dav1d, 1 year later

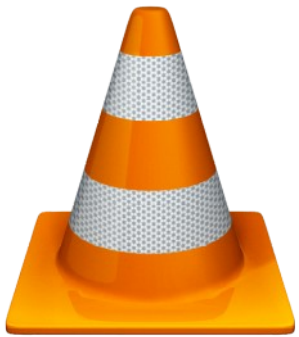
Jean-Baptiste Kempf

0202-2020

President of VideoLAN

Work/Manage VLC, x264, FFMpeg, dav1d

Other multimedia projects



VP9++?

- VP9 is a semi-failure
- Good format, royalties OK
- Rarely used
  - Have you ever watched an anime rip in VP9?
  - Spec?
- YT, Netflix

AV1

- Different from just **VP10**
- AOM, Mozilla, Cisco
- Excellent results

AV1 ecosystem

- Numerous encoders
  - libaom, SVT-AV1, rav1e
  - EVE-AV1, Ateme, Harmonic, Bitmovin
  - Ngcodec, FPGA, …
- Numerous deployments
  - Youtube, Netflix, Facebook
  - Cloud vendors
- Hardware is coming in 2020
  - Intel, nVidia, AMD?
  - Samsung TV, Amlogic, Broadcom

- Competion is coming?
  - VVC in July 2020, EVC in April 2020
  - MPEG-5 LC-EVC
  - AV2???
- Royalties
  - VVC is based on HEVC
    - 5 patent pools? :D
    - Are improvements enough to justify?
    - HEVC semi-failure
  - EVC is not enough
    - Gains?
    - MC-IF
  - LC-EVC is not actually a codec

Dav1d goals

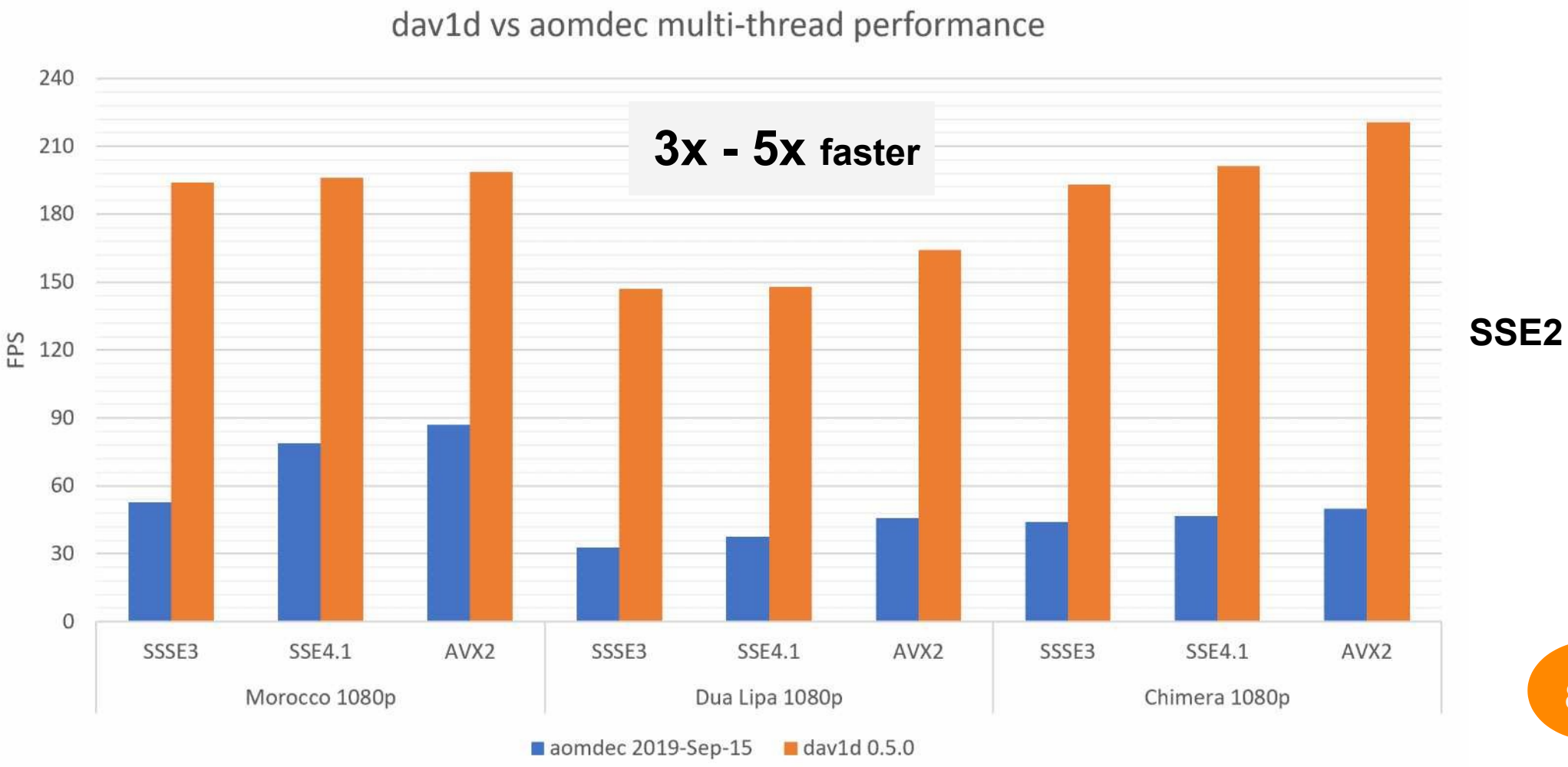
- "AV1 needs a great software decoder"
- Faster decoder everywhere
- Very portable and cross-platform
- Small binary size *(ffvp9)*

Launched last year

- Announced at VDD 2018
- First release in december 2018
- Last release: ***0.5.2***, *0.6.0 soon*

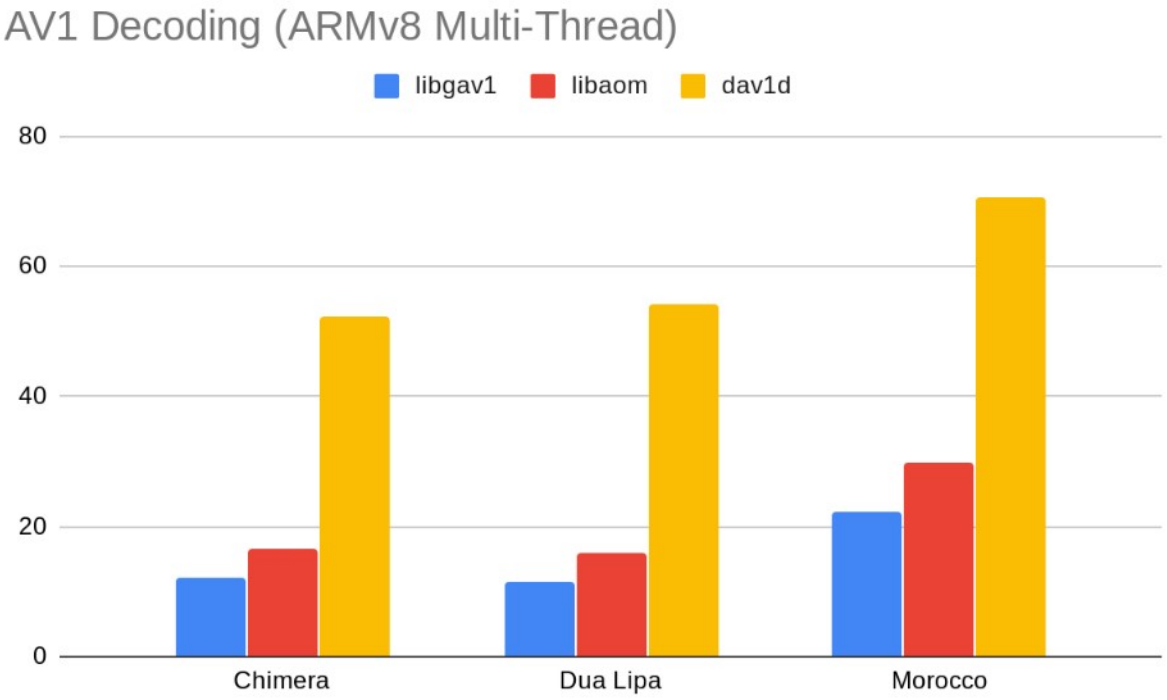
- *Oct '18* Announce
- *Dec '18* **0.1** 4x faster than libaom on x64
- *Mar '19* **0.2** 2x faster than libaom on ARM64, 4x on ARM32, 5x on x64
- *May '19* **0.3** Focus on SSSE3 *(+25%)*, ARM *(+12%)*
- *Aug '19* **0.4** Bugs, MSAC, RAM usage, VSX
- *Oct '19* **0.5** Finish ARM64, SSSE3
- Dec '19 **0.5.2** SSE2, ARM32

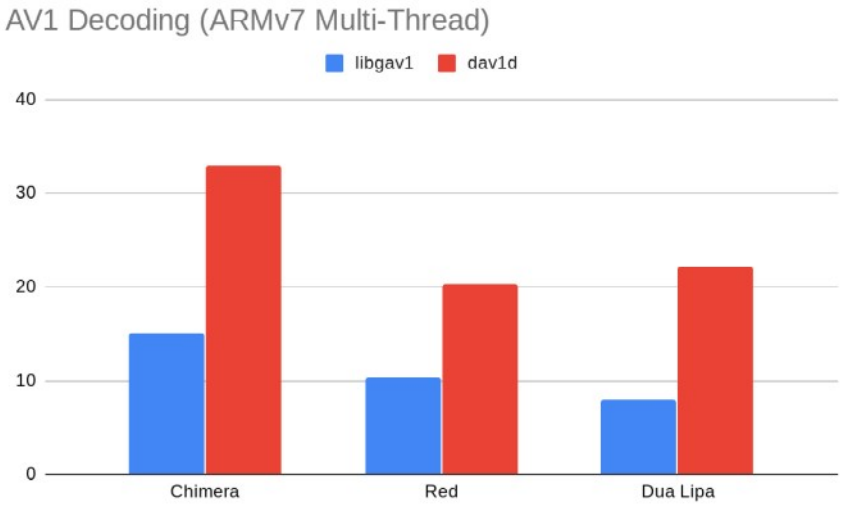
Fast on desktop



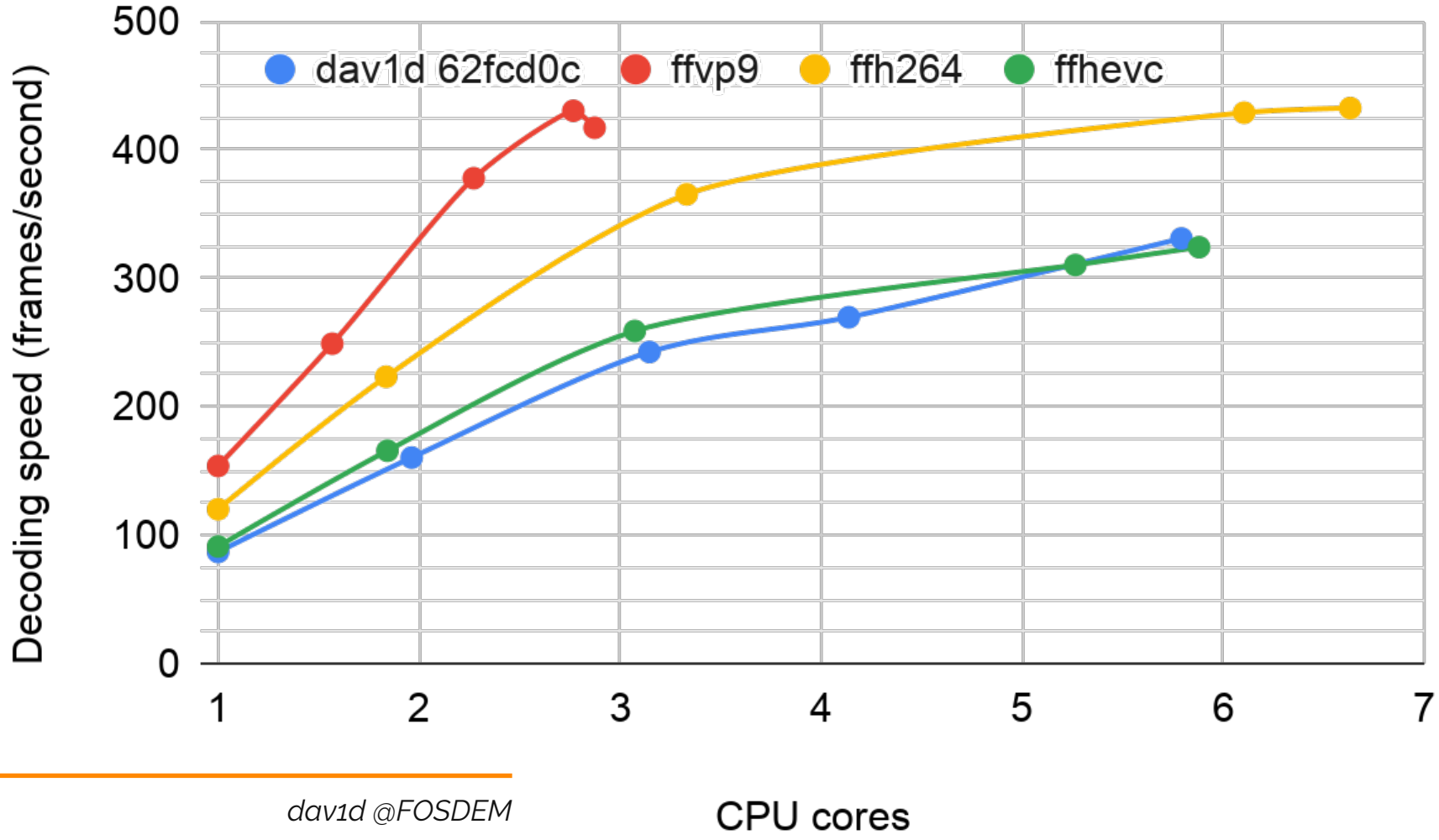
SSE2

Faster on ARM



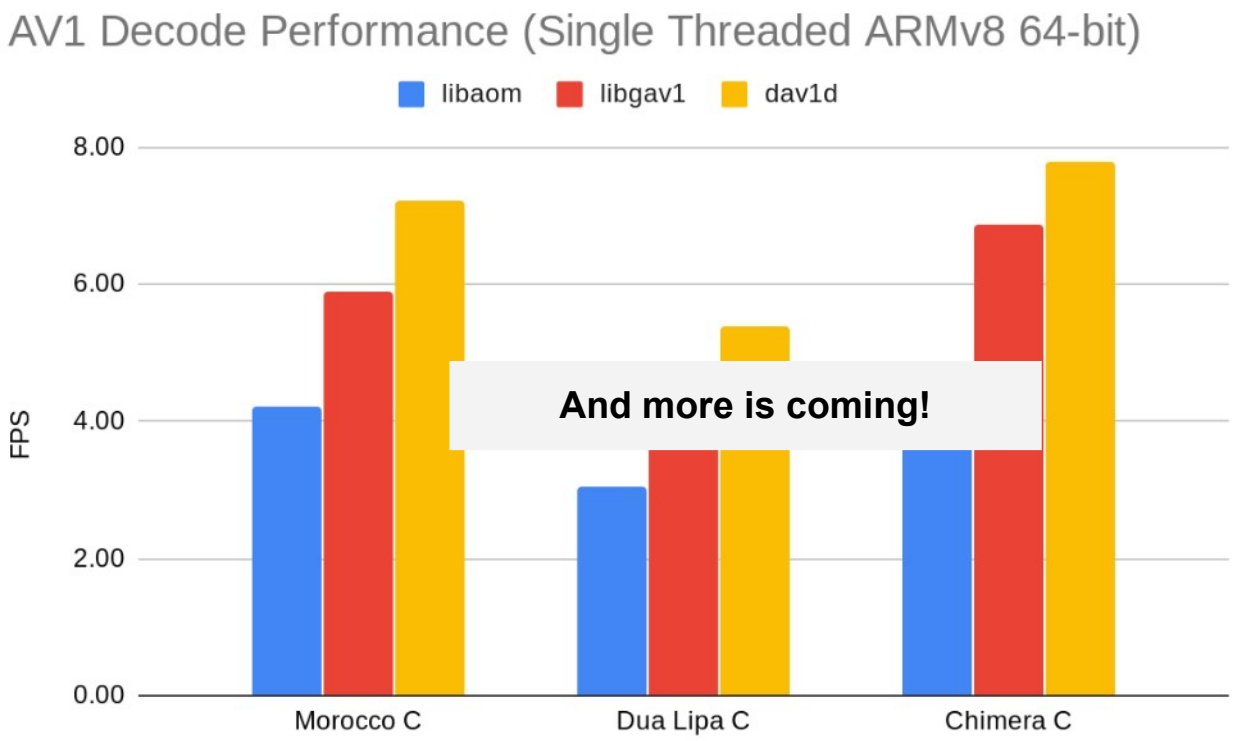


2,5x - 4x faster

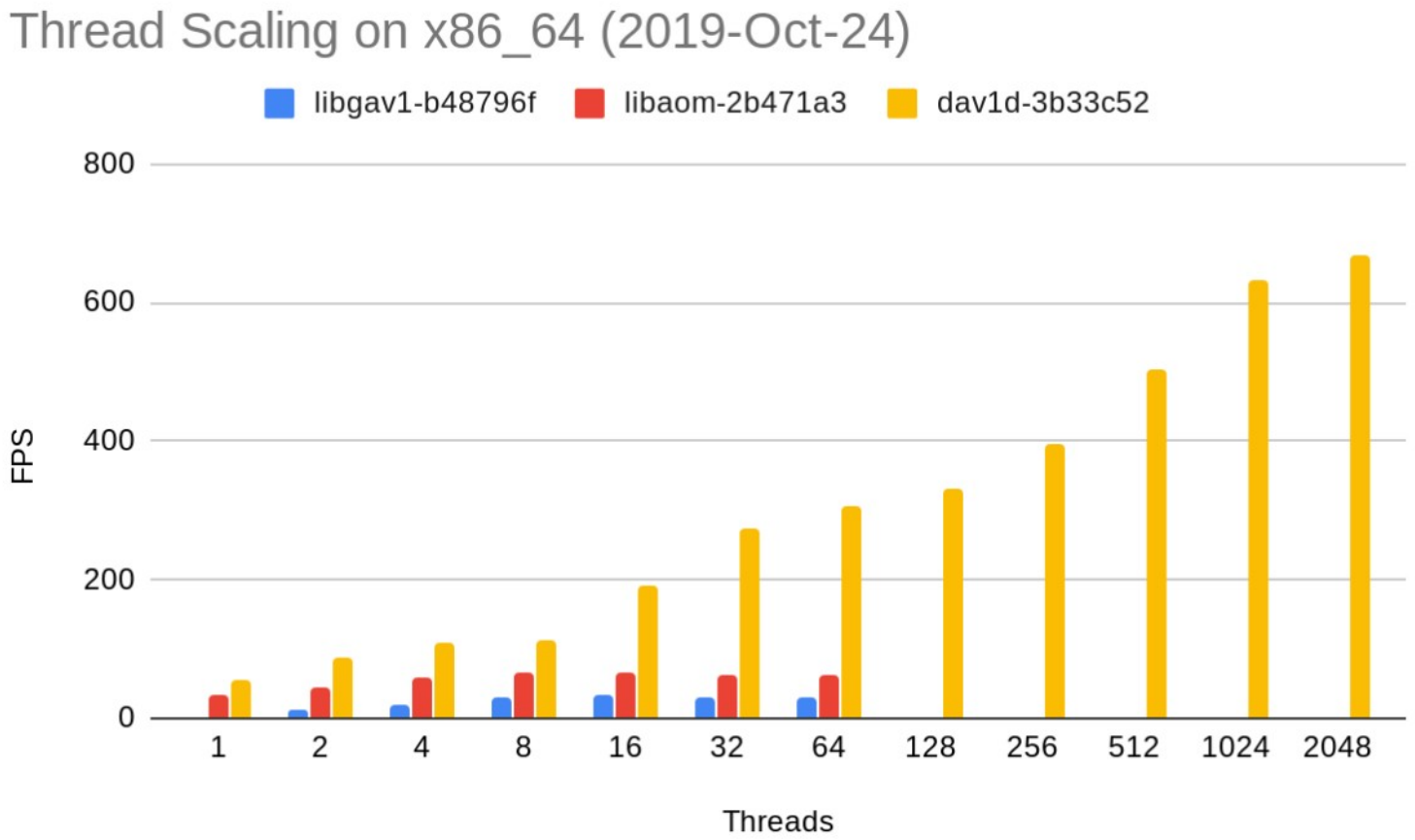

dav1d 62fcd0c
ffvp9
ffh264
ffhevc
Decoding speed (frames/second)
500
400
300
200
100
0
1
2
3
4
5
6
7
CPU cores

- Dual Passes
  - Rare inside a decoder
  - First pass to analyze, Second to decode
- Dual Threading model
  - Tile Thread
  - Frame Thread
  - Need to set both to get best decoding

1. C version is faster



2. Threading is better



3. low-level development

C (no C++ overhead)

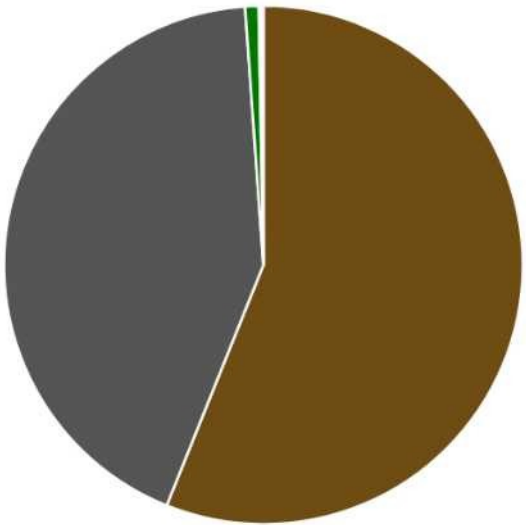
Hand-written asm

No intrinsics

Programming languages used in this repository

Measured in bytes of code. Excludes generated and vendored code.

Assembly	56.08 %
C	42.78 %
Meson	0.84 %
C++	0.15 %
Objective-C	0.14 %



ASM aware code

- MSAC
- Inverse Transform
- Motion Compensation
- Intra Pred
- Loopfilter
- Loop Restoration
- CDEF
- Film Grain

Non-ASM code

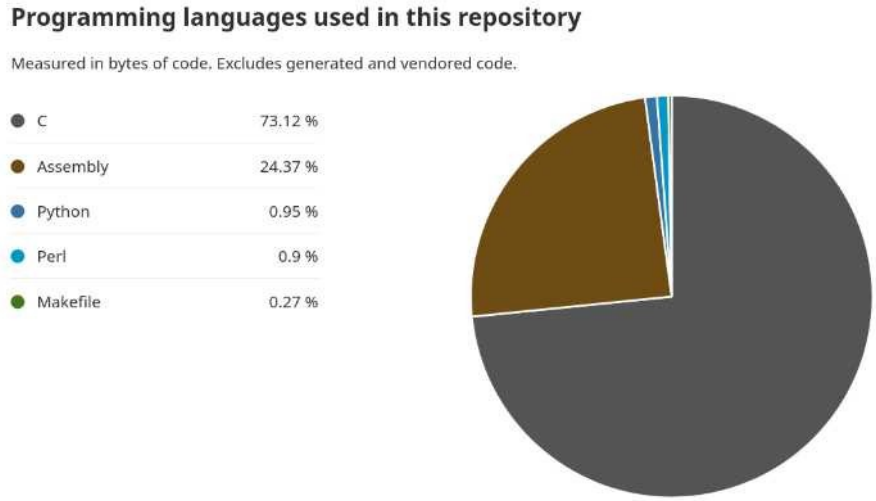
- Decode_coef (8%)
- Ref_mv (12%)
- Decode

dav1d

	AVX-2	SSSE-3 32 + 64bit	ARM64	ARM32
MSAC	→	Only SSE2	Yes	No
Inverse Transform	Yes	Yes	Yes	No
Motion Compensation	Yes	Yes *Warp SSE2*	Yes *emu_edge*	Yes *emu_edge*
Intra Pred	Yes *z1, z2, z3*	Yes	Yes *z1, z2, z3*	Partial
Loopfilter	Yes	Yes	Yes	Yes
Loop Restoration	Yes	Yes *Wiener SSE2*	Yes	Yes
CDEF	Yes	Yes + SSE2	Yes	Yes
Film Grain	Yes *Except 4:4:4*	Yes	No	No

X264, libavcodec

- x264
  - 68kLoC **C**
  - 37kLoC **asm** (25k x86, 12k ARM)
- libavcodec
  - 540 kLoC **C**
  - 80 kLoC **asm** (40k x86, 40k ARM)
- dav1d
  - 25 kLoC **C**
  - 64 kLoC **asm** (45k x86, 19k ARM)



GSoC 2019:
GPU optimizations

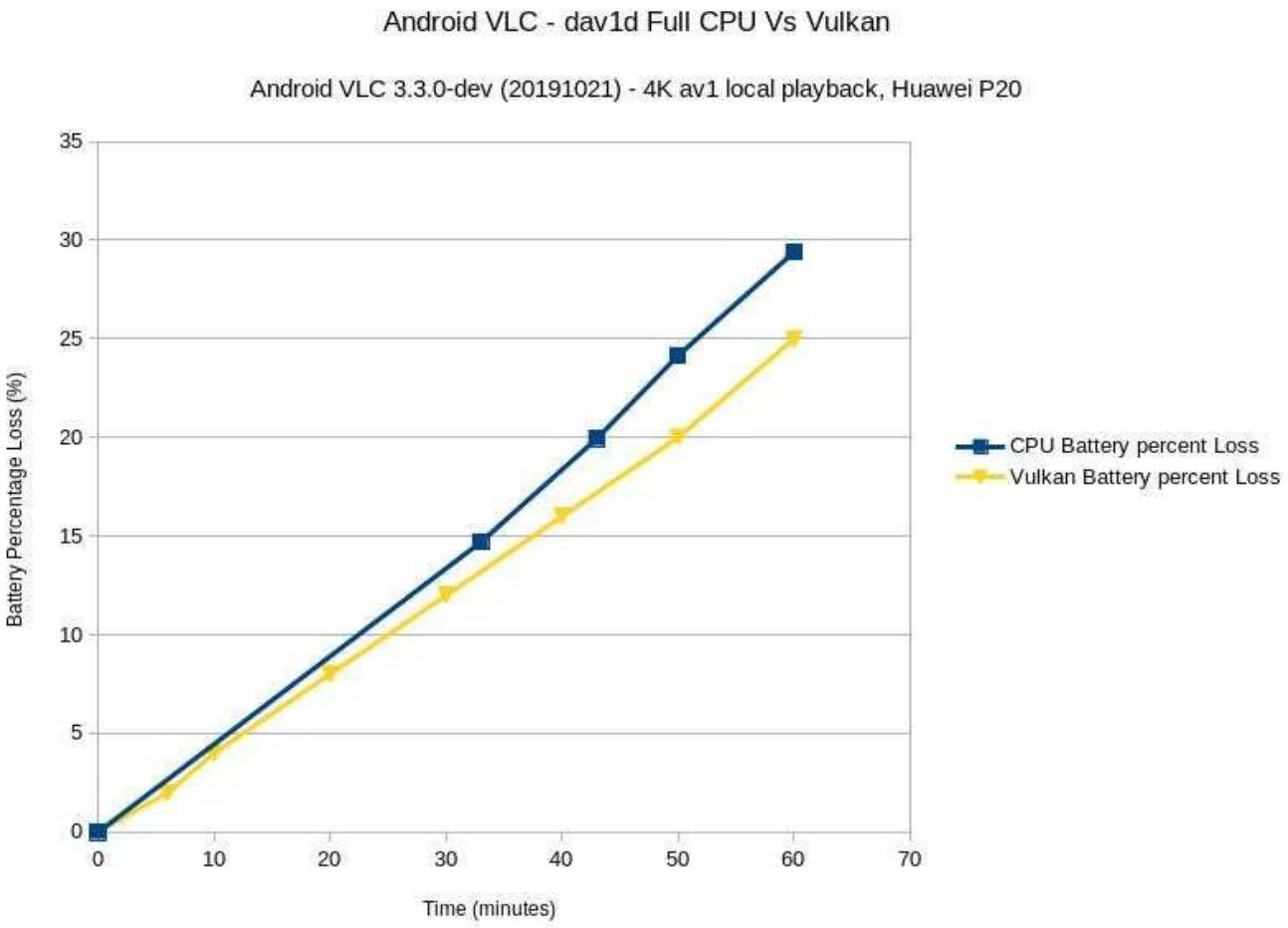
- Vulkan Shaders
- Android only

Done:

- Loop Restoration (SGR, Wiener)
- CDEF
- Film Grain in GLSL

Future:

- Finish?



Future

- 10bit
  - 16bit
  - ARM64/ARM32 ongoing
  - X86 ??
- GPGPU


Thanks!
dav1d