NVIDIA VIDEO TECHNOLOGIES

Abhijit Patait, 5/8/2017

NVIDIA

AGENDA

NVIDIA Video Technologies

New SDK Release

Major Focus Areas

Video SDK Features

Software Flow

FFmpeg Performance and Benchmarking Tips

Benchmarks

NVIDIA VIDEO TECHNOLOGIES

NVIDIA. DESIGNWORKS™

VIDEO CODEC SDK

A comprehensive set of APIs for GPU-accelerated Video Encode and Decode

The SDK consists of two hardware acceleration interfaces:

NVENCODE API for video encode acceleration

NVDECODE API for video decode acceleration (formerly called NVCUVID API)

Independent of CUDA/3D cores on GPU

NVIDIA Video Codec SDK technology is used to stream video with NVIDIA ShadowPlay running on NVIDIA GPUs

NVIDIA VIDEO TECHNOLOGIES

SOFTWARE

FFMPEG & LIBAV

Easy access to NVIDIA GPU hardware acceleration

FFMPEG libav

VIDEO CODEC SDK

NVIDIA. DESIGNWORKS

A comprehensive set of APIs for GPU-accelerated Video Encode and Decode for Windows and Linux
CUDA, DirectX, OpenGL interoperability

NVIDIA DRIVER

HARDWARE

NVENC

Independent Hardware Encoder Function

H.264 MPEG-4/AVC H.265 HEVC

NVDEC

Independent Hardware Decoder Function

H.264 MPEG-4/AVC H.265 HEVC MPEG2 VP8 VP9

NVIDIA VIDEO TECHNOLOGIES

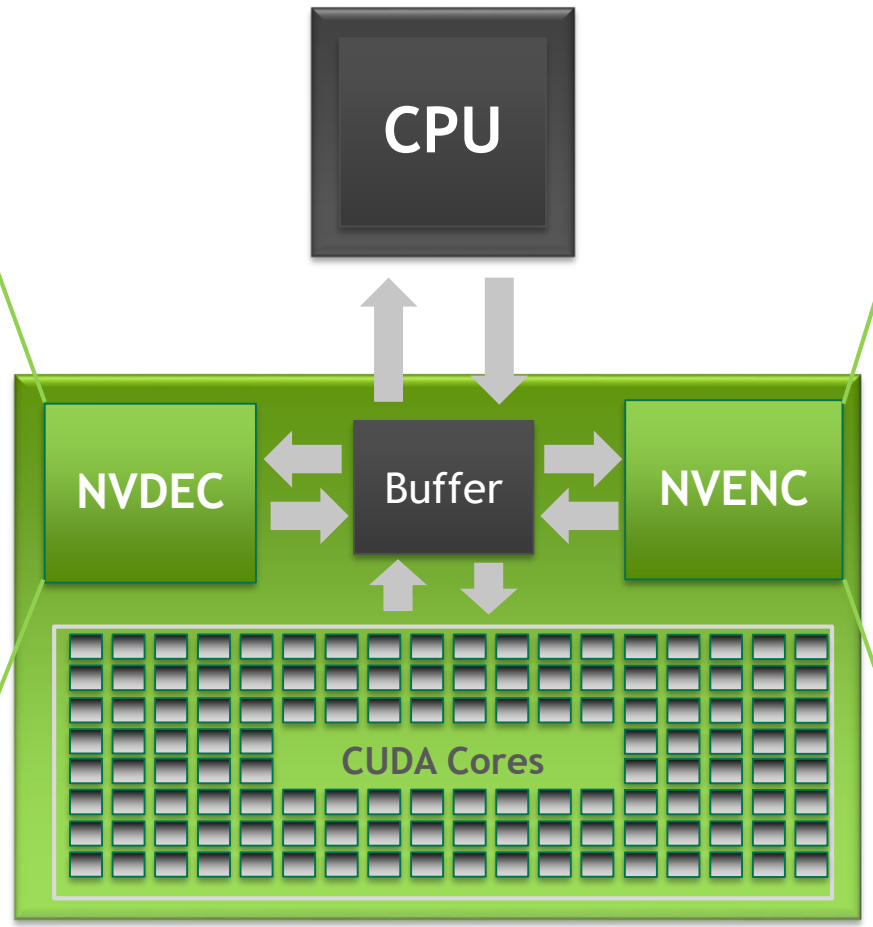
Decode HW*

Formats:
- MPEG-2
- VC1
- VP8
- VP9
- H.264
- H.265
- Lossless

Bit depth:
- 8 bit
- 10 bit

Color**
- YUV 4:2:0

Resolution
- Up to 8K***



Encode HW*

Formats:
- H.264
- H.265
- Lossless

Bit depth:
- 8 bit
- 10 bit

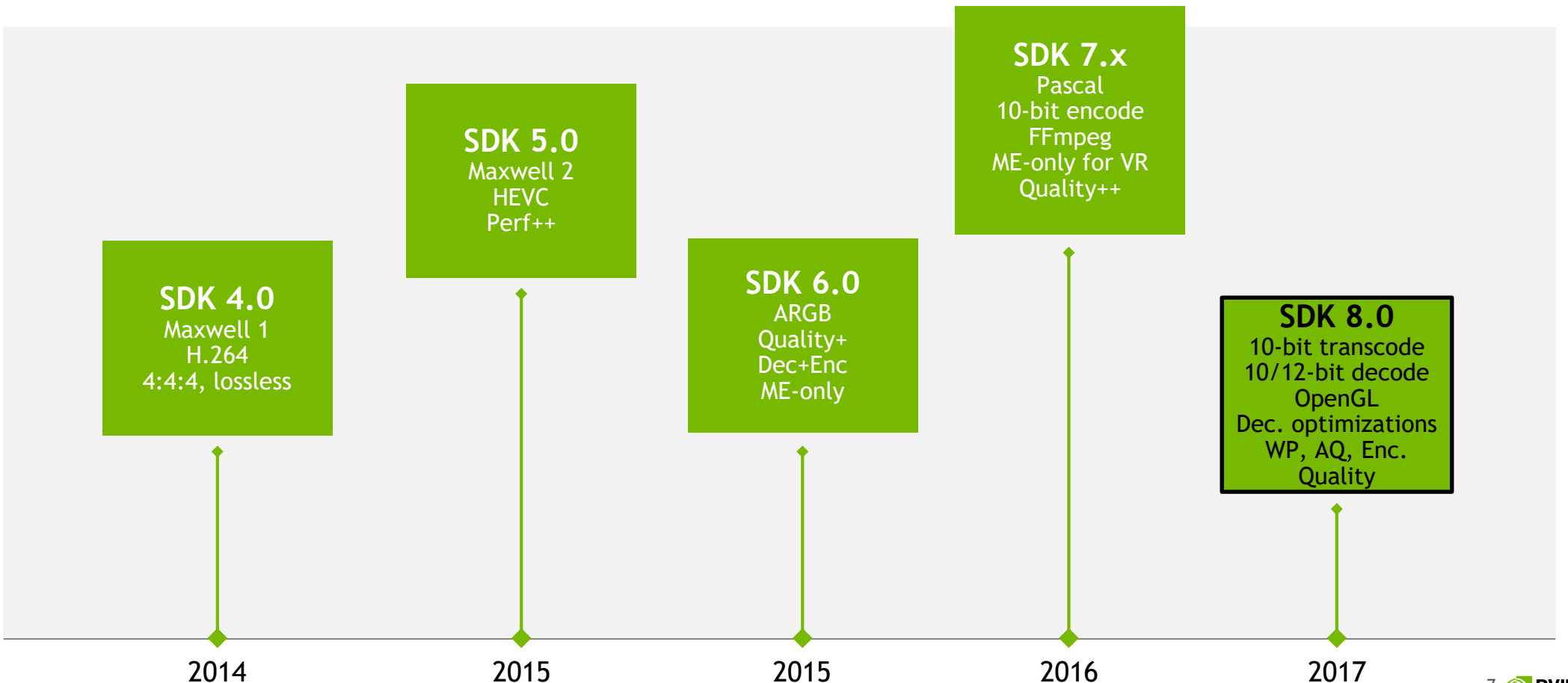
Color**
- YUV 4:4:4
- YUV 4:2:0

Resolution
- Up to 8K***

** See support diagram for previous NVIDIA HW generations*
*** 4:2:2 is not natively supported on HW*
**** Support is codec dependent*

VIDEO SDK EVOLUTION

Video SDK 8.0

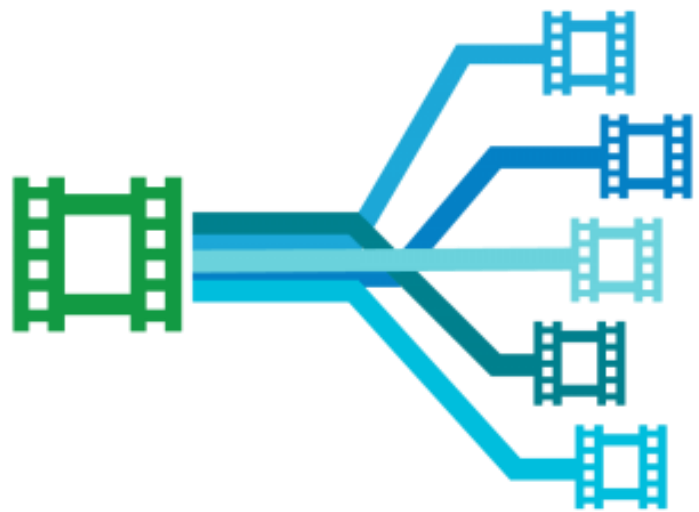


MAJOR FOCUS AREAS

VIDEO TRANSCODING

Performance/Watt

- Content variety
- Codecs, resolutions, quality, bitrate
- Live, VOD, ultra-low-latency, broadcast, archives
- Pre-encoded or encoded-on-demand
- Performance/Watt



GAME/APP STREAMING

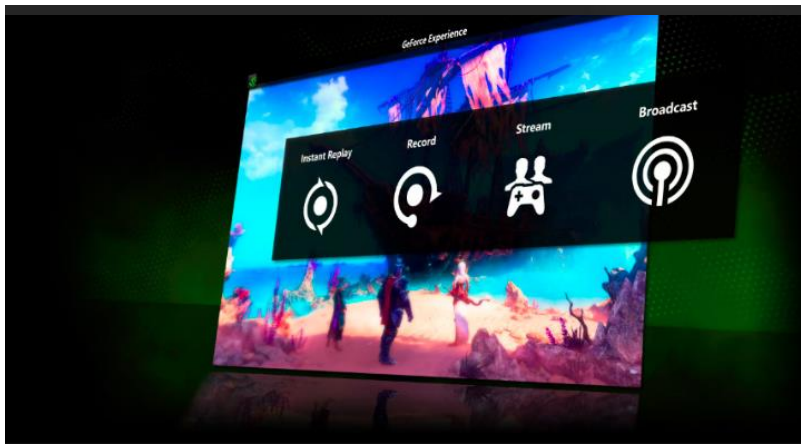
Ultra-low-latency

Stream

- Interactive, single frame latency
- Capture: NvFBC, Encode: NvENC, Decode: NvDEC
- 4K, HDR

Record, Broadcast

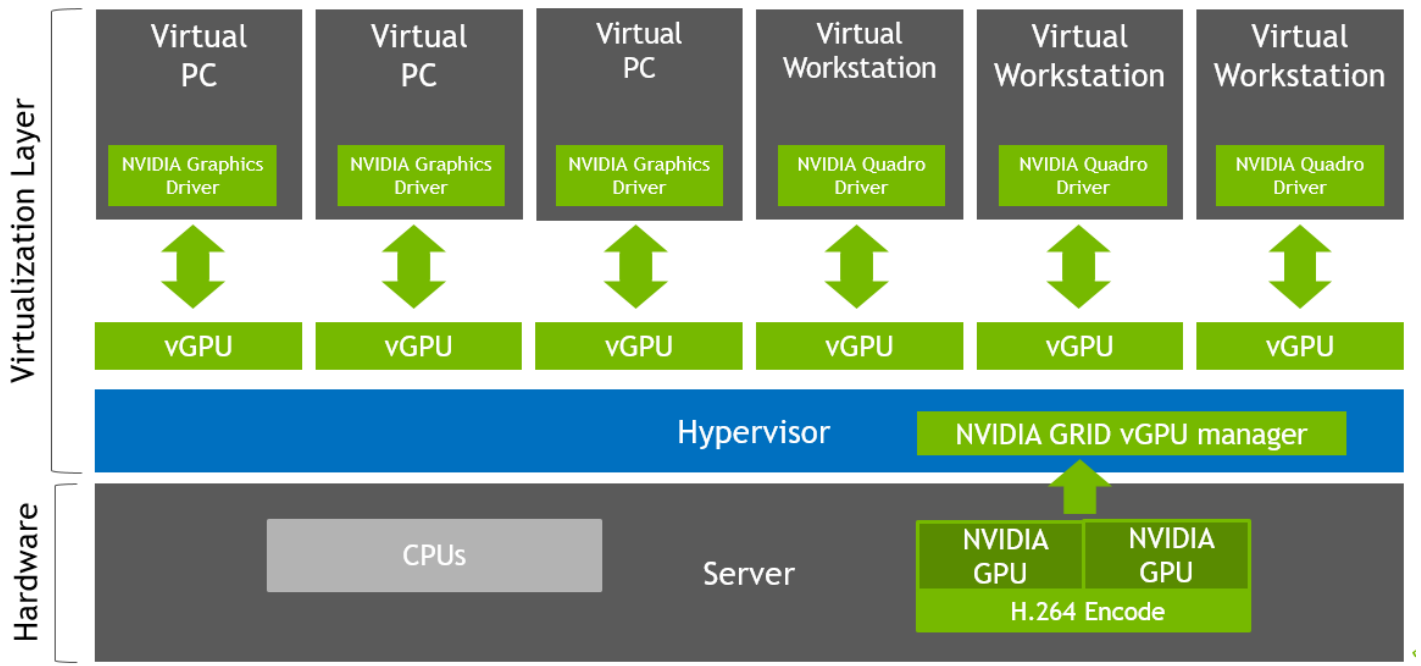
- Quality



GPU VIRTUALIZATION

Quality & reliability

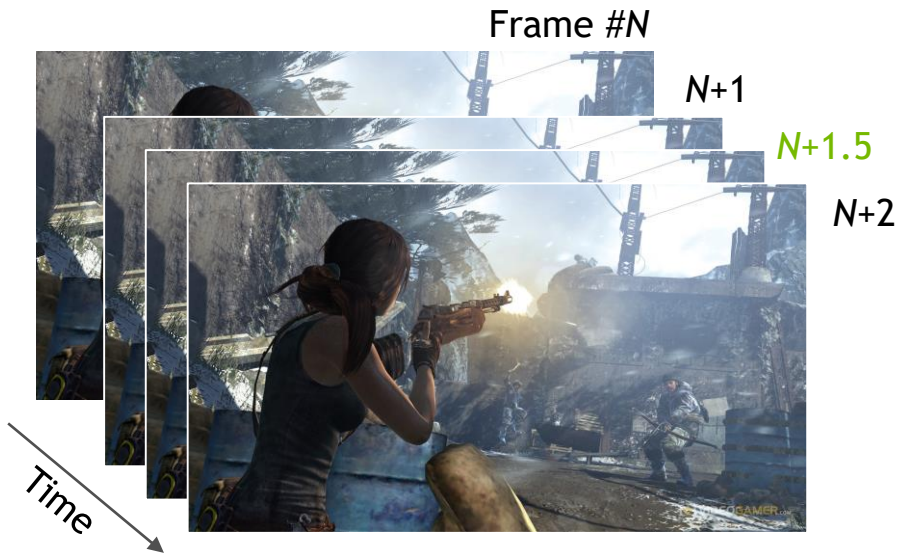
- Capture + encode
- Low-latency
- H.264, HEVC
- 4:2:0, 4:4:4, lossless
- Multiple-displays



MOTION-ESTIMATION ONLY MODE

Accuracy

- Video frame interpolation
- Camera stitching (mono to stereo)
- Camera stabilization
- Computer vision



Frame #(*N*+1.5) is interpolated based on motion vectors between frame #*N* and frame #(*N*+1)

VIDEO SDK FEATURES

ENCODE FEATURES (1/2)

H.264	HEVC	Use-case
Base, Main, High	Main, Main10	Baseline standards
8-bit	8-bit, 10-bit	10-bit for HDR
B-frames	No B-frames	Higher compression & quality
Up to 4096 × 4096	Up to 8192 × 8192	High-res
YUV 4:2:0, 4:4:4		Subsampled or full-res chroma (e.g. wireframes)
Lossless		High-quality archiving
Error resiliency: Intra refresh, LTR, ref-pic invalidation		Handle streaming bit errors

ENCODE FEATURES (2/2)

H.264	HEVC	Use-case
Rate control modes:1-pass, 2-pass		Quality vs performance
Look-ahead		Efficient bit distribution across GOP; higher quality
Adaptive quantization, ΔQP		Finer quality control
Weighted prediction (SDK 8.0)		Fade-in/fade-out, explosion
RGB inputs		Direct NVFBC interoperability
ME-only mode, **MV-hints (SDK 8.0)**		Motion stabilization, Optical flow for VR stereo stitching, Frame interpolation
1-3 NVENCs per chip		High throughput
CUDA, DX, **OGL (Linux) (SDK 8.0)**		Easy integration

DECODE FEATURES

Feature	Use-case
MPEG2, VC-1, MPEG-4, H.264, HEVC, VP8, VP9	Baseline standards
8-bit (all codecs), **10/12 bit (HEVC, VP9) (SDK 8.0)**	HDR decoding
Up to 8192 × 8192 for HEVC, 4096 × 4096 for H.264	High-res
Error resiliency and concealment	Internet streaming

VIDEO SDK – CONTENTS (1/2)

- Header, documentation, sample applications
- Binaries (.dll, .so) in NVIDIA display driver
- Unified API for Windows & Linux
- NVIDIA developer zone
- Encode limitations
  - Unconstrained: Tesla, GRID, Quadro ≥ *X*2000 (*X* = K, M, P)
  - 2 sessions/system: GeForce, Quadro < *X*2000
- No decode limitations

VIDEO SDK – CONTENTS (2/2)

Sample Applications

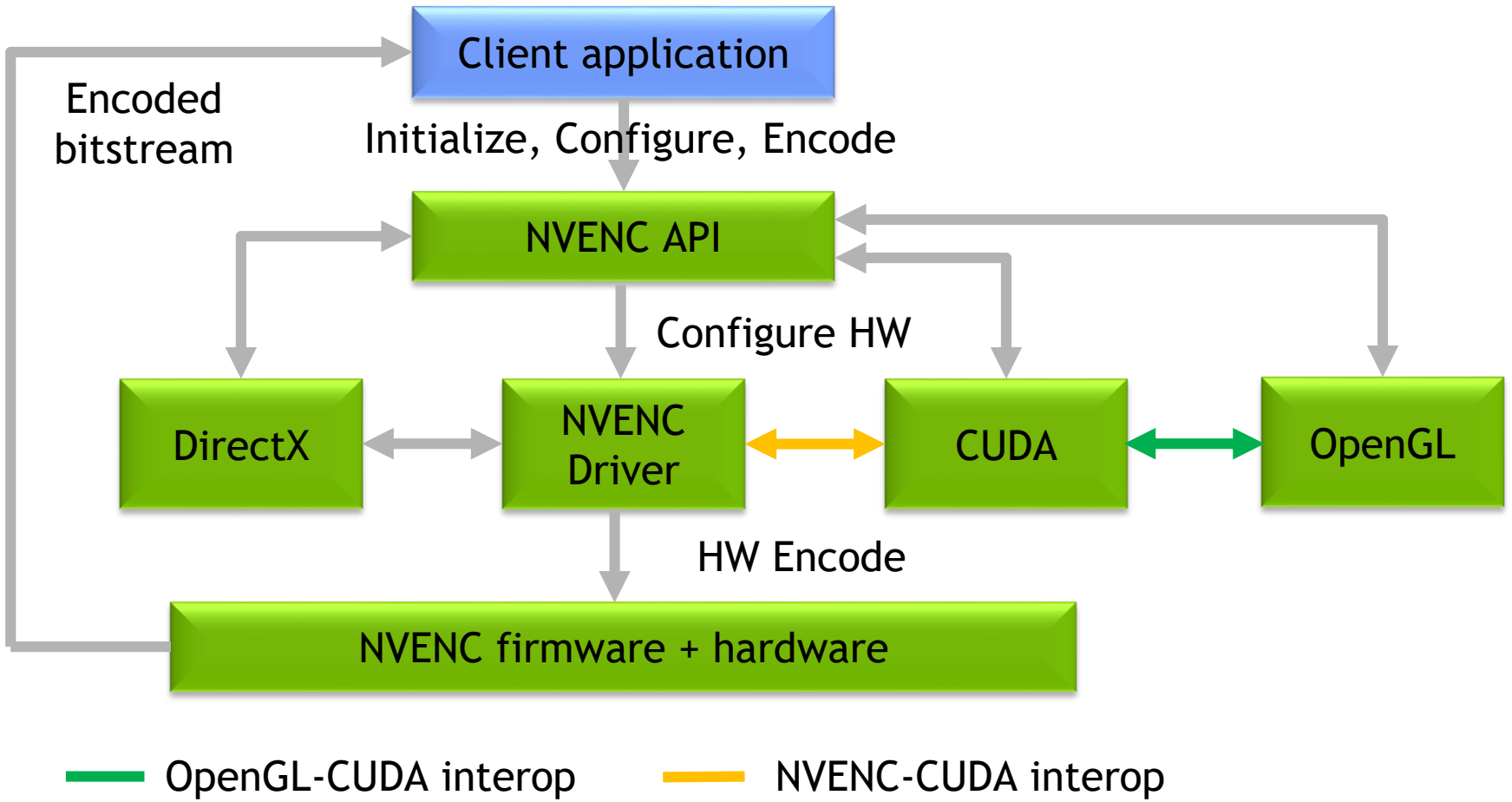
- Decode: DX9, DX11, CUDA, OpenGL
- Encode: Basic functionality, features (NvEncoder)
- Encode: Performance (NvEnodePerf)
- Encode: CUDA interop, D3D interop, OGL interop,
- Encode: Low-latency (NVEncoderLowLatency)
- Transcode (NvTranscoder)
- Coming soon: Reusable classes

FFMPEG/LIBAV

- Major SW focus area for past 6 months
- Feature parity with Video SDK 7.1, SDK 8.0 post GTC
- End-to-end FFmpeg transcoding @ best possible quality & perf

SOFTWARE FLOW

ENCODE APP FLOW



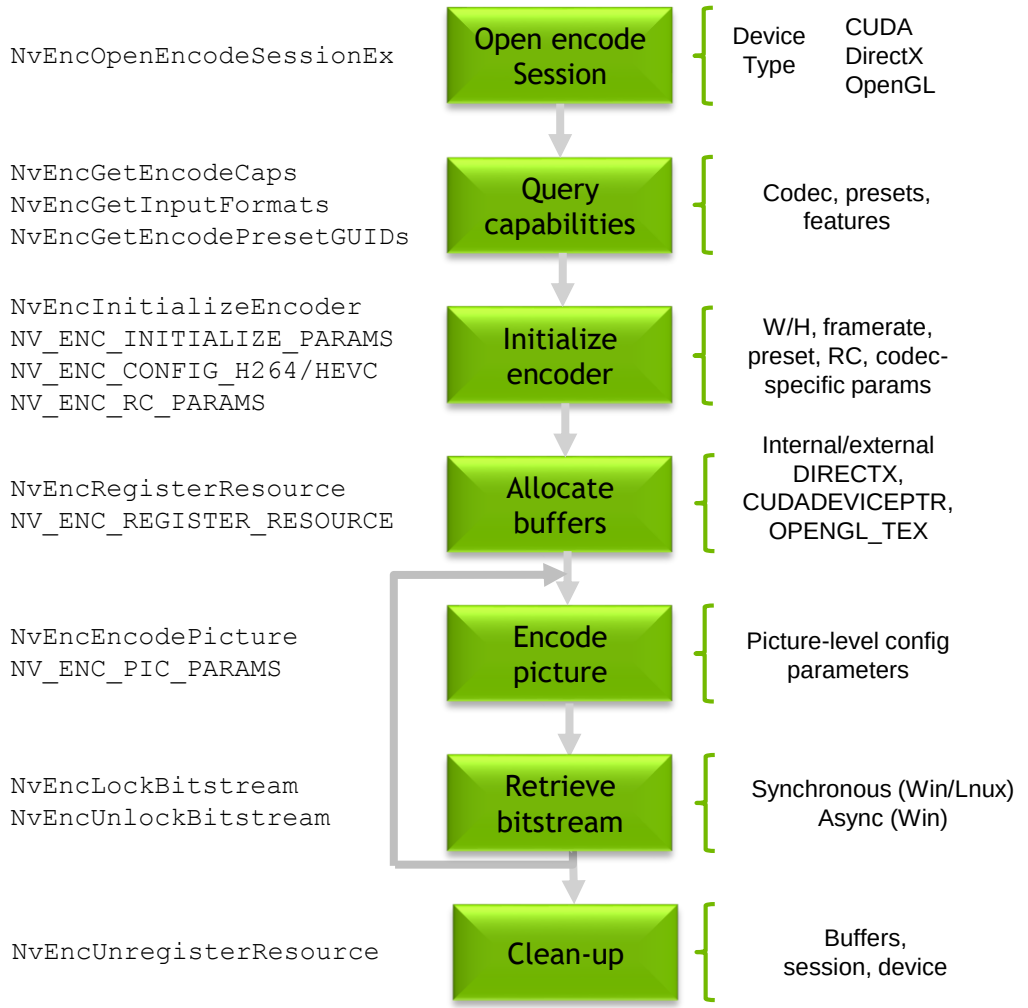
ENCODE APP FLOW

APIs

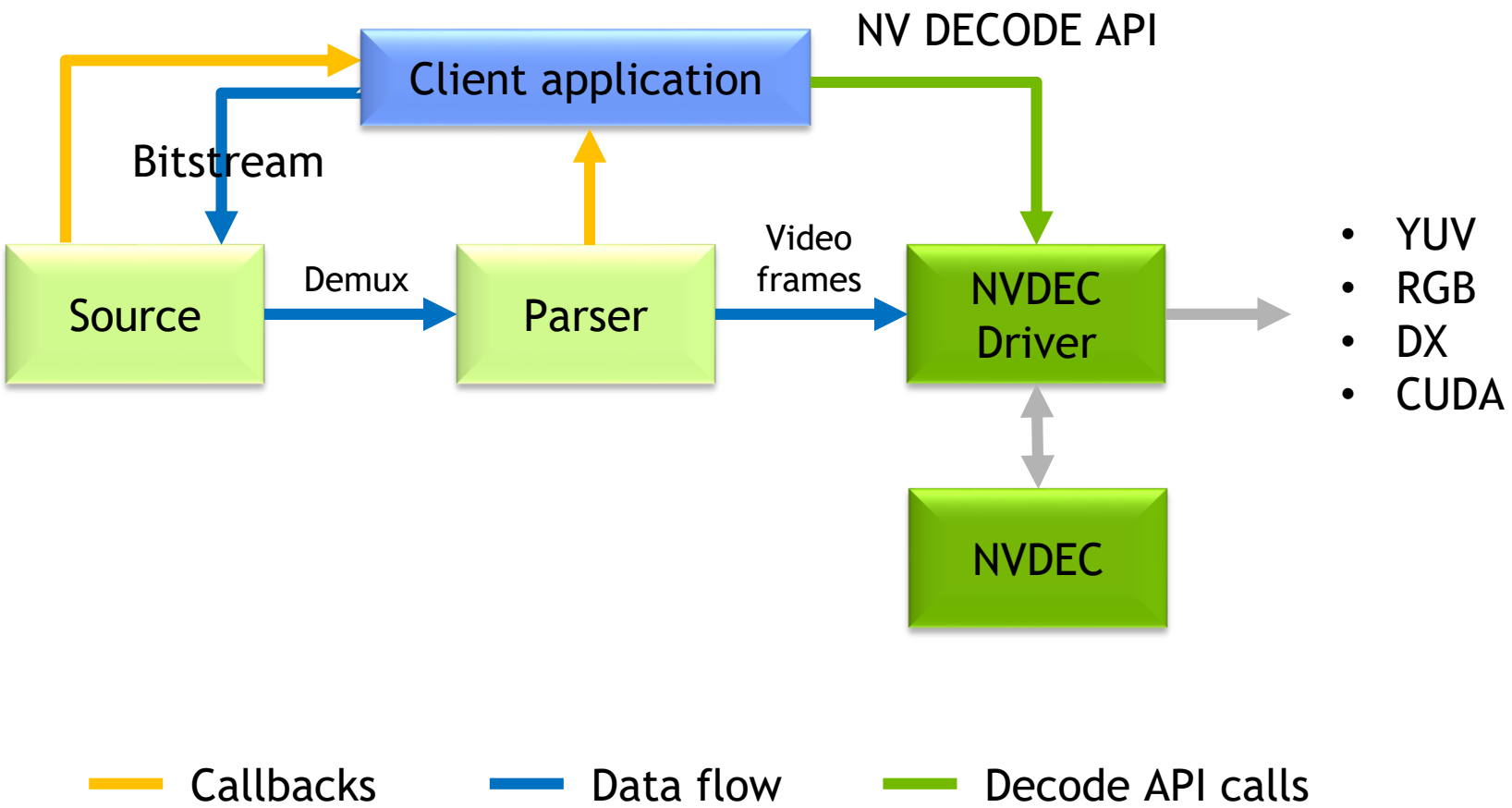
API Functions

Structures

Defined in nvEncodeAPI.h



DECODE APP FLOW



DECODE APP FLOW

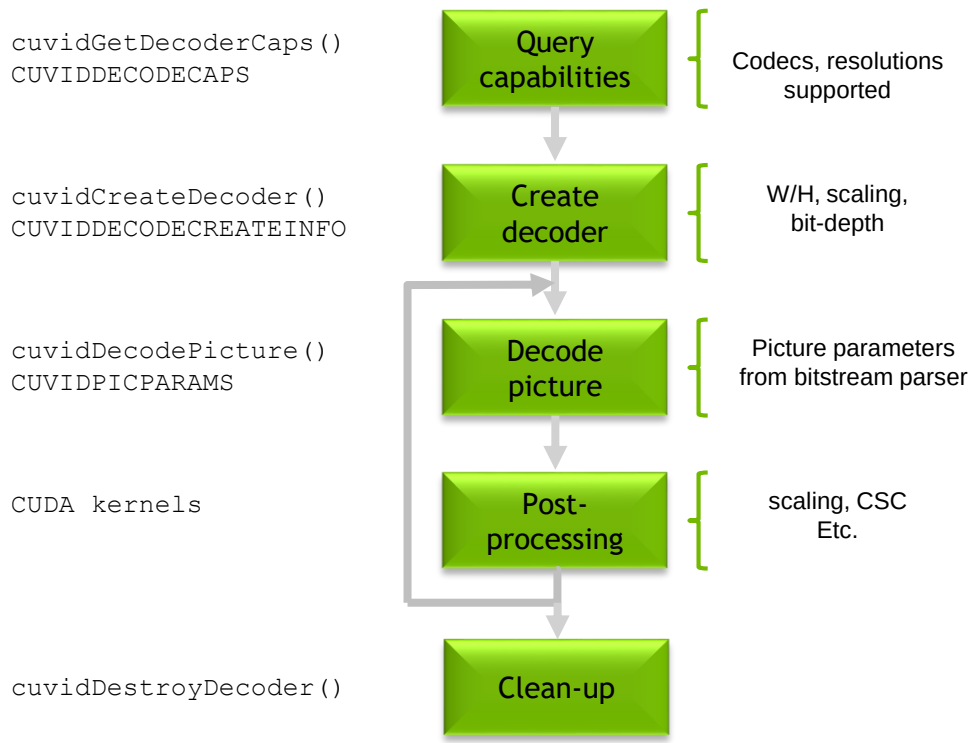
APIs

API functions

Structures

Defined in dynlink_nvcuvid.h,

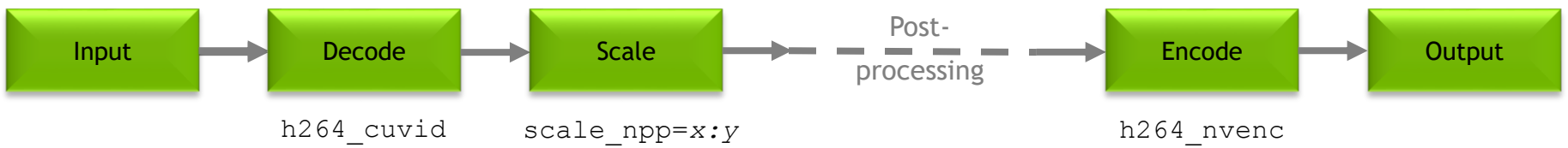
dynlink_cuviddec.h



FFMPEG APP FLOW

```
ffmpeg -y -vsync 0 -hwaccel cuvid -c:v h264_cuvid -i input.mp4 -c:a copy -vf scale_npp=1280:720 -c:v
h264_nvenc -b:v 5M output.mp4
```

- Chain of filters



- `-hwaccel cuvid`: Use end-to-end NVIDIA hardware acceleration
- `h264_cuvid`: Use NVCUVID/NVDECODE
- `h264_nvenc`: Use NVENCODE
- `scale_npp`: high-perf CUDA scaling

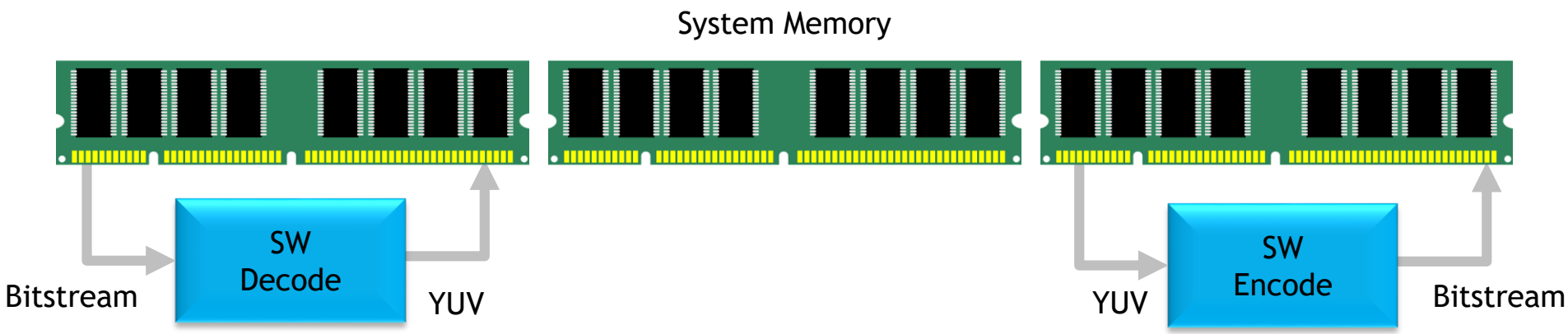
HARDWARE ACCELERATED TRANSCODE USING FFMPEG

PERFORMANCE CONSIDERATIONS - FFMPEG

- Minimize memory (PCIe) transfers
- Saturate on-chip encoder/decoder
- Efficient *M*:*N* command line
- Minimize I/O
- Encode settings
- GPU Clocks

SW TRANSCODE

```
ffmpeg -c:v h264 -i input.mp4 -c:a copy -c:v h264 -b:v 5M output.mp4
```

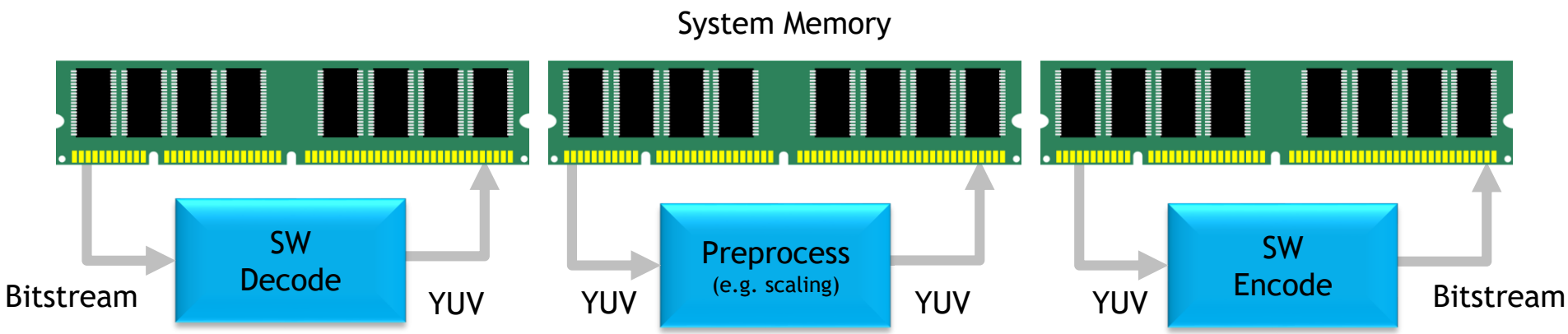


32 fps*

**1:2 transcode, fps per session*
4 GHz Intel i7-6700K

SW TRANSCODE + SCALE

```
ffmpeg -c:v h264 -i input.mp4 -vf scale=1280:720 -c:a copy -c:v h264 -b:v 5M output.mp4
```

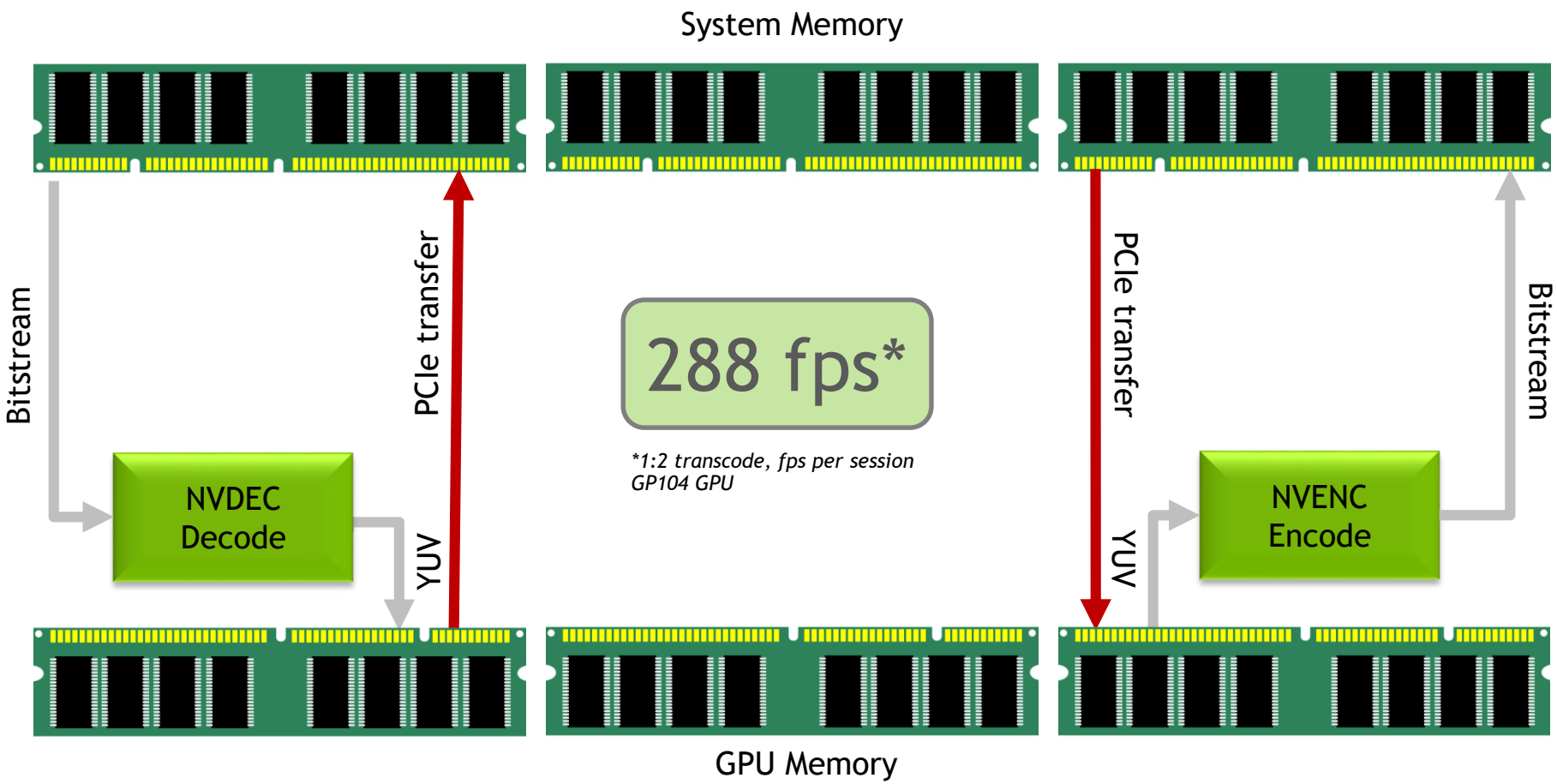


29 fps*

**1:2 transcode, fps per session*
4 GHz Intel i7-6700K

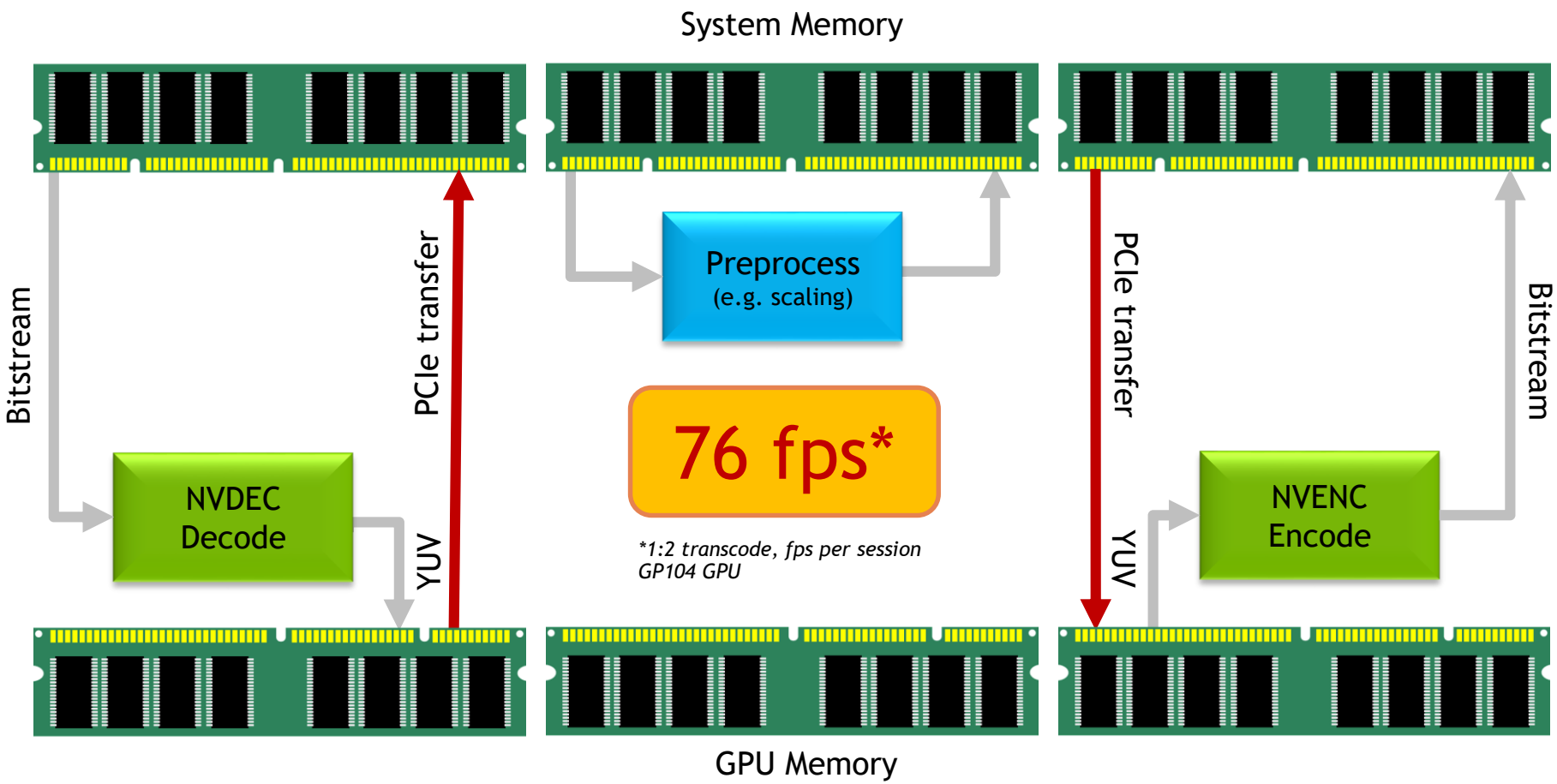
GPU UNOPTIMIZED TRANSCODE

```
ffmpeg -y -vsync 0 -c:v h264_cuvid -i input.mp4 -c:a copy -c:v h264_nvenc -b:v 5M output.mp4
```



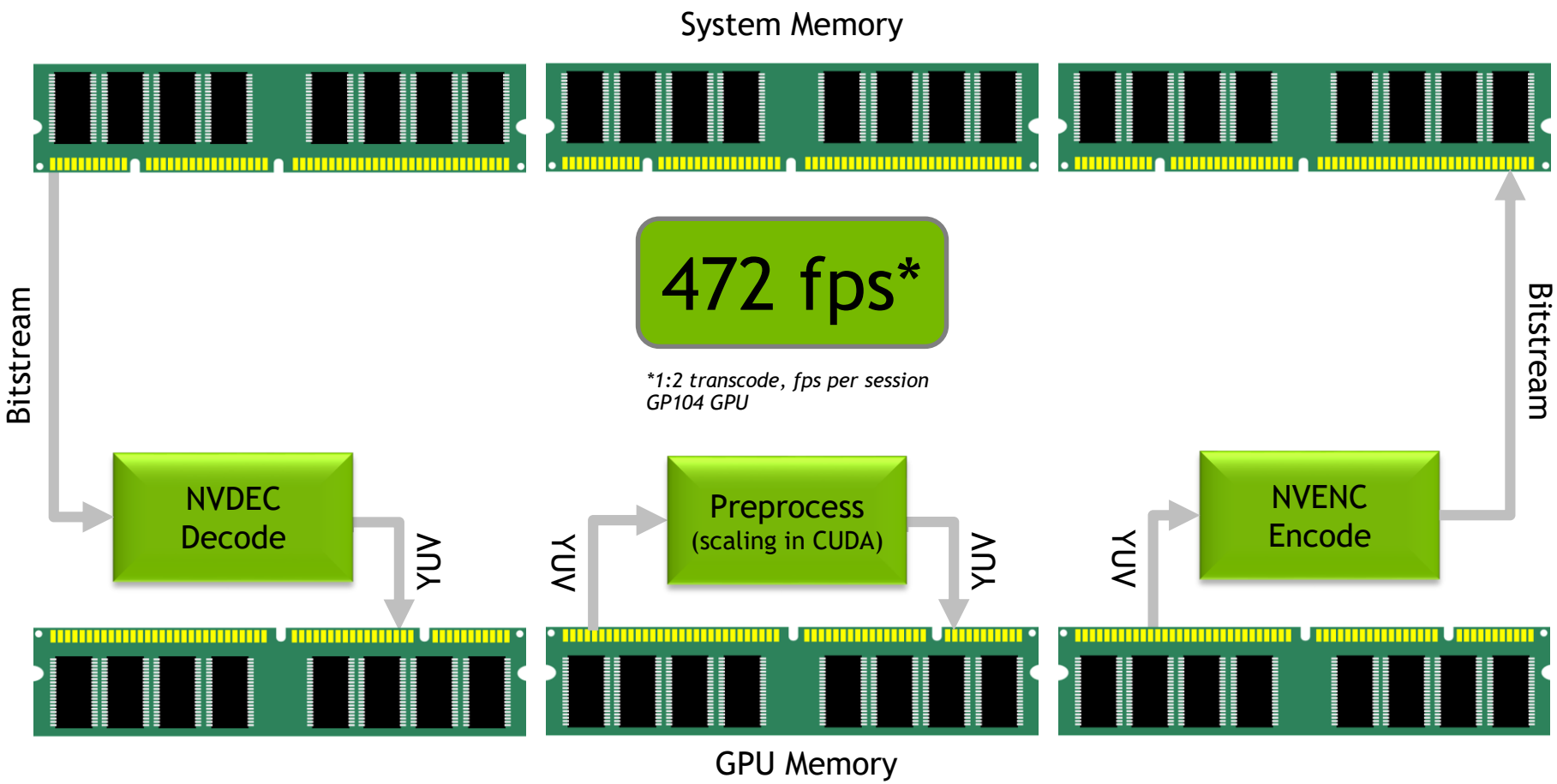
GPU UNOPTIMIZED TRANSCODE + CPU SCALE

```
ffmpeg -y -vsync 0 -c:v h264_cuvid -i input.mp4 -c:a copy -vf scale=1280:720 -c:v h264_nvenc -b:v 5M output.mp4
```



HIGH-PERF GPU OPTIMIZED TRANSCODE

```
ffmpeg -y -vsync 0 -hwaccel cuvid -c:v h264_cuvid -i input.mp4 -c:a copy -vf scale_npp=1280:720 -c:v h264_nvenc -b:v 5M output.mp4
```



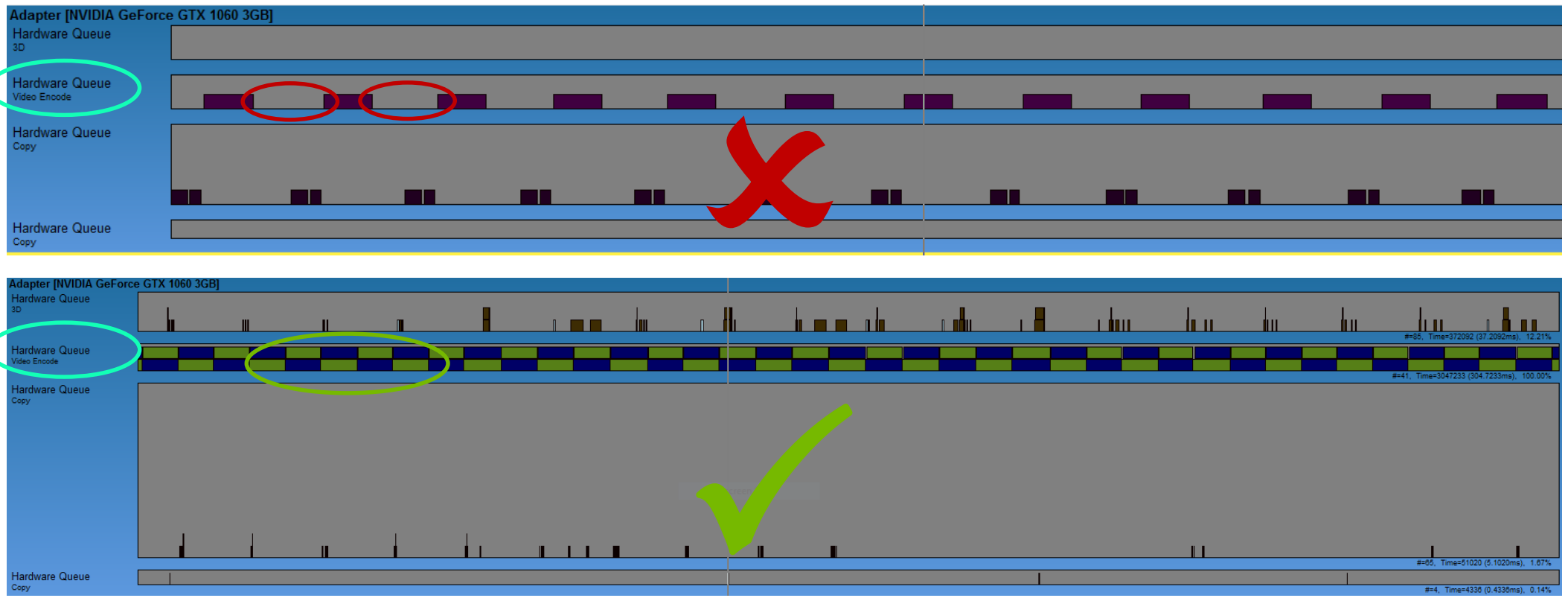
PERFORMANCE CONSIDERATIONS

Saturating encoder/decoder

- Pipelining
- Input/output buffers
- Tools: nvidia-smi, Microsoft GPUView

ANALYZING PERFORMANCE BOTTLENECKS

Microsoft GPUView (Windows only)



ANALYZING PERFORMANCE BOTTLENECKS

nvidia-smi (Windows & Linux)

test@kashq: ~

```
test@kashq:~$ nvidia-smi dmon
# gpu    pwr   temp     sm    mem    enc    dec   mclk   pclk
# Idx      W      C      %      %      %      %    MHz    MHz
    0      5     44      0      4      0      0    405    139
    0      5     44      0      3      0      0    405    139
    0      5     44      1      4      0      0    405    139
    0     24     45      5      3     14     12   3802   1506
    0     33     47     14      9     44     36   3802   1885
    0     32     47     13      9     42     38   3802   1885
    0     33     48     15     10     54     41   3802   1885
    0     32     46     13      9     44     38   3802   1885
    0     33     48     15      9     44     36   3802   1885
    0     33     48     15     10     55     43   3802   1885
```

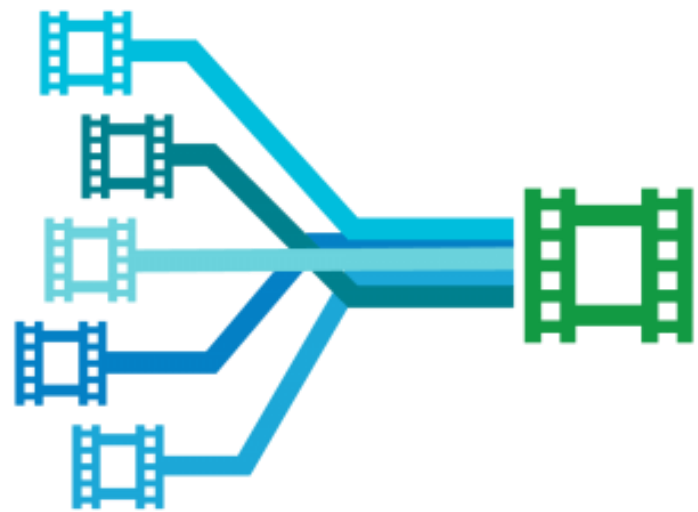
test@kashq: ~

```
test@kashq:~$ nvidia-smi dmon
# gpu    pwr   temp     sm    mem    enc    dec   mclk   pclk
# Idx      W      C      %      %      %      %    MHz    MHz
    0     38     55     41     20    100     80   3802   1873
    0     39     55     40     20    100     79   3802   1873
    0     39     56     44     21     99     91   3802   1873
    0     40     54     41     22     99    100   3802   1873
    0     40     56     41     22     99    100   3802   1873
```

PARALLEL TRANSCODES (*1:N*)

Single command line

```
ffmpeg -y -vsync 0 -hwaccel cuvid -c:v h264_cuvid -i input.mp4
-vf scale_npp=1920:1080 -c:a copy -c:v h264_nvenc -b:v 8M output_1080p.mp4
-vf scale_npp=1280:720 -c:a copy -c:v h264_nvenc -b:v 5M output_720p.mp4
-vf scale_npp=640:480 -c:a copy -c:v h264_nvenc -b:v 3M output_480p.mp4
-vf scale_npp=320:240 -c:a copy -c:v h264_nvenc -b:v 2M output_240p.mp4
-vf scale_npp=160:128 -c:a copy -c:v h264_nvenc -b:v 1M output_128p.mp4
• • •
```

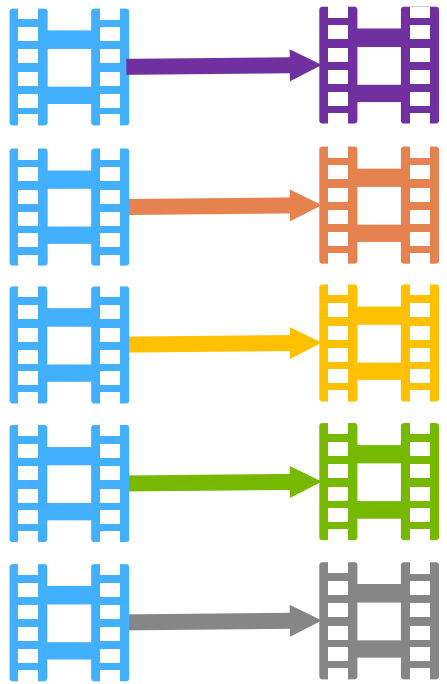


PARALLEL TRANSCODES (*1:N*)

Multiple command lines

```
ffmpeg -y -vsync 0 -hwaccel cuvid -c:v h264_cuvid -i input.mp4 -vf
scale_npp=1920:1080 -c:a copy -c:v h264_nvenc -b:v 5M output1.mp4

ffmpeg -y -vsync 0 -hwaccel cuvid -c:v h264_cuvid -i input.mp4 -vf
scale_npp=1280:720 -c:a copy -c:v h264_nvenc -b:v 5M output2.mp4

ffmpeg -y -vsync 0 -hwaccel cuvid -c:v h264_cuvid -i input.mp4 -vf
scale_npp=640:480 -c:a copy -c:v h264_nvenc -b:v 5M output3.mp4

ffmpeg -y -vsync 0 -hwaccel cuvid -c:v h264_cuvid -i input.mp4 -vf
scale_npp=320:240 -c:a copy -c:v h264_nvenc -b:v 5M output4.mp4

ffmpeg -y -vsync 0 -hwaccel cuvid -c:v h264_cuvid -i input.mp4 -vf
scale_npp=160:128 -c:a copy -c:v h264_nvenc -b:v 5M output5.mp4
```

• • •



PARALLEL TRANSCODES (*1:N*)

Single command line

PROS

Low init time per transcode (amortized)

Minimize memory transfers

Leverage high encoder perf

Low memory overhead

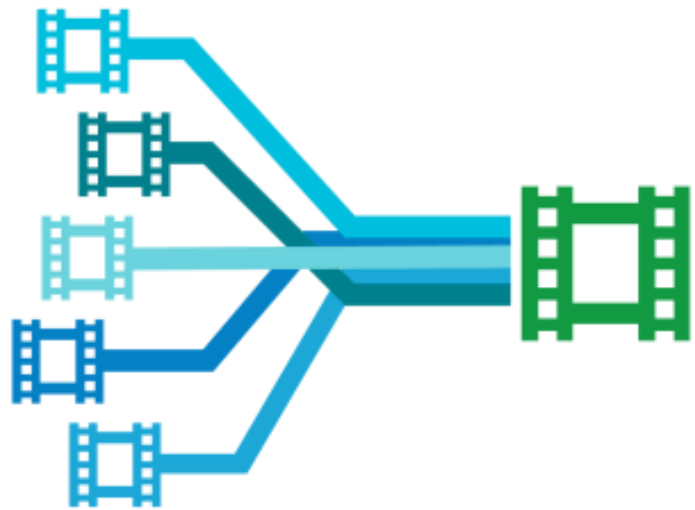
CONS

Complex command line

1:N use-case only

Unsuitable for 1:1 VOD

Typically encoder-limited



PARALLEL TRANSCODES (*1:N*)

Multiple command lines

PROS

Simple command line

Easy scripting

Use-case: 1:1 VOD

CONS

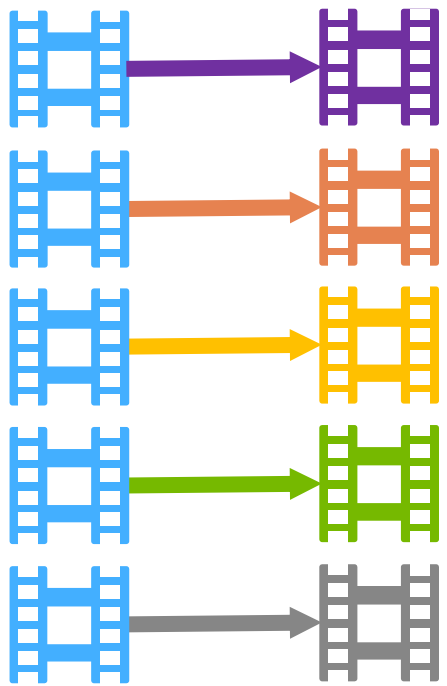
High init time per transcode

High memory overhead

Process-level scheduling optimizations

Typically decoder-limited

Multiple disk I/O for input



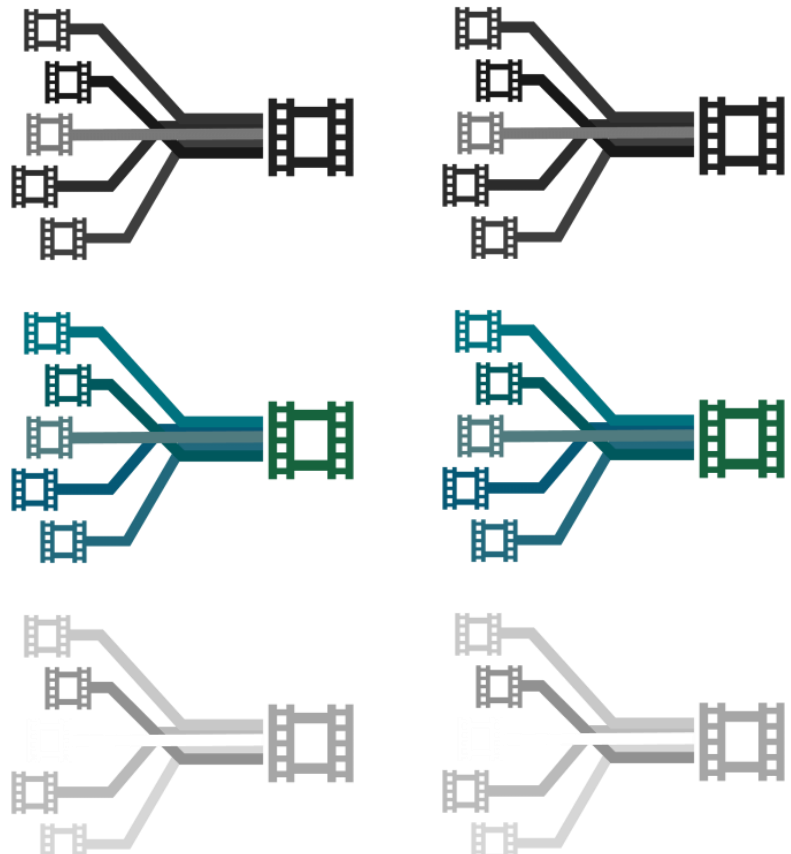
PARALLEL TRANSCODES (*M:N*)

Hybrid approach

Most flexible approach

Balance memory utilization/complexity/perf

Maximum utilization of encode/decode capacity



ENCODE SETTINGS

	Highest Quality	Minimum Delay	Highest Performance
Use-case	Transcoding, Archiving, Broadcast streaming (w/ latency), surveillance	Game & app streaming, surveillance	All, w/ high performance requirement
NVENC API preset to use	High quality (HQ) presets	Low-latency (Low delay) presets	High performance (HP) presets
Latency (set by the application via VBV buffer size)	Depends on what application sets; Typically > 8-10 frames	Depends on what application sets; Typically 1 frame	Depends on what application sets
PSNR delta (0 = High quality)*	0 dB (reference)	Approx. -0.5 dB	Approx. -0.5-2 dB
Advanced features typically used	Look-ahead, B-frames (H.264 only), adaptive B-frames (H.264 only), AQ (Adaptive Quantization)	Strict frame-size compliance low VBV (Video Buffering Verifier), AQ (Adaptive Quantization)	
Motion search and mode	High, 2-pass	Medium, 1-pass/2-pass	Low, 1-pass
Modes	All high quality modes	Most modes	Most modes
Entropy coding	CABAC (H.264)	CABAC (H.264)	CAVLC (H.264)

BENCHMARKS

ENCODE PERFORMANCE

H.264 1080p (1920x1080) 4:2:0 8bit 30fps (SINGLE NVENC)

Number of Streams / NVENC

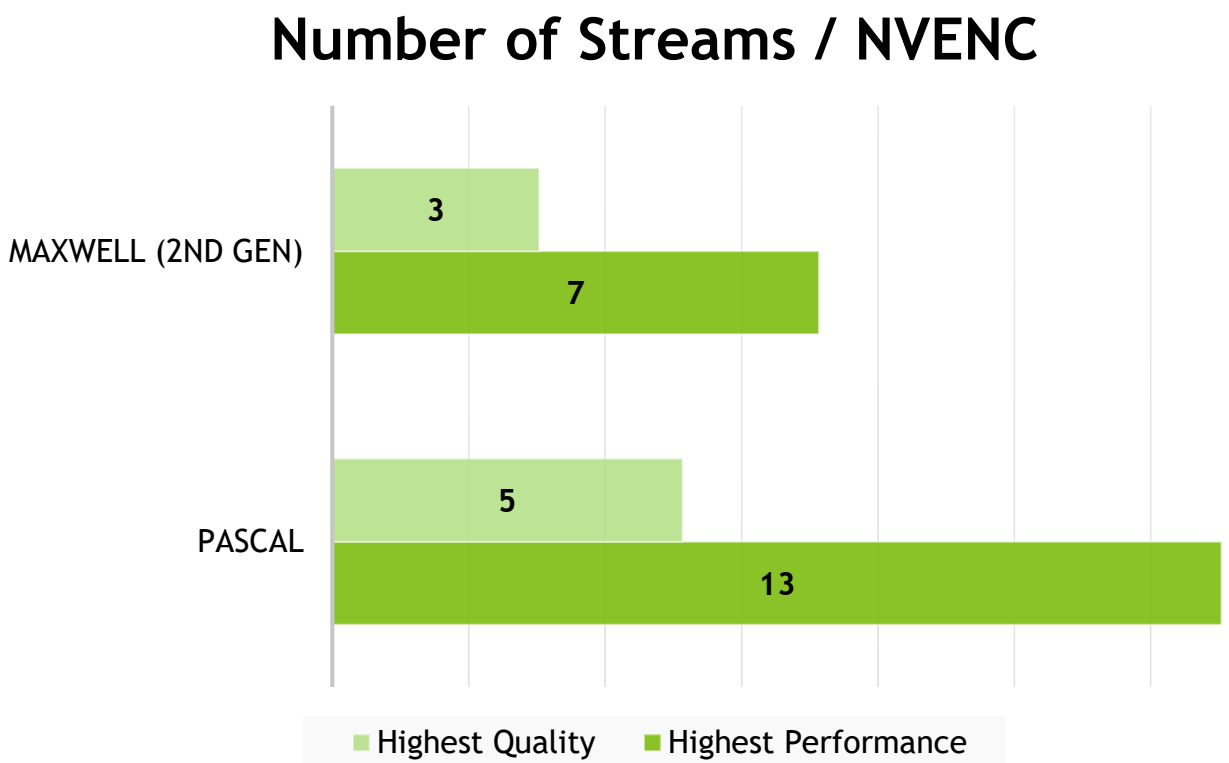
#NVENC	GPUs
x1	Kepler Quadro K2000/K2000D/K4000/K4200/ K5000/K5200/K6000 Kepler Tesla K20X/K40 Maxwell Quadro K2200 (1st Gen)/M2000 (2nd Gen) Maxwell (2nd Gen) Tesla M4 Pascal Quadro P2000/P4000
x2	Kepler Tesla K10/K80 Kepler GRID K2/K520 Maxwell (2nd Gen) Quadro M4000/M5000/M6000 Maxwell (2nd Gen) Tesla M6/M40 Pascal Quadro P5000/P6000 Pascal Tesla P4/P40
x3	Pascal Quadro GP100 Pascal Tesla P100
x4	Kepler GRID K1/K340 Maxwell (2nd Gen) Tesla M60

Note: All GPUs not featured above are limited to 2 simultaneous sessions

Performance represents an approximation of max performance and may vary based on GPU clock speed, OS, software versions, and motherboard configuration

ENCODE PERFORMANCE

HEVC 4K (3840x2160) 4:2:0 8bit 30fps (SINGLE NVENC)



#NVENC	GPUs
x1	Maxwell Quadro M2000 Maxwell Tesla M4 Pascal Quadro P2000/P4000
x2	Maxwell Quadro M4000/M5000/M6000 Maxwell Tesla M6/M40 Pascal Quadro P5000/P6000 Pascal Tesla P4/P40
x3	Pascal Quadro GP100 Pascal Tesla P100
x4	Maxwell Tesla M60

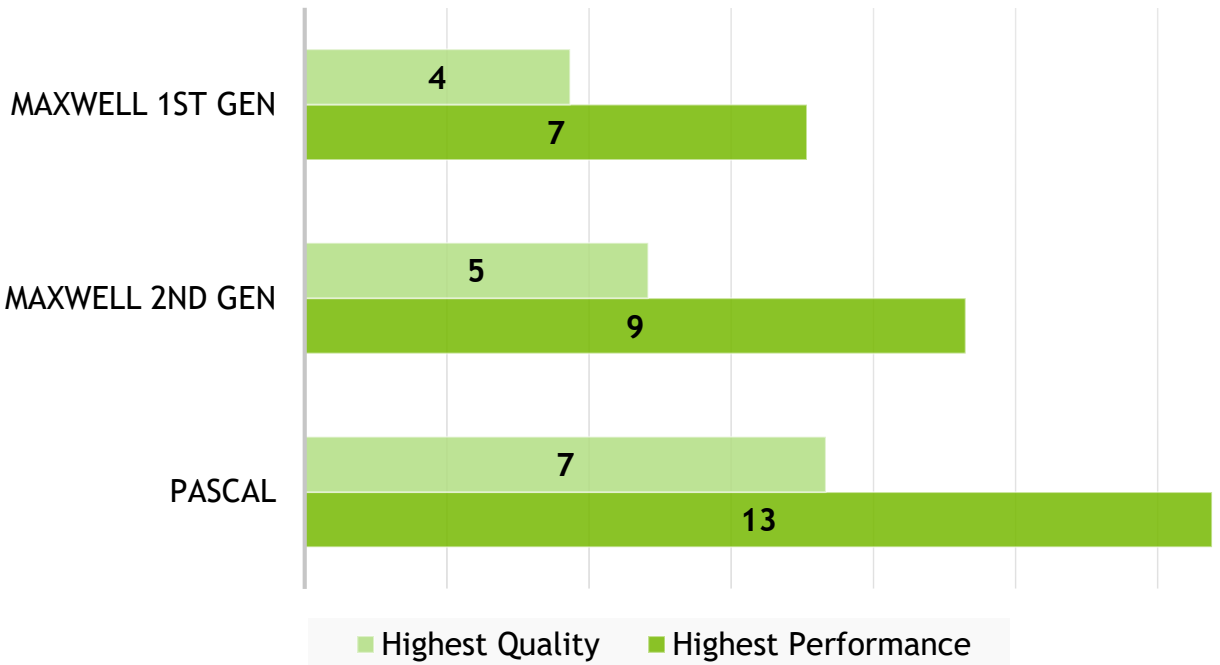
Note: All GPUs not featured above are limited to 2 simultaneous sessions

Performance represents an approximation of max performance and may vary based on GPU clock speed, OS, software versions, and motherboard configuration

ENCODE PERFORMANCE

H.264 1080p (1920x1080) 4:4:4 8bit 30fps (SINGLE NVENC)

Number of Streams / NVENC



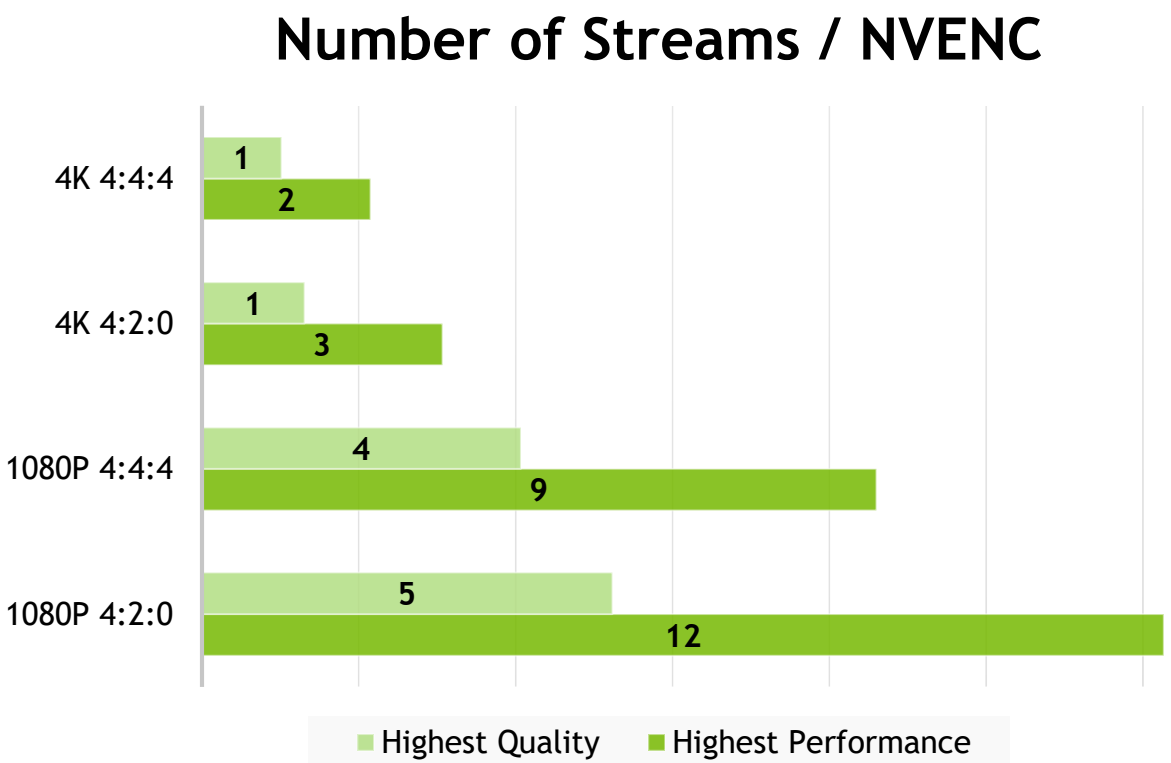
#NVENC	GPUs
x1	Maxwell Quadro K2200 (1st Gen)/M2000 (2nd Gen) Maxwell (2nd Gen) Tesla M4 Pascal Quadro P2000/P4000
x2	Maxwell (2nd Gen) Quadro M4000/M5000/M6000 Maxwell (2nd Gen) Tesla M6/M40 Pascal Quadro P5000/P6000 Pascal Tesla P4/P40
x3	Pascal Quadro GP100 Pascal Tesla P100
x4	Maxwell (2nd Gen) Tesla M60

Note: All GPUs not featured above are limited to 2 simultaneous sessions

Performance represents an approximation of max performance and may vary based on GPU clock speed, OS, software versions, and motherboard configuration

ENCODE PERFORMANCE

Pascal HEVC 10bit 30fps (SINGLE NVENC)



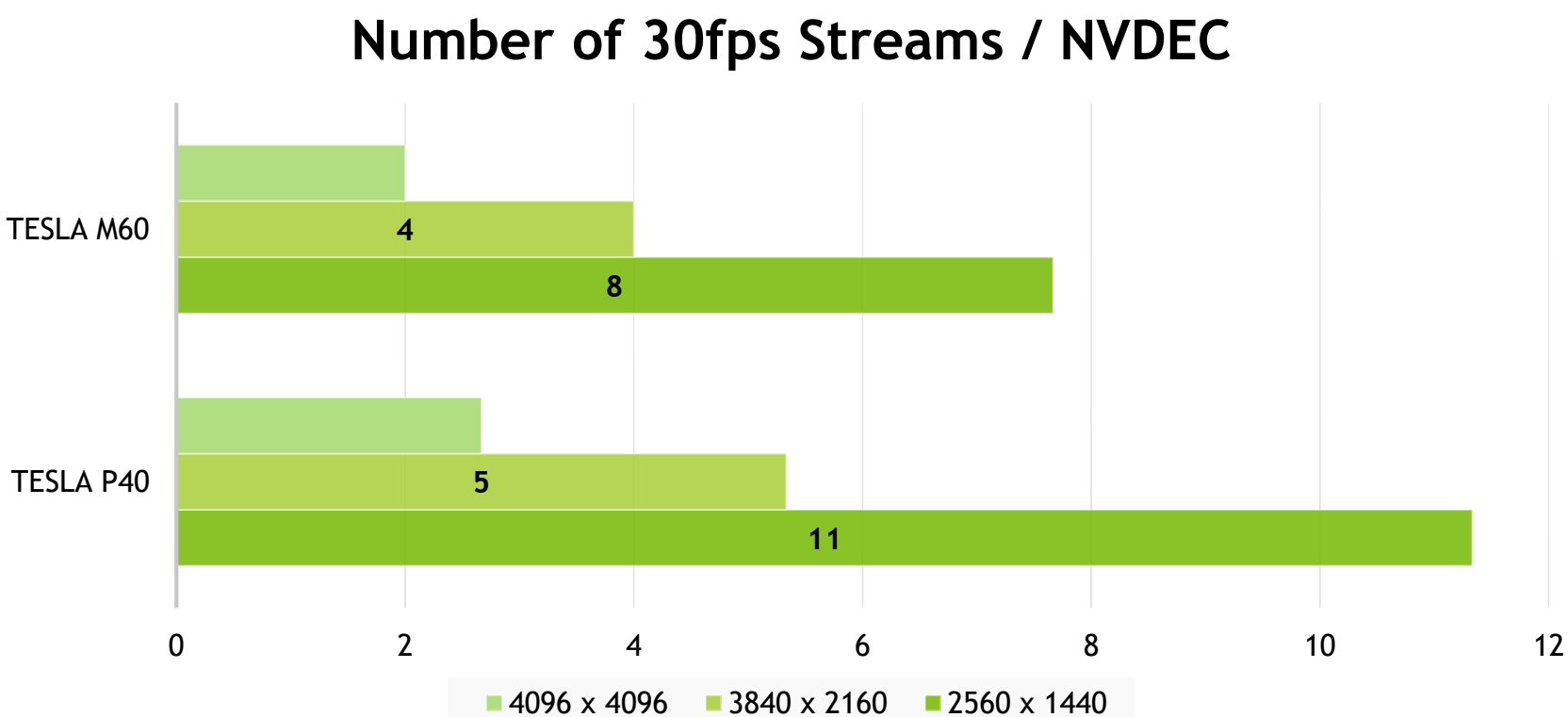
#NVENC	GPUs
x1	Pascal Quadro P2000/P4000
x2	Pascal Quadro P5000/P6000 Pascal Tesla P4/P40
x3	Pascal Quadro GP100 Pascal Tesla P100

Note: All GPUs not featured above are limited to 2 simultaneous sessions

Performance represents an approximation of max performance and may vary based on GPU clock speed, OS, software versions, and motherboard configuration

DECODE PERFORMANCE

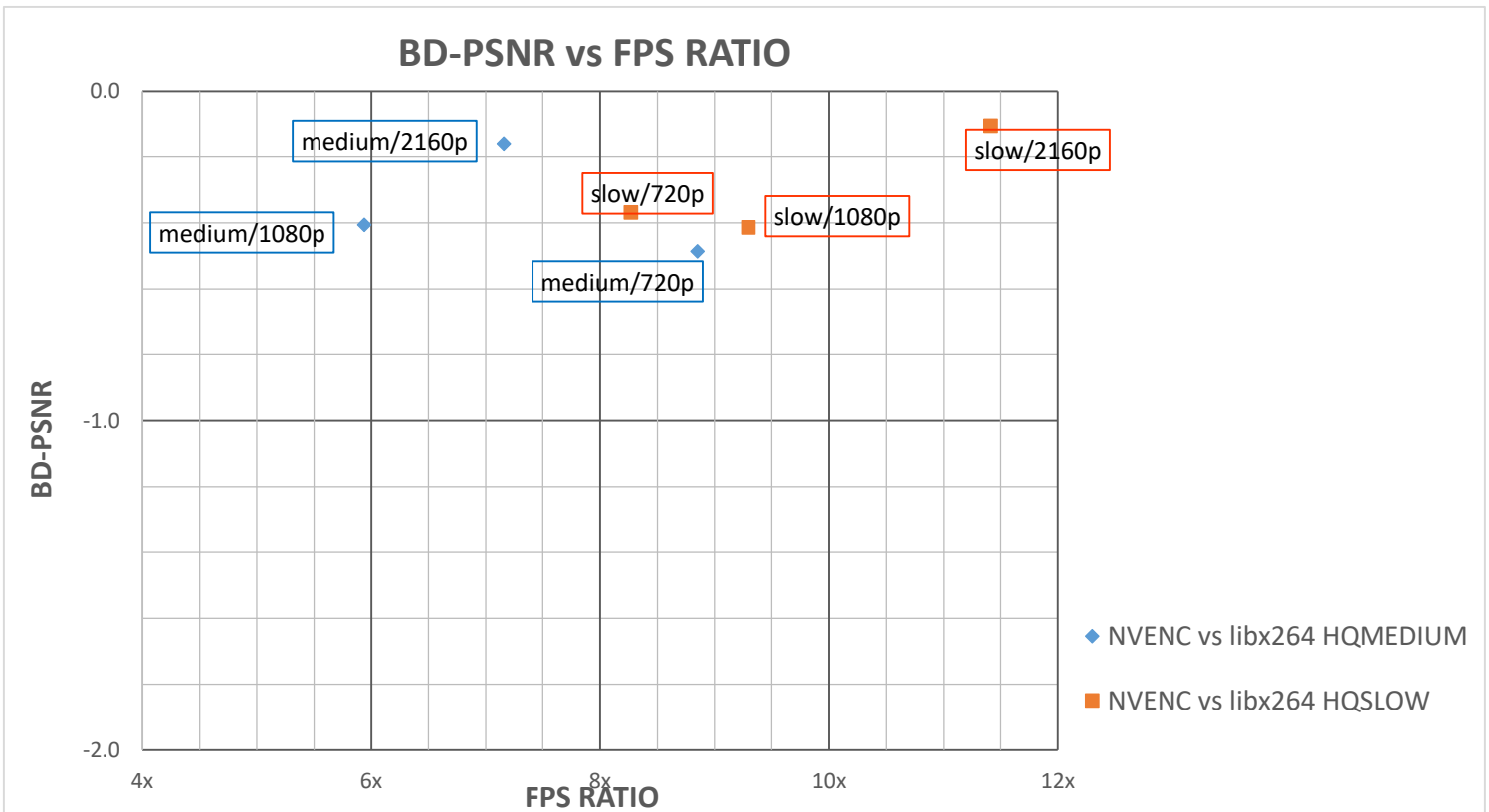
NVDEC H.264 YUV 4:2:0



Performance represents an approximation of max performance and may vary based on GPU clock speed, OS, software versions, and motherboard configuration

ENCODE PERF/QUALITY

Encode quality latest results (slow/med: ±0.4 dB within x264)



MOTION VECTOR QUALITY

- KITTI Vision Benchmark Suite for Optical Flow
- Measures distortion of motion vectors compared to “true” motion
- Average distortion ≈ 7%, improves 1-2% by motion hints

ME-ONLY MODE

Frame 0

Source: http://www.cvlibs.net/datasets/kitti/, under Creative Commons License

ME-ONLY MODE

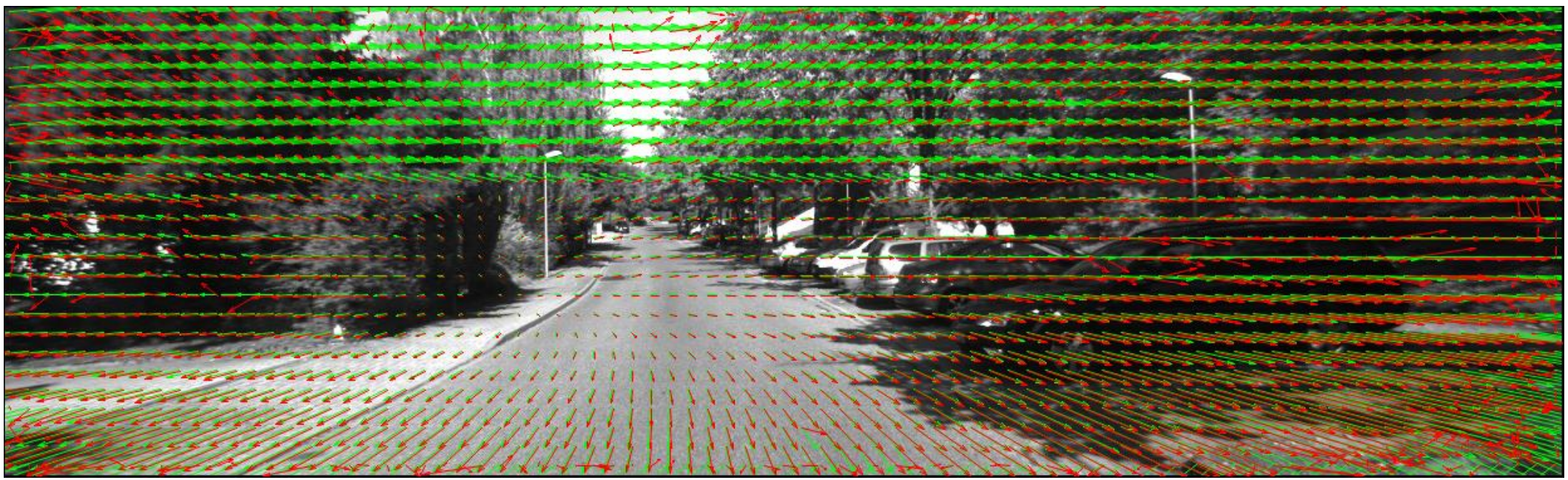
Frame 1

Source: http://www.cvlibs.net/datasets/kitti/, under Creative Commons License

ME-ONLY MODE

Motion Vector Distortion

→ "True" motion → NVENC estimated motion **Distortion score = 2%**



RESOURCES

Video Codec SDK: https://developer.nvidia.com/nvidia-video-codec-sdk

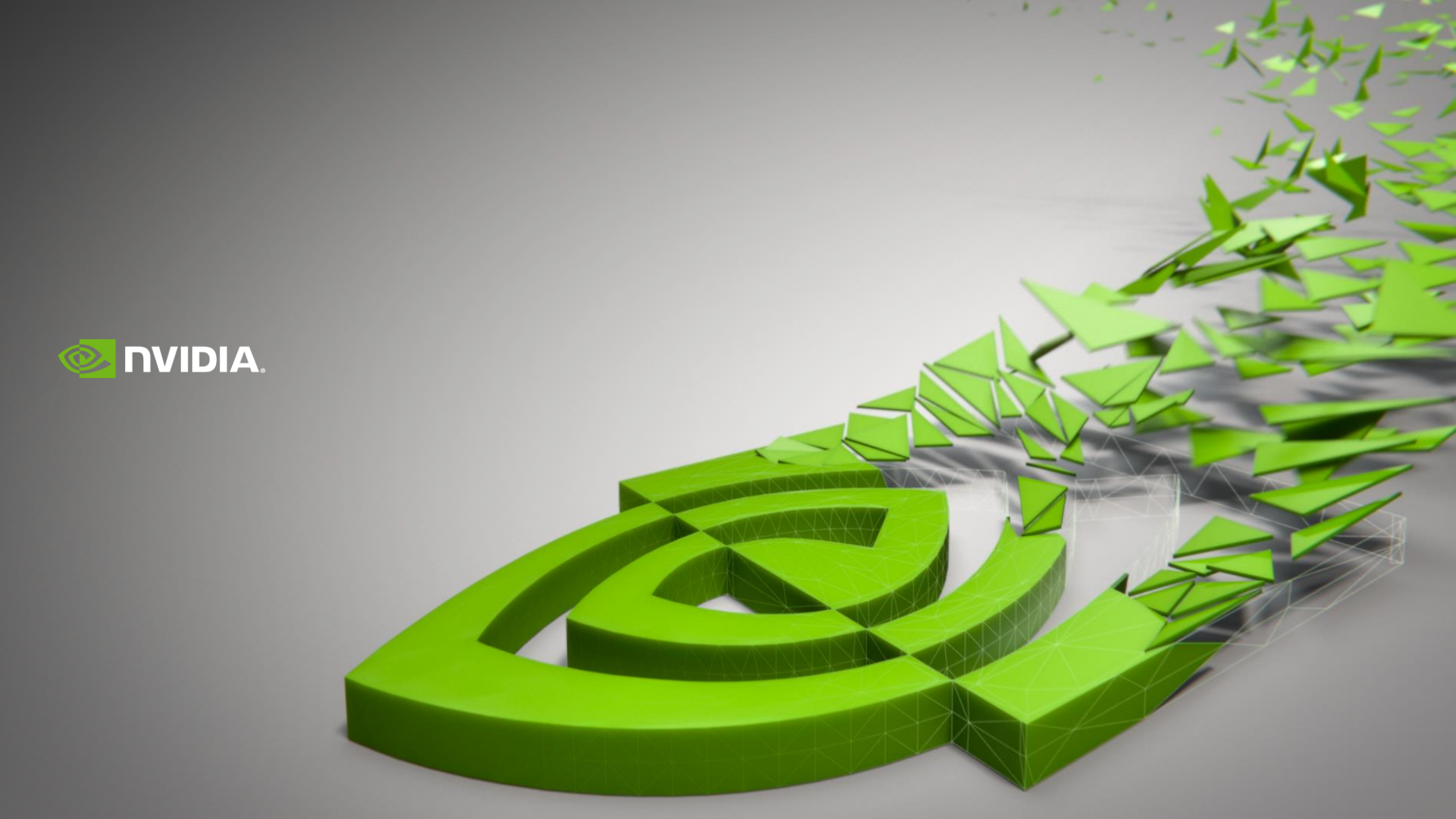
FFmpeg GIT: https://git.ffmpeg.org/ffmpeg.git

Libav GIT: https://git.Libav.org/libav.git

FFmpeg builds with hardware acceleration: http://ffmpeg.zeranoe.com/builds/

Video SDK support: video-devtech-support@nvidia.com

Video SDK forums: https://devtalk.nvidia.com/default/board/175/video-technologies/


NVIDIA