turing (RTX8000)**	**2nd Gen Maxwell (M2000)**	**Pascal (P2000)**	**Turing (RTX8000)**
High Performance	Single Pass	215	471	695	719	218	412	810
	Dual Pass	112	375	556	571	179	340	640
High Quality	Single Pass	80	260	365	423	150	259	159
	Dual Pass	59	295	432	306	128	227	132
Low latency High Performance	Single Pass	135	366	528	695	218	412	496
	Dual Pass	86	322	484	557	179	340	423
Low latency High Quality	Single Pass	80	260	361	418	217	410	328
	Dual Pass	58	302	444	397	178	338	304
Lossless			333	470	429		244	277

- 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. 540 MHz, 1129 MHz, 1683 MHz, 1755 MHz for K2000, 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 here.
- Software: Windows 10, Video Codec SDK 9.1, NVIDIA display driver: 436.15
- 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.
- Please note, some of the numbers may look slightly different from the earlier SDKs as the content used for evaluation is different.

5. PROGRAMMING NVENC

Video Codec SDK 9.0 and Video Codec SDK 9.1 are supported on R418 and R435 drivers and above respectively. 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.

6. FFMPEG AND LIBAV SUPPORT

FFmpeg and Libav are the most popular multimedia transcoding tools used extensively for video and audio transcoding.

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

Note that FFmpeg and Libav are open-source projects and their usage is governed by specific licenses and terms and conditions for each of these projects.

Notice

Trademarks

Copyright