RESEARCH · APPLE METAL · IMAGE MODELS

Image Models at Full Load: Normalisation, Rotary Embeddings and Attention Blocks on Apple Silicon

GroupNorm for whole batches · rotary embeddings from a table · Winograd for more channel counts and stride 2 · Metal FlashAttention for shared heads, head sizes 64 and 512 and a larger block

Exe — Exe AI, developer studio
Measured on Apple M2 Max (38-core GPU, 64 GB unified memory)

27×
GROUP_NORM at 1024×1024×128: 140.7 ms → 5.16 ms; every image of a batch normalised

−27 %
SDXL 1024², one image, 8 steps: release 24.08 s → 17.56 s, back to back

−14 %
FLUX.3 Klein 1024², 20 steps, whole run: release 199.6 s → 171.0 s

17,762
ggml op tests on Metal, all passed, final build `de043708`

SETUP

Start point	ggml fork `34e0dc1c`, the end state of Report 3 · application release `master-1000-82a37b3`
End point	ggml fork `de043708` · engine `7bdf150` (submodule pointer and notes only)
Question	which part of an image model family – text encoder, sampling steps, VAE – keeps the GPU below 70 % load, and why
Target	every part at ≥ 70 % load · no model slower than before
Rules	every change in our ggml fork · stable-diffusion.cpp code unchanged · every figure from a log or raw file · model comparisons only back to back on an idle GPU (Section 9) · one run per figure unless stated · model measurements on their own sheet (Section 11)

1 Where the time went

Profiler	`GGML_METAL_OP_PROFILE=1`: GPU time per op label, summed per graph
Load of an op	max(GFLOP/s ÷ 13,500, GB/s ÷ 400), at most 100 % · reference values: 400 GB/s is the M2 Max memory bandwidth, 13.5 TFLOP/s the order of the measured matrix-product ceiling (Report 3: 12.3–13.2)
Load of a part	time-weighted mean over the ops of the last call of that part (`nightrun/part_load.py`) · ops below 0.5 % of the time are not listed by the profiler and count as 0 · Winograd counts the arithmetic of the direct convolution
Runs	1024², 2–4 steps, application setting `--conv-direct auto,vae=on` unless stated

GRAPH	GPU TIME	LARGEST ITEMS
SDXL, one UNet call, `diffusion=on`	1,059 ms	MUL_MAT f16 40.1 % (10.2–10.4 TFLOP/s) · CONV_2D 21.4 % · **GROUP_NORM 13.7 % (9 GB/s)** · FLASH_ATTN_EXT 10.3 % · ADD 4.5 %
SDXL, one UNet call, application setting	–	MUL_MAT f16 53.3 % · **GROUP_NORM 13.2 %** · FLASH_ATTN_EXT 10.0 % · IM2COL 8.3 %
SDXL, VAE decode	3,319 ms	**GROUP_NORM 57.4 % (1,904 ms, 8 GB/s)** · CONV_2D 24.0 % · **FLASH_ATTN_EXT 11.3 % (1 call, head size 512, 1.5 TFLOP/s)**
Krea 2, one DiT call	11.6–11.8 s	MUL_MAT f16 78.9 % (10.7–11.0 TFLOP/s) · FLASH_ATTN_EXT 12.0 % (8.1–8.4 TFLOP/s) · **CONT f32 order 3012: 3.2 %, 20 GB/s**
Qwen-Image 2.1, one DiT call	7.3–7.6 s	MUL_MAT f16 69 % (11.0–11.3 TFLOP/s) · FLASH_ATTN_EXT 14.2 % · **CONT f32 order 3012: 5.8 %, 21 GB/s**
FLUX.3 Klein, one DiT call	4.49 s	MUL_MAT q8_0 73 % (load 76 %) · **FLASH_ATTN_EXT 20 % (load 58 %)** · ROPE_APPLY 0.8 %

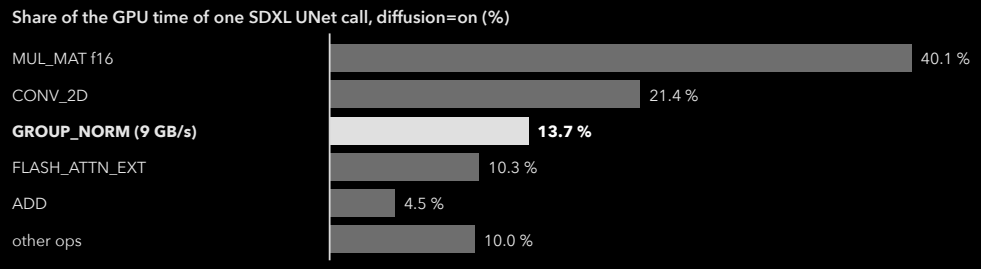


Figure 1. Op profile of one SDXL UNet call at 1024², ggml `571ed8aa`, `measurements/sdxl-profil-…/`. One run.

- Matrix products run at 76–83 % of the reference in every family – the last 10 %, not the first lever.
- GROUP_NORM: 13 % of an SDXL step, 57 % of the SDXL decode, at 8–9 GB/s (Section 2).
- Rotary embedding: a permuted copy at 20 GB/s, 112 calls per Krea 2 DiT call (Section 5).
- Attention: 10–20 % of every step at 54–61 % load; Krea 2 did not run on Metal FlashAttention at all (Section 6).
- Text encoders: computing below 0.5 s per image; 14–32 s are reading the weights from disk (16.0 s of 16.5 s for Krea 2, 32.4 s of 33.4 s for Qwen) – not a kernel matter.
- Logs: `measurements/sdxl-profil-…/`, `profil-…-krea-qwen/`, `profil-…-flux/`.

2 GroupNorm for whole batches

- Cause (upstream code, checked against ggml-org/llama.cpp): `ggml_metal_op_group_norm` starts one threadgroup of **32 threads** per group; the loop for more threads is commented out → 32 groups × 32 threads = 1,024 threads for the whole GPU.
- Bug found: the Metal kernel normalises only the first image of a batch (`ne03` is never read; the CPU reads it). New tests with N = 2 and N = 3 fail (error 1.0 and 1.6).
- Consequence: sd-cli's default batched CFG (condition and uncondition as a batch of 2) gives a **white SDXL and SD 1.5 image** on ggml's kernel (Figure 5) – the reason the application passes `--batched-cfg off`.

- Change (27e68125): heavy_metal_exe_group_norm – up to 1,024 threads per (group, image) · statistics in one read pass (shifted sums, shift = first value of the group) · a second pass writes · float4 where possible · the last group divided by its own size, as on the CPU · ggml's kernel with GGML_METAL_GROUP_NORM_KERNEL=ggml.

GROUP_NORM, 32 GROUPS, TEST-BACKEND-OPS PERF	GGML	HEAVY_METAL_EXE_GROUP_NORM	FACTOR
128×128×320 (SDXL UNet)	5.00 ms	0.20 ms	25×
64×64×640	2.25 ms	0.070 ms	32×
32×32×1280	0.92 ms	0.043 ms	21×
32×32×2560	2.23 ms	0.073 ms	31×
128×128×512 (VAE)	8.69 ms	0.32 ms	27×
512×512×256 (VAE)	70.6 ms	2.60 ms	27×
1024×1024×128 (VAE)	140.7 ms	**5.16 ms**	**27×**

- Memory traffic at 1024×1024×128: ≈ 305 GB/s (two reads, one write).
- A first try (not built in): ggml's kernel with 1,024 threads and the missing barrier between its two reductions – 140.7 → 8.0 ms.
- Tests: GROUP_NORM 6/6 (ggml's kernel 4/6) · all ops 17,729/17,729 · review: no finding.
- In the model: Section 11, rows G1–G3.

3 Winograd for more channel counts

- Cause: the Winograd gate of Report 3 asked for C_{in} % 64 and C_{out} of 128 or 48 → SDXL's 320 channels ran on the direct kernel.
- Change (c26f73ce): C_{in} % 8 (the last K block ends at any multiple of 8) · a new 64×64 GEMM tile for C_{out} % 64; it wins over the 48 tile by 2–4 % where both fit (576: 21.7 against 22.5–22.7 ms; 192: 14.0 against 14.3–14.5 ms).
- Names (e2d92f02): the kernels are named after their method – heavy_metal_exe_winograd (was wino) and heavy_metal_exe_gemm (was exe); the old values of GGML_METAL_CONV3D_KERNEL keep working.

3×3, F16 WEIGHTS, MEAN OF TWO RUNS	IM2COL + MUL_MAT	HEAVY_METAL_EXE_GEMM	WINOGRAD NEW	AGAINST
SDXL 320→320, 128²	2.81 + 2.84 = 5.65 ms	5.02 ms	**2.27 ms**	2.5× IM2COL
SDXL 640→640, 64²	1.40 + 2.97 = 4.37 ms	4.45 ms	**1.93 ms**	2.3× IM2COL
SDXL 1280→1280, 32²	0.70 + 2.97 = 3.67 ms	5.40 ms	**2.53 ms**	1.45× IM2COL
SDXL 2560→1280, 32²	1.40 + 5.98 = 7.38 ms	10.82 ms	**5.56 ms**	1.33× IM2COL
Qwen VAE 288→288, 512²	–	67.1 ms	**28.4 ms**	2.4× before
Qwen VAE 144→144, 1024²	–	94.4 ms	**40.1 ms**	2.35× before
Wan VAE 96→96, 1024²	–	34.7 ms	**22.7 ms**	1.53× before

FOUND IN THE FIRST MODEL RUNS	CAUSE	FIX (C26F73CE)
Qwen decode slower, VAE buffer almost doubled	the fast path split only by channels and images; at 1024² with 144 channels V and Y are over 1 GB each (65,536 tiles)	a frame that does not fit the scratch budget runs in tile runs (multiples of 64): t_base, tiles per run
SDXL with batched CFG 11× slower	batched CFG gives CONV_2D a batch of 2; the fast path took only N = 1, every UNet convolution fell back to ggml's one-output-per-thread kernel	a batch runs image by image, in ascending order (the scratch of image n lies above the still empty outputs n+1 …)
SDXL decoder 17 % slower at the same kernel choice	suspected: the remainder check kc in the innermost GEMM loop	GEMM as a template TAIL: full blocks unchanged, _tail variants only for C_{in} % 64 ≠ 0

- Kernel choice (log lines [conv2d]): 320→320, 960→320, 2560→1280, 640→640 on Winograd · 1×1 on heavy_metal_exe_gemm (intended) · 4→320 on ggml (C_{in} % 8 ≠ 0).

- Tests: CONV_2D 2,056/2,056 and CONV_3D 314/314, both also with `GGML_METAL_WINO_SCRATCH_MB=1` (forces tile runs); batch cases run on Winograd · all ops 17,708/17,708 · review: one finding (documentation of the old gate), fixed; after the fixes no finding.
- In the model: Section 11, rows W1–W5.

4 Stride 2 and the small convolutions

SDXL CANDIDATE, GGML `27E68125`	KERNEL	TIME	CALLS
3×3 stride 2, 320 channels, 128²	ggml	43.2 ms	2 per step
3×3 stride 2, 640 channels, 64²	ggml	42.2 ms	2 per step
VAE conv_out 128→3, 1024²	heavy_metal_exe_gemm	24.2 ms	1 per image
VAE conv_in 4→512	ggml (C_{in} % 8)	7.4 ms	1 per image
UNet conv_in 4→320	ggml (C_{in} % 8)	4.6 ms	2 per step
UNet conv_out 320→4	heavy_metal_exe_gemm	1.1 ms	2 per step

- Change (`e599acce`): a stride-2 3×3 convolution runs as a stride-1 convolution over the whole frame on `heavy_metal_exe_winograd`; the output stage writes only even rows and columns. 4× the arithmetic, but **43.2 → 2.05 ms** and **42.2 → 2.23 ms**.
- Tests: 4 new cases (padding 1 and 0, odd sizes, batch) · CONV_2D 2,060/2,060.
- Left out: the four small convolutions, together ≈ 0.1 s per image (≈ 0.5 %).

5 Rotary embeddings from a table

- Cause: stable-diffusion.cpp applies RoPE from a precomputed table (`src/model/common/rope.hpp:1075–1131`). ggml has the op `GGML_OP_ROPE_APPLY` for CPU, CUDA and Vulkan, **not for Metal** → `ggml_backend_supports_op` says no and the fallback runs: permute with copy (order 3012), two REPEAT, MUL, ADD.
- Count: 112 copies per Krea 2 DiT call = 56 RoPE calls (q and k per block).
- Users: Flux, Qwen-Image, Krea 2, Z-Image (interleaved pairs), Anima and MiniMax H3 (split pairs), LTX (per block).
- Change (`b100353a`): `heavy_metal_exe_rope_apply`, both pair layouts – interleaved (2j, 2j+1) and split (j, j + d/2); arithmetic as the CPU version (`ggml_compute_forward_rope_apply`).
- Kernel: one thread per pair · one threadgroup (up to 256 threads) covers the rows of several tokens of one head → reads of x and writes of dst contiguous along a row · any strides for x and the table; packed rows (the usual case) read a pair as float2 and its 2×2 rotation as float4.
- Switch: `GGML_METAL_ROPE_APPLY=0` reports the op as unsupported → stable-diffusion.cpp's fallback, for comparisons.

128 × 24 × 4096 (DIT AT 1024²), `TEST-BACKEND-OPS PERF`	INTERLEAVED	SPLIT
first version (single values, tokens per threadgroup)	602 µs, 162 GB/s	577 µs, 169 GB/s
float2 / float4 on packed rows (built in)	**483 µs, 202 GB/s**	**500 µs, 195 GB/s**
heads instead of tokens per threadgroup (rejected)	508 µs, 192 GB/s	522 µs, 187 GB/s

- Comparison: ggml's f32 copy of the same size (64 MB, permuted) 245 GB/s, large copies 330 GB/s · the fallback's copy alone took 3.4 ms per call at 20 GB/s · further tuning: < 0.1 % per step.
- Images not bit-identical: the Metal version fuses multiply and add (FMA); 57.9 dB and 60.8 dB.
- Tests against the CPU: ROPE_APPLY 12/12 – both layouts × {128 × 24 × 64; 64 × 5 × 33 with batch 2 and permuted x; 6 × 3 × 7 with batch 3; 1024 × 1 × 5 (more pairs than threads); permuted table (general path) twice}.
- In the model: Section 11, rows R1–R2.

6 Metal FlashAttention for shared heads and head sizes 64 and 512

Shapes	head size 128, no mask: Z-Image, Qwen-Image, Wan, LTX, MiniMax, LingBot, MOSS, Ming, Hunyuan · shared K/V heads: Krea 2 (48 on 12), Iris (20 on 5) · head size 64: SDXL UNet · 40/80/160: SD 1.5 · 512: SD/SDXL VAE (`krea2.hpp:32–41`, `iris.hpp:13–17`, `unet.hpp:23–55`)

Generator	Metal FlashAttention as shipped in ccv, commit `3564f9f`; it rebuilds the 128 source and library of Report 3 byte-identical

6.1 Shared K/V heads

- Cause: the gate asked for `q->ne[2] == k->ne[2]`; the kernel itself knows shared heads (`H_Hk_ratio`) but is started with one head per batch entry.
- Change (`c26f73ce`, host code only): gate on a whole number of query heads per K/V head · one dispatch per K/V head over its query heads, K/V batch stride 0 · kernel and metallib unchanged.
- Tests: 4 new cases on the MFA path (4:1, 4:1 with batch, permuted, non-view 3:1) · FLASH_ATTN_EXT 5,001/5,001.

KREA 2 SHAPE, 48 QUERY HEADS ON 12 K/V HEADS	GGML	MFA	FACTOR
1,024 tokens	4.46 ms	**3.31 ms**	1.35×
4,096 tokens	68.9 ms	**54.4–54.9 ms**	1.26×

6.2 Head sizes 64 and 512

- Generator range (checked with DeepWiki on liuliu/ccv, `lib/nnc/mfa/v2/AttentionDescriptor.cpp`): table rows on M1/M2 up to D = 384; for a larger D it takes the last row.
- Change: `build_mfa.sh` builds every head size into one library `fa_mfa.metallib` (`kernel_flash_attn_ext_mfa_dk128`, `_dk64`, `_dk512`); the sources differ only in D · gate: head size 128, 64 or 512, K and V as large as Q (`571ed8aa`, `e599acce`).

SHAPE, BLOCK 32/80/16 OF REPORT 3	GGML	MFA
SDXL self, 10 × 64, 4,096 tokens	7.87 ms	**6.42 ms** (6.7 TFLOP/s)
SDXL cross, 10 × 64, 4,096 → 77	0.35 ms	**0.22 ms**
SDXL self, 20 × 64, 1,024 tokens	1.00 ms	**0.75 ms**
SDXL cross, 20 × 64, 1,024 → 77	0.18 ms	**0.07 ms**
SD/SDXL VAE, 1 × 512, 16,384 tokens	366 ms	**96.4 ms** (5.7 TFLOP/s)

- Tests: FLASH_ATTN_EXT 5,018/5,018 with head size 64 · head size 512: 49/49.
- In the model: Section 11, rows A1–A2.

7 A larger Metal FlashAttention block

Tables of the generator on M1/M2	f32 intermediates (Report 3): block 32/80/16 (queries / keys / head columns), Q not kept in registers · f16 intermediates ("mixed"): block 32/128/32, Q in registers, 8 KB threadgroup memory
f16 intermediates	S = Q·K summed in f16 (ccv's comment: a ten times larger rounding error), P in f16
Switches	`tools/gen.cpp`: `MFA_BLOCK`, `MFA_CACHE_Q`, `MFA_CACHE_O`, `MFA_S`, `MFA_P`; without them it writes the previous source byte-identical
Check per variant	rounds 1 and 2: all 5,035 FLASH_ATTN_EXT tests against the CPU, all passed · round 3: the three timing cases only, the winner then with all MFA cases

ROUND 1, KREA 2 SHAPE	BLOCK	Q IN REGISTERS	S / P	1,024 TOKENS	4,096 TOKENS
Report 3	32/80/16	no	f32 / f32	3,085 µs	52,219 µs
mixed (ccv's choice)	32/128/32	yes	f16 / f16	2,892 µs	44,337 µs
block, f32	32/128/32	yes	f32 / f32	2,774 µs	42,662 µs
P in f16 only	32/80/16	no	f32 / f16	3,045 µs	51,443 µs
mixed with S in f32	32/128/32	yes	f32 / f16	2,785 µs	42,419 µs

- Finding: the gain comes from the block, not from f16 → the arithmetic stays f32, no loss of accuracy.
- Round 2 (blocks around 32/128/32, O in registers): best **32/128/32 with Q not in registers**: 2,492 µs and 38,230 µs → −19 % and −27 % against Report 3's block; Q in registers costs 11 %, Q and O 33 %.

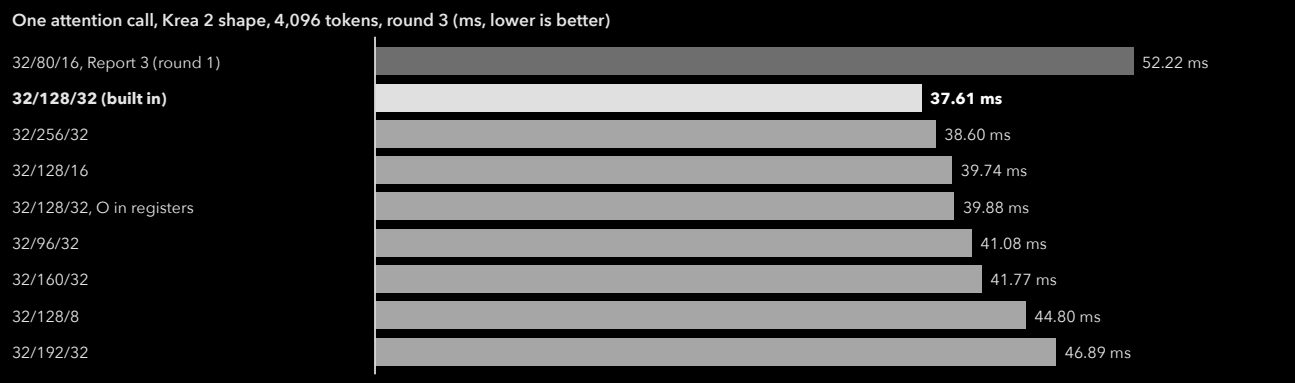


Figure 2. Round 3, `test-backend-ops perf`, all f32, Q not in registers unless stated. The Report 3 bar is from round 1. One run each.

ROUND 3, THREE CASES	1,024 TOKENS	4,096 TOKENS	32 HEADS × 14,080 TOKENS
32/128/32	**2,464 µs**	**37,608 µs**	299,075 µs
32/160/32	2,842 µs	41,769 µs	**295,685 µs**
32/256/32	2,518 µs	38,601 µs	307,814 µs

OTHER HEAD SIZES	32/80/16	32/128/32	BEST OTHER
64: 10 heads × 4,096	5,603 µs	**4,437 µs (−21 %)**	32/64/32: 4,479
64: 10 × 4,096, text 77	180 µs	**116 µs (−36 %)**	32/128/64 with Q: 97
64: 20 × 1,024	680 µs	**589 µs (−13 %)**	32/128/16: 591
512: 1 × 16,384	100,512 µs	**66,075 µs (−34 %)**	32/128/64: 70,537

- Built in (`de043708`): block 32/128/32, Q not in registers, all f32, for head sizes 64, 128 and 512 · `build_mfa.sh` sets the block; the three sources are byte-identical with the measured variants · threadgroup memory 8 KB instead of 2.5 KB.
- 32/160/32 is 1 % faster only at 14,080 tokens and 11–15 % slower at 1,024 and 4,096 → not taken.
- Tests: all 755 MFA cases of FLASH_ATTN_EXT (head size 64/128/512, no mask) · all ops 17,762/17,762 (`measurements/tests-…-de043708/`).
- In the model: Section 11, rows B1–B2.

8 Kernel load

KERNEL	SHAPE	TIME	REAL TFLOP/S	LOAD	DIRECT-EQUIVALENT
heavy_metal_exe_winograd	SDXL 320, 128²	2.27 ms	3.33	25 %	13.3
heavy_metal_exe_winograd	SDXL 640, 64²	1.93 ms	3.91	29 %	15.6
heavy_metal_exe_winograd	SDXL 1280, 32²	2.53 ms	2.98	22 %	11.9
heavy_metal_exe_winograd	SDXL 2560→1280, 32²	5.56 ms	2.72	20 %	10.9
heavy_metal_exe_winograd	VAE 288, 512²	28.4 ms	3.45	26 %	13.8
heavy_metal_exe_winograd	VAE 144, 1024²	40.1 ms	2.44	18 %	9.8
heavy_metal_exe_winograd	VAE 96, 1024²	22.7 ms	1.92	14 %	7.7
heavy_metal_exe_gemm	SDXL 320, 128²	5.02 ms	6.02	45 %	=
heavy_metal_exe_gemm	SDXL 640, 64²	4.45 ms	6.79	50 %	=
heavy_metal_exe_gemm	SDXL 1280, 32²	5.40 ms	5.59	41 %	=
kernel_flash_attn_ext_mfa, 32/80/16	Krea 2, 4,096 tokens	54.4 ms	7.58	56 %	–
kernel_flash_attn_ext_mfa, 32/80/16	Krea 2, 1,024 tokens	3.31 ms	7.79	58 %	–
ggml mul_mm (comparison)	Q8_0 matrix product	–	10.6	78 %	–

- Counting: Winograd real = 2 · 36 · tiles · C_{in} · C_{out}, tiles = (W/4)² · direct-equivalent and GEMM = 2 · W² · C_{in} · C_{out} · 9 · attention = 4 · T² · D · H · load against 13.5 TFLOP/s, batch 1.
- Winograd computes a quarter of the multiplications → at 25 % real load it is faster than `heavy_metal_exe_gemm` at 45 %. Where the rest goes (transforms, float intermediates) is not measured per stage.
- Attention with the block of Section 7: 37.6 ms at 4,096 tokens → 11.0 TFLOP/s by the same count.

9 Measurement conditions

DISTURBANCE	SEEN IN	EFFECT
Image server of the application (`sd-server`, Krea 2 F16 + Qwen3-VL 4B) left running	`footprint` 32 GB · swap 15.9 of 16 GB (`sysctl vm.swapusage`)	the unchanged release 2.29–2.37 s/it instead of 2.09–2.10 s/it; first step 13.9 s instead of 3.3 s
Open application window and WindowServer on the same GPU	`ioreg` "Device Utilization" at idle 0–99 %, mean 41 %; window process 29 % CPU	everything ≈ 10 % slower, also ops RoPE does not touch: MUL_MAT 9.4–9.8 instead of 10.7–11.0 TFLOP/s, Krea 2 VAE 1.63 instead of 1.28 s
Application closed	idle mean 10 % (WindowServer, display)	reference state for every figure of Section 11
Model cold on the external SSD	first step 12.5–12.8 s instead of ≈ 2 s	first step excluded from medians; whole-run figures only from a second round
Drift of the unchanged release between sessions	SDXL 2.135 against 2.25 s/it at the same memory state	only runs back to back are compared

- Rules adopted: every log carries the machine state (`machine_state`: other sd processes, memory, swap) · before every run the GPU is sampled for two seconds (`gpu before the run`), warning from a 2 % mean · comparisons only back to back, before and after in one session.
- Discarded: the first grouped-attention timing (beside an application run, GPU 100 %) · the absolute profile and load figures of the round with the open window (its with/without comparison stands: same conditions, back to back).

10 Open points and next steps

OPEN POINT	NEXT STEP	SECTION
Q8_0 matrix product	76 % load in FLUX.3 Klein; f16 at 76–83 % in the others – the last 10 %	1, 8
Attention	58–62 % load with the old block; FLUX.3 Klein profile with the new block not yet re-run	7, 11
Application defaults	the UNet gains (Winograd, stride 2) need `--conv-direct …,diffusion=on`; the application passes `auto,vae=on`	3, 4
Krea 2 at 512²	release 5.81 s/it, laboratory 5.45 s/it, without MFA 101.81 s sampling against 100.69 s → MFA explains ≈ 1 %; the rest of the gap is not explained	11
LTX-2.3 1280×704	+2.4–2.6 % per step lies inside the spread of the release itself (58.4–60.7 s); several runs needed	11
Families not profiled	MiniMax H3, LTX-2.3, Wan 2.2, Wan 2.1, HunyuanVideo 1.5, LingBot, MOSS-Image, Ming-Image, Iris, SD 1.5	1
Laboratory column	MiniMax H3, LTX-2.3, Qwen-Image and Krea 2 not yet run in the night-run format on the laboratory build	11
Small convolutions	C_{in} = 4 and C_{out} ≤ 4: ≈ 0.5 % per image, left out	4
Repeats	most kernel figures are two runs, most model figures one	all

Licences and acknowledgements

COMPONENT	ORIGIN
stable-diffusion.cpp, ggml, ccv's 3D convolution kernel	Report 1

COMPONENT	ORIGIN
Metal FlashAttention kernels (`kernels/mfa`), generated by the MFA generator in ccv	ccv (BSD-3-Clause, © 2010 Liu Liu) · Metal FlashAttention (MIT, © 2024 Philip Turner); both texts in `kernels/mfa/LICENSE`
`heavy_metal_exe_group_norm`, `heavy_metal_exe_rope_apply`, Winograd gate, 64 tile, tile runs, batches and stride 2, grouped-head dispatch, the generator switches, measurements, analysis	Exe AI, own work; ggml's licence (MIT) applies to the kernel files
GPL-licensed code	none used

References

1. Exe AI, *Fast 3D Convolutions for Local Video Generation on Apple Silicon*, Technical Report 1.
2. Exe AI, *Memory and Speed Together: Optimising Local Video Generation on Apple Silicon*, Technical Report 2.
3. Exe AI, *Beyond the Convolution: Attention, Small Ops and Image Models on Apple Silicon*, Technical Report 3.
4. P. Turner, Metal FlashAttention; L. Liu, ccv (`lib/nnc/mfa`, commit `3564f9f`), github.com/liuliu/ccv.
5. ggml, `GGML_OP_ROPE_APPLY` (CPU version `ggml-cpu/ops.cpp`), github.com/ggml-org/ggml.
6. Project measurements: `measurements/ab-…*/`, `measurements/profil-…-*/`, `measurements/nightrun-…/`, `measurements/tests-…-de043708/`; step log `paper/paper4-schritte.md`; overview `UEBERSICHT.html`.

11 Model measurements

Machine	Apple M2 Max, 38-core GPU, 64 GB · macOS 26.5.2 (25F84) · Xcode 26.6 (17F113) · Metal compiler 32023.883 · models on the external USB SSD
Command	`sd-cli … --steps N --sampling-method euler --flash-attn on -s 42 -W w -H h -p "$(cat measurements/prompt.txt)"` · scripts `nightrun/ab_check.py` (back to back) and `nightrun/run_matrix.py` (night-run format)
Builds	Original = stable-diffusion.cpp `master-1000-95ff762`, unmodified · Release = `master-1000-82a37b3` (ggml `34e0dc1c`) · Laboratory = `engine/build-edge`, ggml as named per row; every log names its ggml commit

MODEL	FILES
SDXL	`SDXL1.0_CyberRealisticXLPlayV10_F16.safetensors` · VAE `SDXL1.0_VAEFP16Fix_F32`
SD 1.5	`SD1.5_RealisticVisionV6.0NVB1_F16.safetensors` · VAE `SD1.5_VAEFtMSE840000_F32`
FLUX.3 Klein Base	`Flux3_KleinBase_Q_8_F16.gguf` · `Qwen3VL4B_Instruct_F16.gguf` · VAE `Flux2_VAE_F32`
Krea 2 Raw	`Krea2_Raw_F16.safetensors` · `Qwen3VL4B_Instruct_F16.gguf` · VAE `Wan2.1_HDRVAEAnimaKrea2Qwen_F32`
Qwen-Image 2.1	`QwenImage2.1_F16.gguf` · `Qwen3VL8B_Instruct_F16.gguf` + vision · VAE `QwenImage2.1_VAE_F16`
LTX-2.3	`LTX2.3_22BDistilled1.1_Q_8_F16.gguf` · Gemma 3 12B Q8_0 · BF16 connectors and VAEs

11.1 Night-run format: Original, Release, Laboratory

WHOLE RUN, 20 STEPS, BATCHED CFG	ORIGINAL	RELEASE	LABORATORY	GGML	LAB VS. RELEASE	DECODE O / R / L
SDXL 512², CFG 5	41.7 s	12.7 s	**11.7 s**	`b100353a`	−8 %	–
SDXL 1024², CFG 5	112.5 s	43.9 s	**37.8 s**	`b100353a`	−14 %	– / 3.65 / 1.43 s
SD 1.5 512², CFG 7.5	28.5 s	8.0 s	**7.0 s**	`b100353a`	−13 %	– / 1.01 / 0.54 s
FLUX.3 Klein 512², CFG 3.5	115.5 s	50.2 s	**43.7 s**	`de043708`	−13 %	13.4 / 1.02 / 0.57 s
FLUX.3 Klein 1024², CFG 3.5	465.0 s	199.6 s	**171.0 s**	`de043708`	−14 %	53.5 / 3.62 / 1.38 s

- SDXL and SD 1.5, Original and Release: white image (brightness 255, spread 0) – batched CFG on ggml's GROUP_NORM (Section 2). The laboratory image is correct; PSNR laboratory against original 3.7–4.6 dB for that reason.
- FLUX.3 Klein images: original against release 58.0 / 58.2 dB · original against laboratory 58.0 / 58.3 dB · release against laboratory 57.6 / 58.3 dB (512² / 1024²).
- Logs: `measurements/nightrun-…/{sdxl_cyberrealistic,sd15_realisticvision,flux3_klein_base}_*_{orig,ours,lab}.log`.

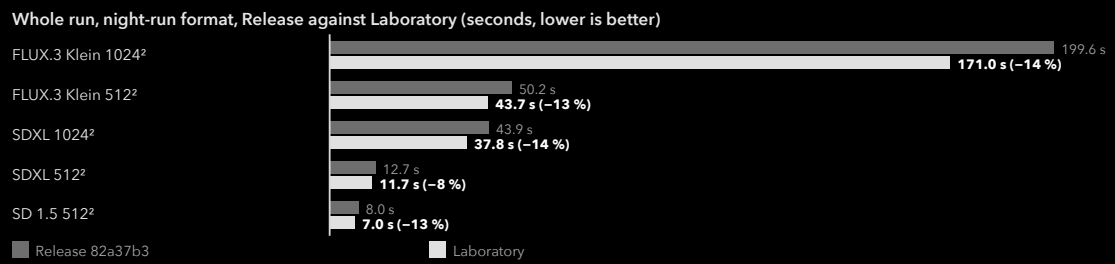
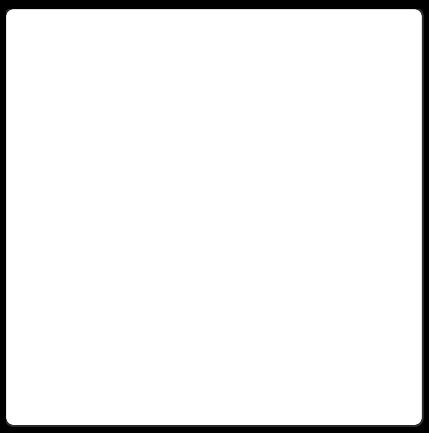
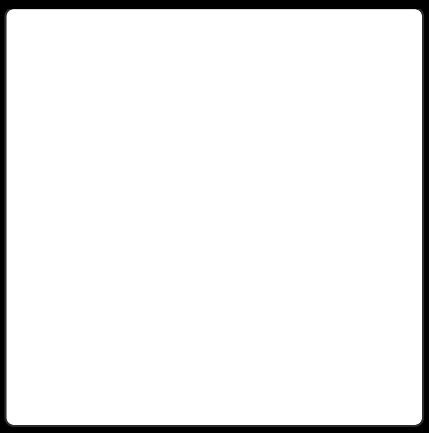


Figure 3. Whole runs from the night-run logs, one run each; ggml of the laboratory as in the table.



SDXL, Original — white



SDXL, Release — white



SDXL, Laboratory



FLUX.3 Klein, Original



FLUX.3 Klein, Release



FLUX.3 Klein, Laboratory

Figure 4. Output at 1024², night-run format, seed 42, same prompt. SDXL: batched CFG on ggml's GROUP_NORM normalises only the first image of the batch. FLUX.3 Klein: 57.6–58.3 dB between all three.

11.2 Back to back, per change

ROW · MODEL, SETTING	MEASURE	BEFORE	AFTER	IMAGE
W1 · Qwen-Image 2.1, 1024², 4 steps	decode · VAE buffer	4.32 s · 4,706 MB	**3.07 s** · 4,614 MB	76.70 dB
W2 · Krea 2 Raw, 1024², 4 steps	sampling · decode	154.07 s · 3.20 s	**143.89 s** · **2.25 s**	50.02 dB
W3 · SDXL 1024², `diffusion=on`, batched CFG	s/it	21.09	**1.85–1.87**	release white
W4 · SDXL 1024², application setting	s/it · VAE buffer	2.03 · 11,150 MB	2.03–2.04 · **4,092 MB**	–
W5 · LTX-2.3 1280×704×121, 4 steps	steps 2–4 · peak	61.79 / 58.93 / 61.42 s · 30.64 GB	62.02 / 61.45 / 63.41 s · **29.58 GB**	pixel-identical
A1 · SDXL 1024², application setting, MFA 64	s/it, median steps 2–8	2.115	**2.075** (−1.9 %)	52.4–52.9 dB
A2 · SDXL 1024², `diffusion=on`, MFA 64	s/it, median steps 2–8	2.130	**2.025** (−4.9 %)	
G1 · SDXL 1024², application setting, GroupNorm	s/it	2.075	**1.90**	52.77 dB; batched CFG 3.19 → 52.77 dB
G2 · SDXL 1024², `diffusion=on`, GroupNorm	s/it · decode	2.025 · 3.56 s	**1.85** · **1.84 s**	
G3 · SDXL 1024², `diffusion=on`, GroupNorm	one image	22.03 s	**19.08 s**	

ROW · MODEL, SETTING	MEASURE	BEFORE	AFTER	IMAGE
S1 · SDXL 1024², release (application setting) → laboratory `e599acce` (`diffusion=on`)	s/it · decode · one image	2.25 · 3.85 s · 24.08 s	**1.69 · 1.59 s · 17.56 s**	51.8–52.6 dB
S2 · as S1, batched CFG	s/it · one image	2.25 · 24.08 s	**1.67 · 17.19 s**	51.8–52.6 dB
R1 · Krea 2 Raw, 1024², 2 steps, RoPE	s/it	22.71	**21.58** (−5.0 %)	57.9 dB
R2 · Qwen-Image 2.1, 1024², 2 steps, RoPE	s/it	14.30	**13.22** (−7.6 %)	60.8 dB
B1 · FLUX.3 Klein, 1024², 8 steps, MFA block	s/it · decode · whole	8.61 · 1.58 s · 74.4 s	**8.29** · **1.38 s** · **70.7 s**	55.5 dB
B2 · SDXL 1024², 8 steps, `diffusion=on`, MFA block	s/it · decode	1.51 · 1.58 s	**1.46** · **1.37 s**	54.9 dB

- Before / after per row: W = release `82a37b3` against laboratory `c26f73ce` · A = laboratory `571ed8aa` with `GGML_METAL_FA_MFA=0` and without, two rounds of six runs, median · G = laboratory before and after `27e68125`, idle memory · S = release against laboratory `e599acce` · R = `b100353a` with `GGML_METAL_ROPE_APPLY=0` and without, idle GPU · B = copy of `b100353a` against `de043708`.
- Images: PSNR against the "before" run of the row (`nightrun/psnr.py`); from 50 dB no difference is visible.
- W5: +2.4–2.6 % per step lies inside the spread of the release itself (58.4–60.7 s per step); peak memory −1.06 GB.
- Logs: `measurements/ab-…/`, `ab-…-rope-2/`, `ab-…-mfa-block/`.

11.3 Load per part against the 70 % target

FAMILY	STEPS BEFORE	STEPS AFTER	VAE BEFORE	VAE AFTER	GGML AFTER
SDXL, `diffusion=on`	75 %	**77 %** (UNet call 781 ms)	82 %	**90 %** (1.11 s)	`b100353a`
SDXL, application setting	68 %	69 % (880 ms)	82 %	87 %	`b100353a`
Krea 2	73 %	**79 %** (DiT call 10.9 s)	84 %	**92 %** (1.16 s)	`b100353a`
Qwen-Image 2.1	71 %	**79 %** (6.6 s)	83 %	**86 %** (1.89 s)	`b100353a`
FLUX.3 Klein	–	**72 %** (4.49 s)	–	**86 %** (1.15 s)	`b100353a`, before Section 7

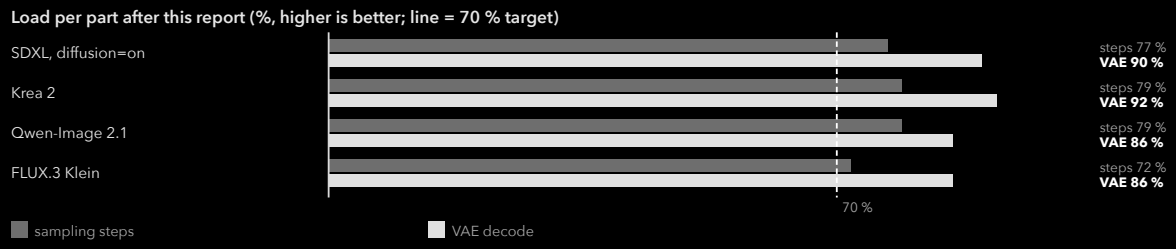


Figure 5. Load per part (Section 1), idle GPU, 1024². Text encoders below 0.5 s of computing per image, not shown. One profile each: `measurements/profil-…-sdxl-3/`, `profil-…-rope-2/`, `profil-…-flux/`.

- "Before" from the profiles on `e599acce` (`profil-…-sdxl-neu/`, `profil-…-krea-qwen/`); "after" includes RoPE and an idle GPU.
- Below target: SDXL in the application setting (69 %, IM2COL 9 % of the time at 14 % load – gone with `diffusion=on`).