ROCm vs CUDA for Local AI in 2026: The Software Gap Nobody Talks About

March 3, 2026 · by Mark Bartlett

Download this guide as PDF

> **Quick Answer:** NVIDIA CUDA extracts roughly 0.13 tok/s per GB/s of bandwidth on Llama 3 8B Q4. AMD ROCm gets about 0.06 tok/s per GB/s -- a 2x efficiency penalty from less-optimized kernels. The hardware is competitive (AMD often has more VRAM per dollar), but the software gap means you're paying for bandwidth you can't fully use. ROCm 7.2 on RDNA 3 (RX 7900 series) works well enough for daily use on Linux. RDNA 4 (RX 9070) is a mess right now -- Vulkan actually beats ROCm HIP by 14-30% on those cards. Buy NVIDIA if you want it to work. Buy AMD if you want the VRAM and accept the tradeoffs.

More on this topic: AMD vs NVIDIA for Local AI | ROCm GPU Detection Fix | GPU Buying Guide | VRAM Requirements

AMD's specs look great on paper. The RX 7900 XT has 800 GB/s bandwidth and 20GB VRAM for $600 used. The RTX 3090 has 936 GB/s and 24GB for $1,040. Competitive hardware, right?

Then you run Llama 3 8B Q4 and the 7800 XT gets 39 tok/s from its 624 GB/s. An RTX 3060 12GB – a $275 card with 360 GB/s – gets 51 tok/s.

That's not a hardware problem. That's a software problem. And it's the thing AMD marketing will never put on a slide.

The numbers

LLM token generation is bandwidth-bound. More bandwidth should mean more tokens per second. It doesn't:

GPU	VRAM	Bandwidth	Llama 3 8B Q4	tok/s per GB/s	Used Price
RTX 3060 12GB	12GB	360 GB/s	51 tok/s	0.14	$275
RTX 3080 12GB	12GB	912 GB/s	107 tok/s	0.12	$305
RTX 3090 24GB	24GB	936 GB/s	112 tok/s	0.12	$1,040
RTX 4060 Ti 16GB	16GB	288 GB/s	48 tok/s	0.17	$430

GPU	VRAM	Bandwidth	Llama 3 8B Q4	tok/s per GB/s	Used Price
RX 7800 XT 16GB	16GB	624 GB/s	39 tok/s	0.06	$465
RX 7900 XT 20GB	20GB	800 GB/s	~45 tok/s*	0.06	$600

*The 7900 XT gets 116 tok/s on Llama 2 7B Q4, but Llama 3 8B Q4 runs slower due to the larger vocabulary and GQA attention. 97 tok/s sustained on Llama 2, roughly 45 tok/s estimated on Llama 3 based on the same ~0.06 efficiency ratio.

NVIDIA averages 0.13 tok/s per GB/s of memory bandwidth. AMD averages 0.06. That's a 2x software efficiency gap.

The RTX 3060 – a card that costs $190 less than the RX 7800 XT – generates more tokens per second despite having 42% less bandwidth. Not because NVIDIA hardware is better. Because CUDA kernels extract more from the hardware they have.

Why the gap exists

CUDA has 18 years of kernel optimization behind it. cuBLAS, cuDNN, and FlashAttention are hand-tuned for every NVIDIA architecture since Kepler. When you run llama.cpp or Ollama on NVIDIA, you're running matrix multiplications on kernels that have been profiled and optimized by thousands of developers.

ROCm's equivalent libraries (rocBLAS, MIOpen, hipBLAS) work. They're not bad. They're just younger and less optimized for consumer GPU architectures. AMD has focused most of its ROCm effort on data center chips (MI300X, MI250X) where the money is. Consumer RDNA GPUs get the trickle-down.

The other piece: CUDA has compute kernels tuned for each specific GPU architecture. ROCm often uses generic kernels that work across architectures but don't fully exploit any of them. On RDNA 3, this costs you maybe 15-20% versus a well-tuned implementation. On RDNA 4, it's worse.

RDNA 4 and the Vulkan problem

AMD's RX 9070 and 9070 XT shipped in early 2026. RDNA 4, 16GB VRAM, 608 GB/s bandwidth, $549-599 MSRP. ROCm 7.2 officially supports them. Sounds good.

Benchmarks on the RX 9070 XT in llama.cpp tell the story:

Model (Q8_0)	ROCm HIP	Vulkan	Vulkan advantage
Llama 3.1 8B	60.9 tok/s	69.2 tok/s	+14%
Qwen3 8B	58.3 tok/s	66.2 tok/s	+13%
Mistral 7B	63.3 tok/s	72.6 tok/s	+15%
GPT-OSS 20B	117.2 tok/s	152.7 tok/s	+30%

Vulkan – the generic, cross-platform graphics API – beats AMD's own dedicated compute stack on AMD's newest GPU. For prompt processing, the gap is even worse: Vulkan hits 2,888 tok/s on 512-token prompts where ROCm HIP manages 1,149. That's 2.5x slower on their own hardware.

The root cause is a Wave32 vs Wave64 mismatch. RDNA 4 consumer GPUs execute in Wave32 (32 threads per wavefront). But ROCm's HIP libraries were optimized for CDNA data center chips that use Wave64. The kernels run, but they're not exploiting the hardware correctly. Flash Attention doesn't even compile with the default Composable Kernel backend on RDNA 4 – you have to switch to the Triton backend manually.

There's also an idle power bug: running llama.cpp with the HIP backend on RDNA 4 locks the GPU at elevated clocks until you kill the process entirely. It's tracked in ROCm issue #5706 but not fixed yet.

For now, if you buy an RX 9070, use the Vulkan backend in llama.cpp instead of ROCm HIP. You'll get better performance with fewer bugs. That's an embarrassing sentence to write about AMD's own compute platform, but the benchmarks are clear.

What actually works on AMD right now

ROCm 7.2.0 on RDNA 3 (RX 7000 series) is in decent shape. The realistic compatibility picture:

Works well:

- Ollama on Linux with RX 7800 XT, 7900 XT, 7900 XTX
- llama.cpp with HIP backend on RDNA 3
- PyTorch 2.10 with ROCm (auto-detects correct wheel since 2.9)
- vLLM with ROCm (93% of test suites pass, considered first-class)

Works with effort:

- Ollama on RDNA 4 (not officially listed yet, community fork ollama-for-amd fills the gap)
- Flash Attention on RDNA 3/4 (needs `FLASH_ATTENTION_TRITON_AMD_ENABLE=TRUE`)

- Older RDNA 2 cards (RX 6000 series) via `HSA_OVERRIDE_GFX_VERSION=10.3.0`

Doesn't work or not worth it:

- Windows ROCm for LLM inference (limited, poorly documented)
- RDNA 4 with ROCm HIP in llama.cpp (use Vulkan instead)
- Any AMD GPU older than RX 6000 series
- Multi-GPU setups with mixed AMD generations

Supported operating systems for consumer GPUs: Ubuntu 24.04.3, Ubuntu 22.04.5, RHEL 10.1, RHEL 9.7. That's it. No Fedora, no Arch, no Manjaro in the official support matrix.

AMD compatibility checklist

Run through this before buying an AMD GPU for local AI:

1. **Are you on Linux?** If not, stop here. Windows ROCm support for consumer GPUs is not worth the pain.
2. **Ubuntu or RHEL?** Other distros work but you're on your own for troubleshooting.
3. **RDNA 3 or newer?** The RX 7800 XT, 7900 XT, and 7900 XTX are the safe picks. RDNA 2 requires workarounds. RDNA 4 has immature software.
4. **Can you use Ollama or llama.cpp?** These have the best AMD support. If your workflow depends on a framework that only supports CUDA, AMD won't work.
5. **Are you comfortable debugging?** Even in 2026, you'll occasionally hit a driver issue, a kernel panic, or a model that doesn't load. NVIDIA users don't deal with this.
6. **Do you need the VRAM?** The only good reason to buy AMD over NVIDIA is getting more VRAM per dollar. If a 12GB NVIDIA card fits your models, buy NVIDIA.

When to buy AMD vs NVIDIA

Buy AMD if:

- You need 16-24GB VRAM on a budget (RX 7900 XT 20GB at $600 vs RTX 3090 24GB at $1,040)
- You're on Linux and comfortable with command-line troubleshooting
- You run models that are bandwidth-bound and can tolerate the efficiency gap
- You want to run 30B+ models that don't fit in 12GB VRAM

Buy NVIDIA if:

- You want it to work immediately on any OS
- You care about tok/s more than VRAM capacity
- You use PyTorch-based training workflows (CUDA is still faster for training, not just inference)
- You're on Windows
- Your budget allows an RTX 3090 ($1,040 used, 24GB, still the best value NVIDIA card for local AI)

The recommendation

For most people reading this: **buy NVIDIA**. The RTX 3090 at $1,040 used gives you 24GB VRAM and 112 tok/s on Llama 3 8B. The RTX 3080 12GB at $305 gives you 107 tok/s if you don't need more than 12GB. These cards just work.

If you're budget-constrained and need VRAM, the **RX 7900 XT at $600 used** is the AMD pick. 20GB VRAM, 800 GB/s bandwidth, and ROCm 7.2 handles it well on Linux. You'll get about half the per-bandwidth efficiency of NVIDIA, but you'll be able to load models that won't fit on a 12GB card.

Skip the RX 9070 for now. The hardware is there (16GB, 608 GB/s) but ROCm is underperforming Vulkan on it. Wait 6 months for AMD to sort out the RDNA 4 kernel optimization. By then, the benchmarks will tell us whether RDNA 4's 4x AI compute improvement actually translates to inference speed or just marketing slides.

The 2x software efficiency gap is real and it's not closing fast. AMD's hardware team keeps delivering. Their software team keeps being 18 months behind. Until that changes, CUDA is still the default for good reason.

Related guides: AMD vs NVIDIA for Local AI | ROCm GPU Detection Fix | Best GPU Under $500 | Used GPU Buying Guide | VRAM Requirements | What Can You Run on 16GB VRAM

Get notified when we publish new guides.

Subscribe — free, no spam

Source: https://insiderllm.com/guides/rocm-vs-cuda-local-ai-2026/