OPEN OPTIMIZED VVC IMPLEMENTATIONS ON ARM ARCHITECTURES

B. Bross, C. Lehmann, G. Hege, A. Wieckowski, D. Marpe

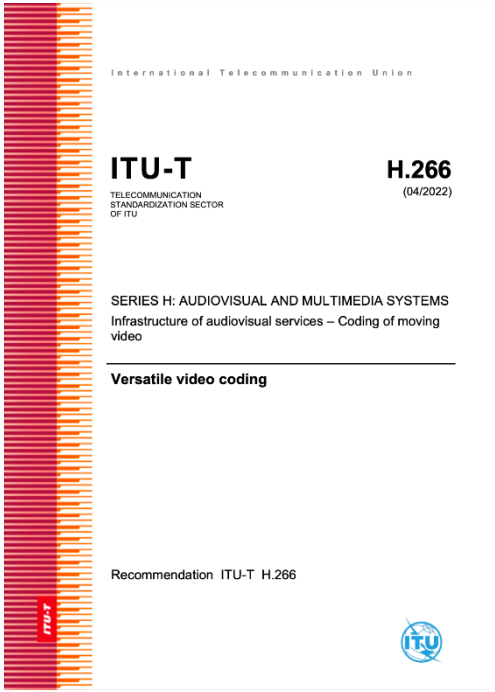
ACM Mile High Video ’23 – Denver, CO, USA

Fraunhofer HHI

Versatile Video Coding (VVC)

The Most Recent Standard

- Coding efficiency
  - 50% bit-rate reduction over HEVC
- Versatility
  - Video beyond standard- and high-definition e.g. 8K, UHD, HDR, WCG
  - Adaptive streaming with resolution changes e.g. Open-GOP DASH
  - Computer-generated, screen-captured content e.g. Gaming, Screen sharing
  - Immersive video e.g. VR / AR

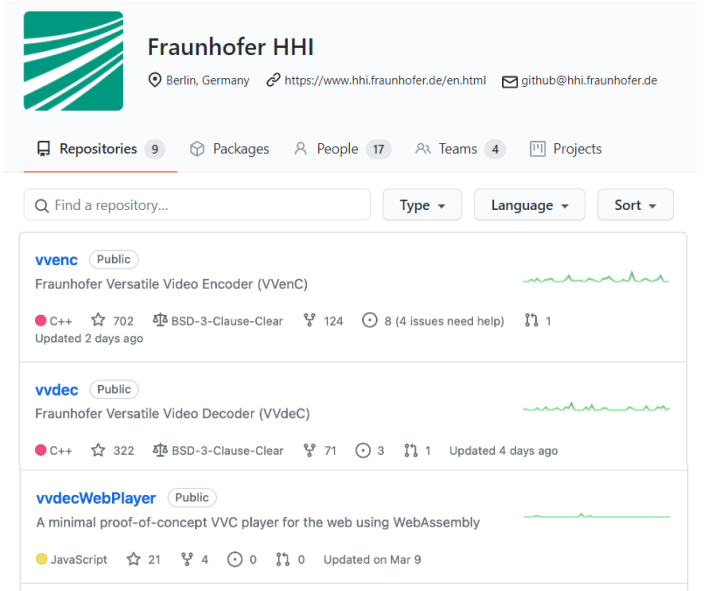


Versatile Video Coding (VVC)

Open & Optimized Implementations

Available on GitHub since Sep. 2020:

- VVenC – Versatile Video Encoder
  - Most efficient and fast open video encoder
  - fast offline encoding with random access
- VVdeC – Versatile Video Decoder
  - Fastest fully-compliant open VVC decoder
  - UHD 10bit live decoding on a laptop
  - HD live decoding on mobile



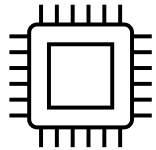
https://github.com/fraunhoferhhi/vvenc

https://github.com/fraunhoferhhi/vvdec

Why ARM…

…and how to run VVenC & VVdeC?

- Nowadays ARM goes beyond mobile
  - Apple M1, M2 silicon for laptops and workstations
  - ARM Neoverse silicon for servers / cloud instances
- Energy efficiency
  - Interesting for video encoders such as VVenC
- Single Instruction Multiple Data (SIMD)
  - Up to 3x speedup in VVenC without losing coding efficiency
  - Developed using x86 instrinsics (SSE4.2 to AVX2)
  - Automatic translation to ARM NEON using SIMDe(verywhere)
    -> easy but with some pitfalls…



Experimental Setup

Systems



	Workstation		Laptop		Mobile
CPU	M1 Ultra	Xeon 6348	M1 Max	i9-12900H	Snapdragon 865+
Cores*	16P @ 3.2 GHz 4E @ 2.06 GHz	28 @ 2.6 GHz	8P @ 3.2 GHz 2E @ 2.06 GHz	6P @ 5.0 GHz 8E @ 3.4 GHz	1/3P @ 3.09/2.4 GHz 4E @ 1.9 GHz
Threads**	20	56	10	20	8
OS	MacOS 13.2	Ubuntu 22.04	MacOS 13.2	Ubuntu 22.04	Android 12
Compiler	clang 14.0.0	gcc 11.3.0	clang 14.0.0	gcc 11.3.0	clang 14.0.6
SIMD	NEON	AVX512	NEON	AVX2	NEON

* P = performance core, E = efficiency core, **including hyper-threading

Experimental Setup

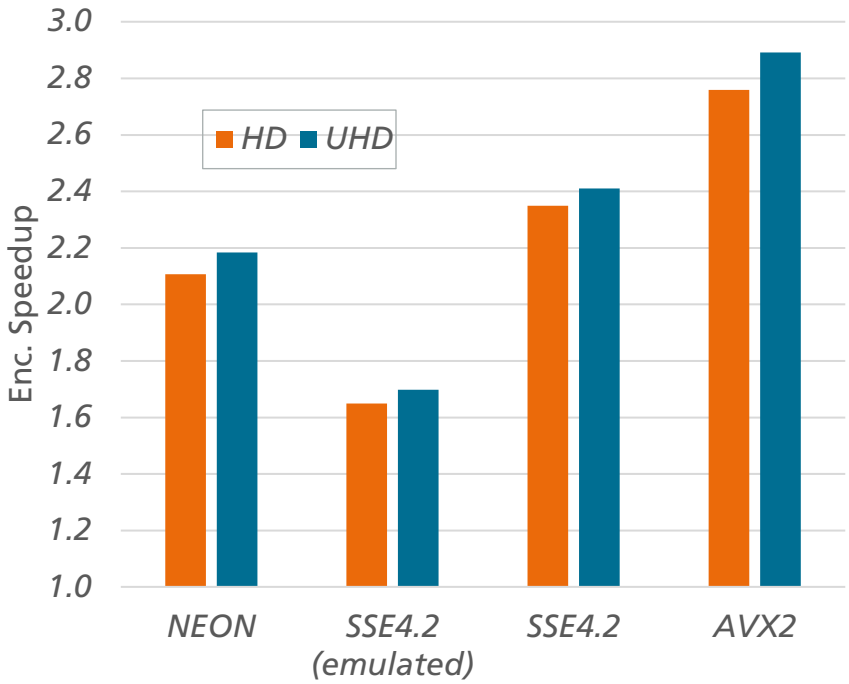
Configuration

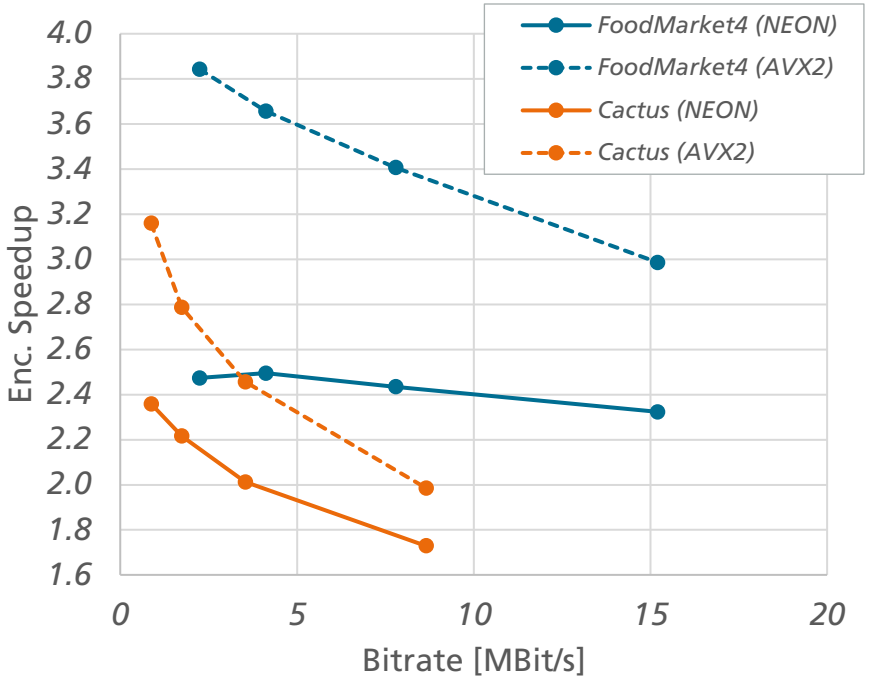
- VVenC v1.7.0
  - Laptop: 8 threads
  - Workstation: 20 threads (using WPP + 2x2 tiles for better scaling)*
- VVdeC v1.6.1
- JVET common test conditions (doc. JVET-T2010)
  - 5 HD & 6 UHD sequences
  - 10-bit coding
  - 1s random access
  - QPs 22, 27, 32, 37
- Unix *time* utility for runtime

*V. George et al., "Efficient Multi-Threading Strategies in VVenC, an Open and Optimized VVC Encoder Implementation," 2022 IEEE ISM

Impact of Vectorization

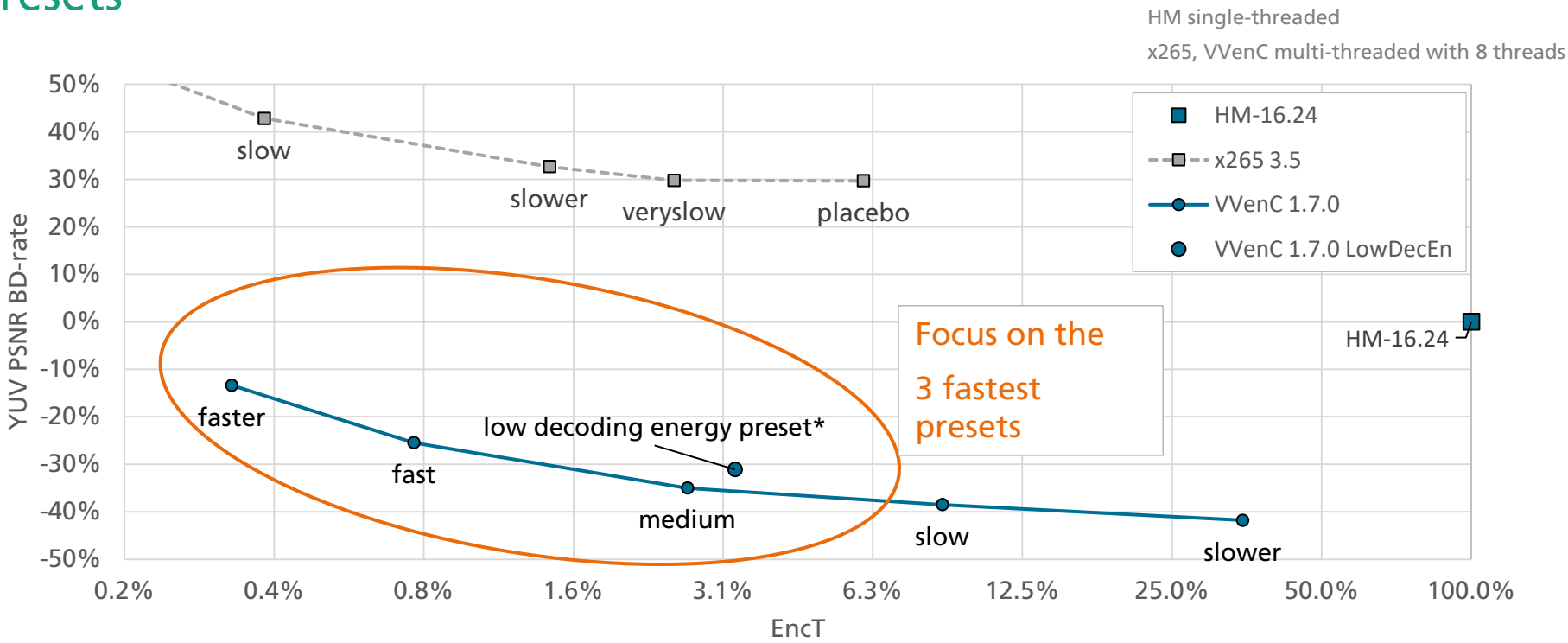
SIMD vs. Scalar





VVenC Encoder Runtime

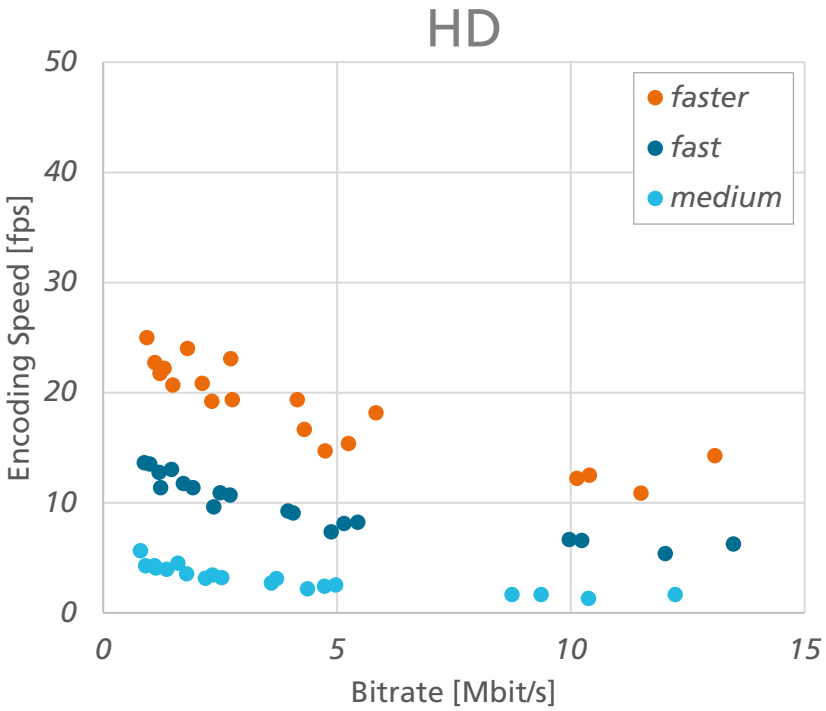
Presets

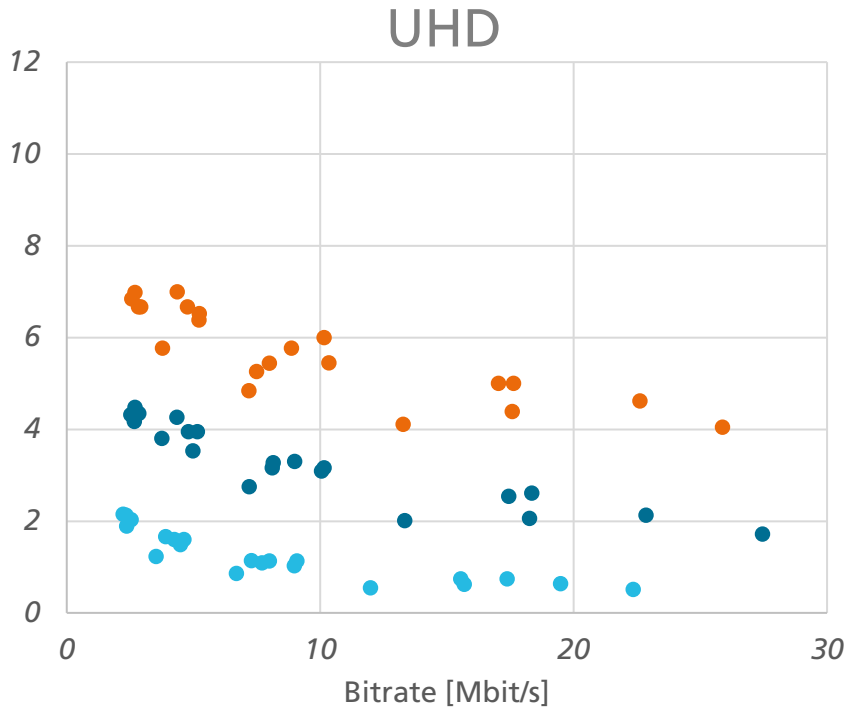


* M. Kränzler et al., "Optimized Decoding-Energy-Aware Encoding in Practical VVC Implementations," in proceedings of ICIP 2022.

VVenC Encoder Runtime

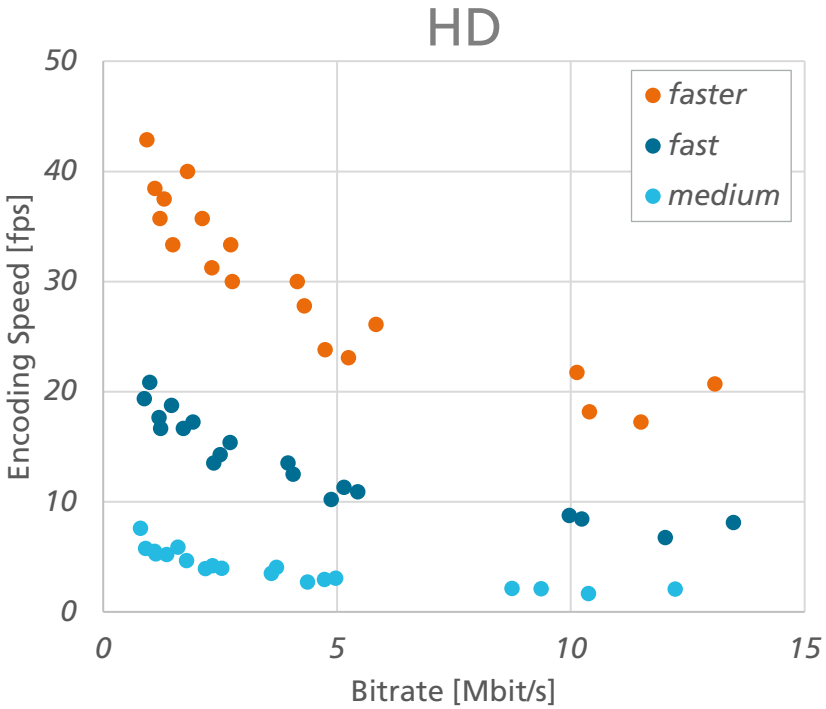
Workstation – 20 threads – x86

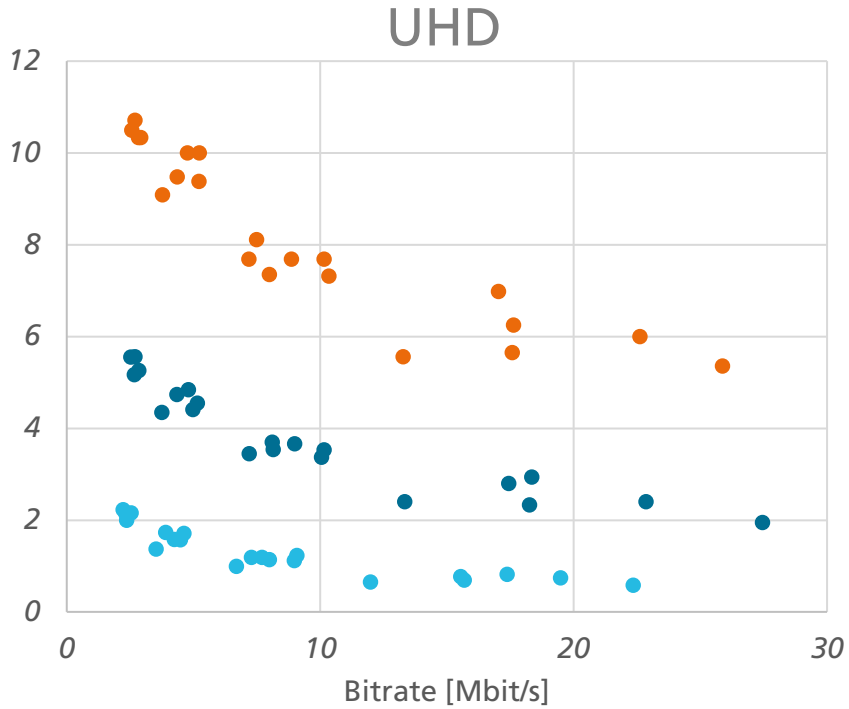




VVenC Encoder Runtime

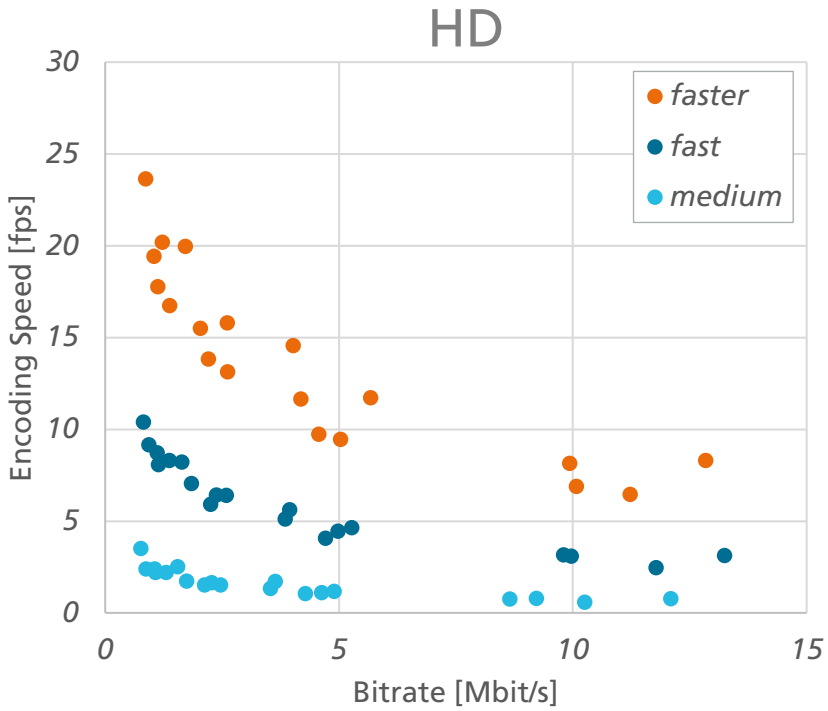
Workstation – 20 threads – ARM

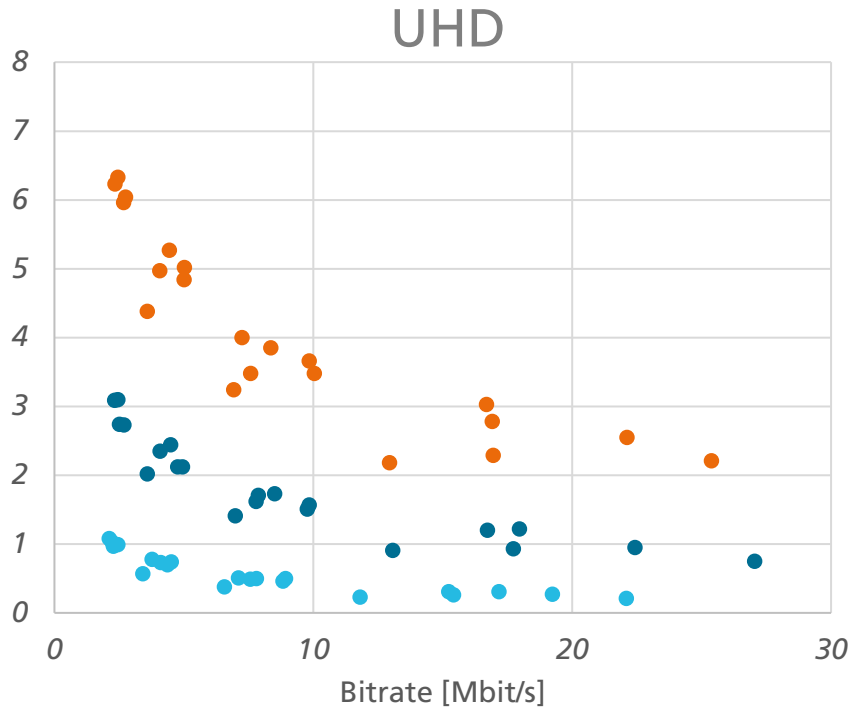




VVenC Encoder Runtime

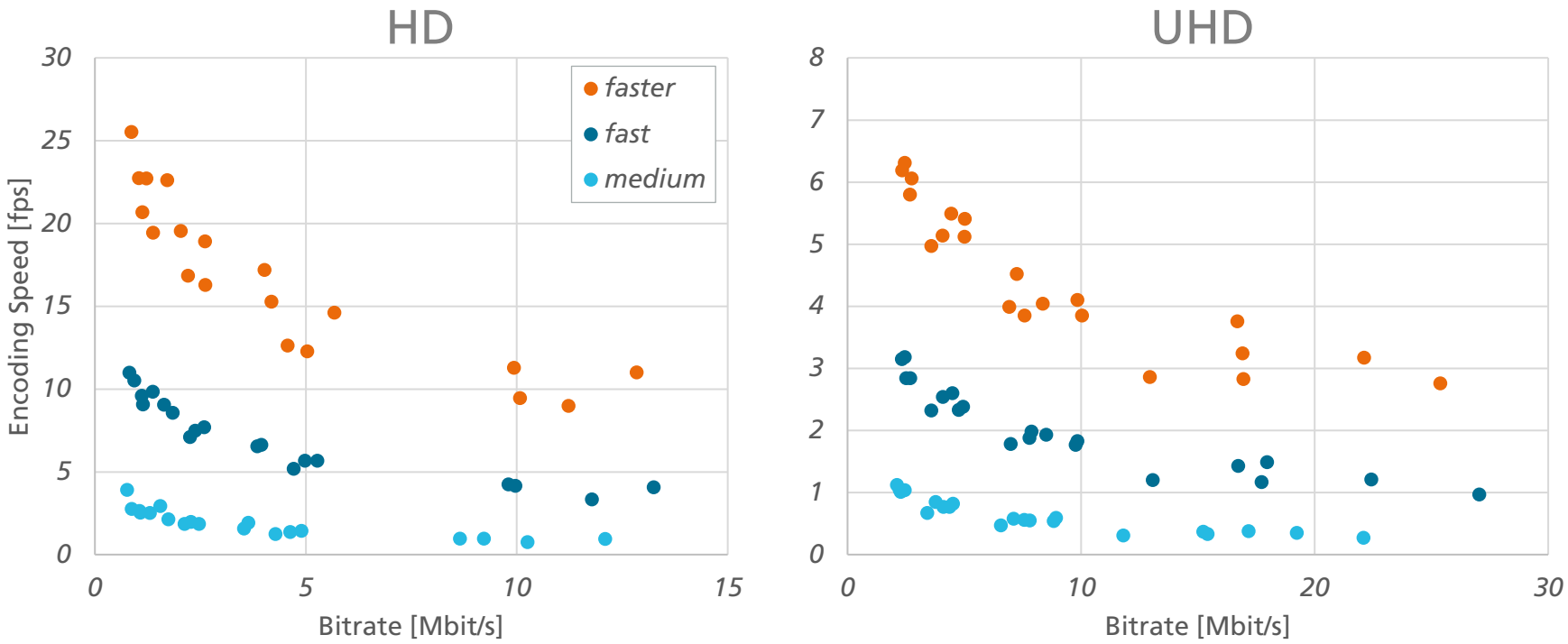
Laptop – 8 threads – x86





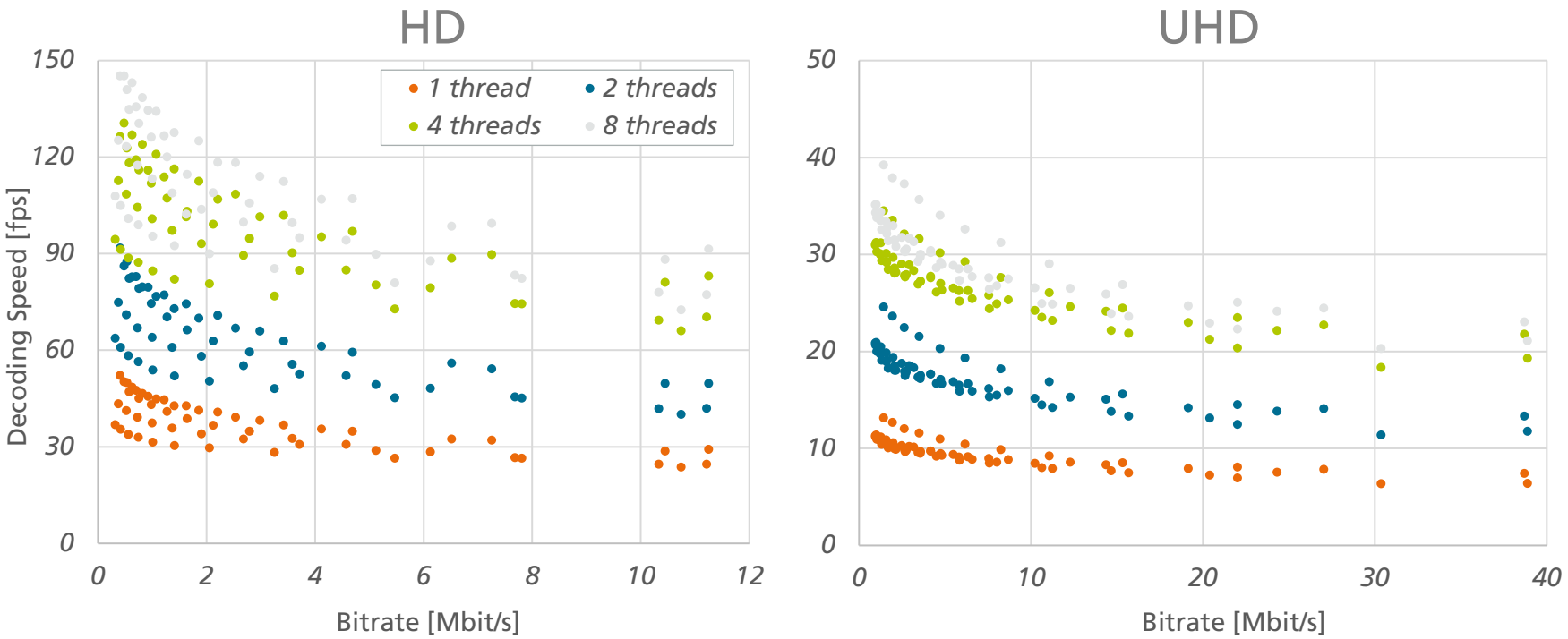
VVenC Encoder Runtime

Laptop – 8 threads – ARM



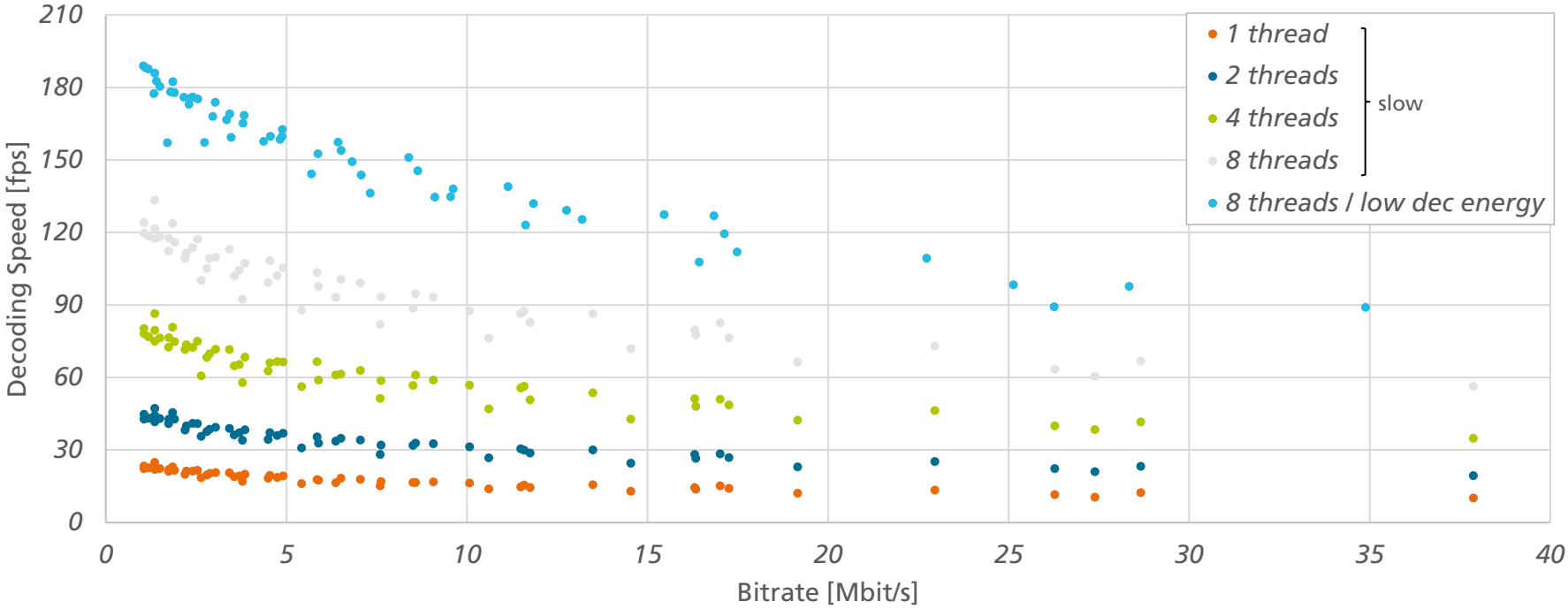
VVdeC Decoder Runtime

Mobile – Low Decoding Energy Preset – ARM



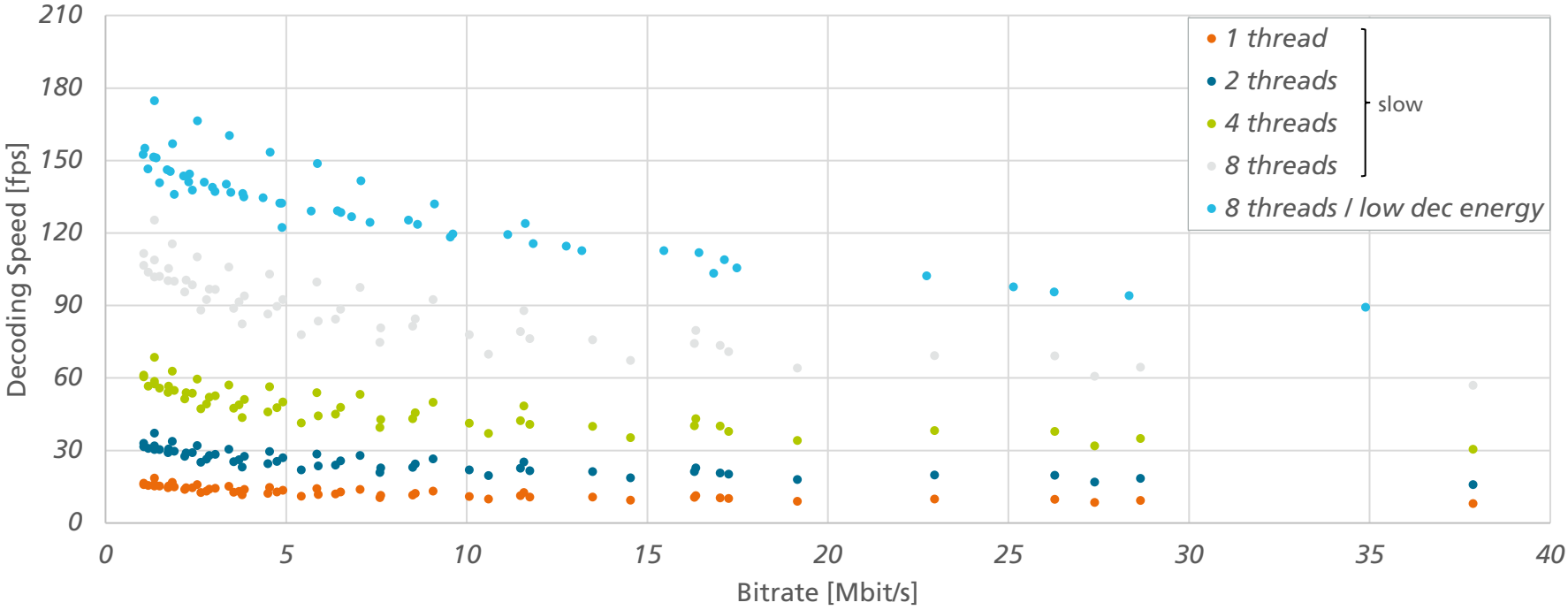
VVdeC Decoder Runtime

Laptop – UHD – x86



VVdeC Decoder Runtime

Laptop – UHD – ARM



Hands on
Android Playback

- aarch64 build of VVdeC on Android:
  - MPV media player
  - ExoPlayer enabling VVC + MPEG-H demo by Fraunhofer HHI + IIS
  - Bitmovin player demo at NAB 2023
- 1080p60 10bit @ 3Mbit/s VVC playback
  - Galaxy 7+, Galaxy 8
- 5h non-stop playback with low decoding engery preset

Hands on

FFmpeg integration

- Patches to integrate VVenC & VVdeC in FFmpeg have been submitted

```
ffmpeg -hide_banner -codecs | grep vvc

DEV.L. vvc                  H.266 / VVC (Versatile Video Coding)
(decoders: libvvdec ) (encoders: libvvenc )
```

- VVenC & VVdeC available via Homebrew for Apple Silicon

```
brew install vvdec vvenc
```

- Instructions how to build FFmpeg using these patches:

  https://github.com/fraunhoferhhi/vvenc/wiki/FFmpeg-Integration

Conclusion & Outlook

VVC on ARM

- Open VVC implementations VVenC & VVdeC ported and tested on ARM platforms:
  - Fast UHD encoding up to 10fps on Apple M1 Ultra and up to 6fps on M1 Max
  - Live HD / UHD decoding on an ARM tablet / Apple M1 Max
  - Automatic translation from x86 SIMD to ARM NEON as a good start
- Future work on VVenC & VVdeC on ARM:
  - Optimizing / “handtuning” SIMD
  - Assessing energy efficiency
  - Evaluating ARM-based server platforms for cloud transcoding scenarios

Annex A – Additional resources

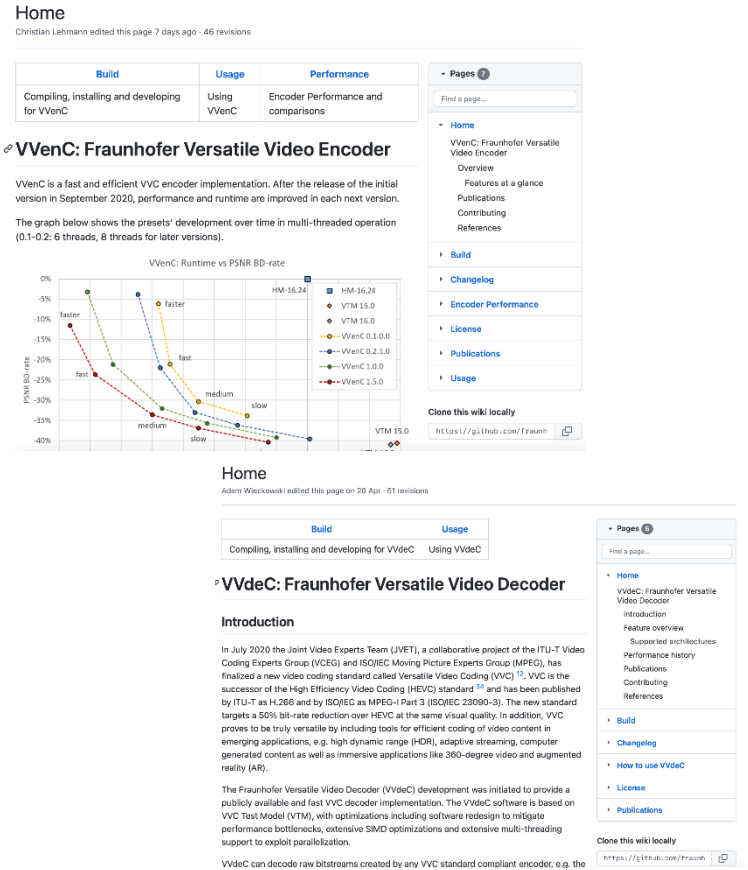
VVenC and VVdeC wiki pages

Most recent information on:

- How to build?
- How to use?
- Performance
- Publications

https://github.com/fraunhoferhhi/vvenc/wiki

https://github.com/fraunhoferhhi/vvdec/wiki



Annex A – Additional resources

Publications

- J. Brandenburg et al., "**Towards Fast and Efficient VVC Encoding**", IEEE 22nd Workshop on Multimedia Signal Processing (MMSP 2020), Tampere, Finland, 2020.
- B. Bross, C. Helmrich, A Wieckowski "**Versatile Video Coding – Open Optimized Implementations**", Workshop on the IEEE Picture Coding Symposium (PCS) 2021, Jul. 2021. https://youtu.be/IWPBuS2diVg
- A Wieckowski et al., "**VVenC: An Open And Optimized VVC Encoder Implementation**" 2021 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), 2021, pp. 1-2. https://ieeexplore.ieee.org/document/9455944
- C. R. Helmrich et al., "**A study of the extended perceptually weighted peak signal-to-noise ratio (XPSNR) for video compression with different resolutions and bit depths**", in ITU Journal: ICT Discoveries, vol. 3, no. 1, May 2020. http://handle.itu.int/11.1002/pub/8153d78b-en
- R. Skupin, C. Bartnik, A. Wieckowski, Y. Sanchez, B. Bross, C. Hellge, and T. Schierl, "**Open GOP Resolution Switching in HTTP Adaptive Streaming with VVC**," 35th Picture Coding Symposium (PCS), Bristol, US, June-July 2021.
- R. Skupin, C. Bartnik, A. Wieckowski, Y. Sanchez, B. Bross, "**Constrained RASL encoding for bitstream switching**," document JVET-W0133, Joint Video Experts Team (JVET), July 2021.
- M. Kränzler et al., "**Optimized Decoding-Energy-Aware Encoding In Practical VVC Implementations**," 2022 IEEE International Conference on Image Processing (ICIP), 2022, pp. 1536-1540.

Annex B – Encoder comparison settings

Encoding with preset P for quality Q

- HD and UHD sequences from JVET common test conditions JVET-T2010:

  https://jvet-experts.org/doc_end_user/documents/20_Teleconference/wg11/JVET-T2010-v2.zip

- Command line options for x265* encoder (no sequence specific parameters) tuned for PSNR (**`T = psnr`**) or MS-SSIM (**`T = ssim`**)

```
-D 10 --preset P --tune T --crf Q --keyint <1s> --min-keyint <1s> --profile main10 --
output-depth 10
```

* optimal multithreading with x265 is achieved by restricting the number of CPU cores to be used to the desired number of threads (e.g. 8) since x265 determines the best combination of frame level and WPP parallel threads.