ROCm Not Detecting GPU: AMD Troubleshooting Guide

February 18, 2026 · by Mark Bartlett

Download this guide as PDF

> **Quick Answer:** First: check if your GPU is supported — run rocminfo and look for your agent. If nothing shows, your user isn't in the render and video groups (sudo usermod -a -G render,video $USER, then re-login). If your GPU is RDNA 2 (RX 6600/6700/6800), you need HSA_OVERRIDE_GFX_VERSION=10.3.0 to trick ROCm into recognizing it. RX 5000 and older won't work. For Ollama, check that ROCm 6.x-7.x is installed and the systemd service has the right environment variables.

📚 **More on this topic:** AMD vs NVIDIA for Local AI · GPU Buying Guide · VRAM Requirements · Planning Tool

AMD GPUs offer more VRAM per dollar than NVIDIA. The RX 7900 XTX gives you 24GB for $700-950 new. But getting ROCm working is where the savings get spent — in hours instead of dollars.

This is the guide you wish existed before you started. Every common error, what it means, and how to fix it.

Step 1: Is Your GPU Even Supported?

This is the first thing to check. Not every AMD GPU works with ROCm.

Officially Supported (ROCm 7.x)

GPU	Architecture	GFX ID	VRAM	Status
RX 9070 XT / 9070	RDNA 4	gfx1201	16 GB	ROCm 7.2+ (rocky at launch)
RX 7900 XTX	RDNA 3	gfx1100	**24 GB**	**Best supported**
RX 7900 XT	RDNA 3	gfx1100	20 GB	Full support
RX 7900 GRE	RDNA 3	gfx1100	16 GB	Full support
RX 7800 XT	RDNA 3	gfx1101	16 GB	Full support

GPU	Architecture	GFX ID	VRAM	Status
RX 7700 XT	RDNA 3	gfx1101	12 GB	ROCm 6.4.2+
RX 7600	RDNA 3	gfx1102	8 GB	Full support

Works with HSA_OVERRIDE Hack

GPU	GFX ID	Override Value	Notes
RX 6900 XT / 6800 XT / 6800	gfx1030	`10.3.0`	HIP SDK supported, override usually not needed
RX 6700 XT / 6750 XT	gfx1031	`10.3.0`	No Tensile library, override required
RX 6600 / 6600 XT	gfx1032	`10.3.0`	Same — override required
RX 6500 XT	gfx1034	`10.3.0`	Very limited VRAM (4GB), barely useful
Ryzen APUs (780M, 890M)	gfx1103/ gfx1150	`11.0.0`	Shared system RAM, slow but works

Won't Work

- **RX 5000 series (RDNA 1)** — no ROCm support, no workaround
- **RX 500/400 series (Polaris)** — too old
- **Most pre-2020 APUs** — no compute support

If your GPU isn't in these tables, it probably won't work with ROCm. Check `lspci | grep VGA` to confirm which GPU you have.

Step 2: Installation Order Matters

ROCm installation has a specific order. Getting it wrong causes cascading failures.

The correct order:

1. Install kernel headers matching your running kernel
2. Install ROCm packages
3. Add your user to `render` and `video` groups
4. Reboot

5. Verify with `rocminfo`
6. Install PyTorch/Ollama/llama.cpp after ROCm is verified

Ubuntu 22.04 / 24.04

```
# 1. Kernel headers
sudo apt install linux-headers-$(uname -r)

# 2. Add ROCm repo (ROCm 7.2, Ubuntu 24.04 — use 'jammy' for 22.04)
sudo mkdir --parents --mode=0755 /etc/apt/keyrings
wget https://repo.radeon.com/rocm/rocm.gpg.key -O - | \
    gpg --dearmor | sudo tee /etc/apt/keyrings/rocm.gpg > /dev/null

echo 'deb [arch=amd64 signed-by=/etc/apt/keyrings/rocm.gpg] https://repo.radeon.com/rocm/apt/7.2
    sudo tee /etc/apt/sources.list.d/rocm.list

# 3. Install
sudo apt update && sudo apt install rocm

# 4. User groups
sudo usermod -a -G render,video $USER

# 5. Reboot
sudo reboot

# 6. Verify
rocminfo
```

If `rocminfo` lists your GPU agent, ROCm is working. If it shows nothing or only the CPU agent, see the errors below.

Step 3: Common Errors and Fixes

“No AMD GPU detected” / rocminfo shows no agents

Check permissions first:

```
groups
# Must include 'render' and 'video'
# If not: sudo usermod -a -G render,video $USER && logout
```

Check if the amdgpu driver is loaded:

```
lsmod | grep amdgpu
# Should show amdgpu module loaded
# If empty: sudo modprobe amdgpu
```

Check /dev/kfd exists:

```
ls -la /dev/kfd
# Should exist with crw-rw---- permissions
# If missing: ROCm kernel driver not installed, reinstall rocm
```

iGPU stealing focus: If you have a Ryzen APU alongside a discrete GPU, ROCm might detect the iGPU instead. Disable the iGPU in BIOS, or use `ROCR_VISIBLE_DEVICES=1` to force the discrete card.

"HIP runtime error" / Version Mismatch

```
hipErrorNoBinaryForGpu: Unable to find code object for all current devices
```

ROCm version doesn't match the PyTorch or Ollama build. Common after upgrading ROCm without rebuilding your tools.

Fix: Reinstall the ROCm-compatible version of your tools:

```
# PyTorch for ROCm 7.2
pip install torch --index-url https://download.pytorch.org/whl/rocm7.2

# Ollama: reinstall from latest release
curl -fsSL https://ollama.com/install.sh | sh
```

After upgrading from ROCm 6.x to 7.x, library symlinks can break. Reinstall the `rocm` meta-package to fix:

```
sudo apt install --reinstall rocm
```

Ollama Not Using AMD GPU

Run `ollama ps` — if it shows CPU compute, Ollama isn't finding ROCm.

Ollama's supported GPU list is hardcoded: `gfx900, gfx906, gfx908, gfx90a, gfx940, gfx941, gfx942, gfx1030, gfx1100, gfx1101, gfx1102`. If your GPU isn't in this list (e.g., gfx1032 for RX 6600), Ollama defaults to CPU.

Fix for unsupported GPUs: Use the HSA_OVERRIDE hack (next section) or use the community build ollama-for-amd which has an expanded GPU list.

llama.cpp ROCm Build Fails

HIP_PATH not set:

```
HIPCXX="$(hipconfig -l)/clang" HIP_PATH="$(hipconfig -R)" \
cmake -B build -DGGML_HIP=ON -DAMDGPU_TARGETS=gfx1100
cmake --build build --config Release -j $(nproc)
```

hipBLAS API error (ROCm 7.x): If you see errors about `hipblasGemmEx` or deprecated types, make sure you're using a recent llama.cpp version (July 2025+) that has updated HIP API calls. Pull the latest:

```
git pull && git submodule update --init --recursive
rm -rf build && cmake -B build -DGGML_HIP=ON
```

Set your target architecture explicitly:

```
# Single GPU
-DAMDGPU_TARGETS=gfx1100
```

```
# Multiple (builds kernels for each)
-DAMDGPU_TARGETS="gfx1030;gfx1100;gfx1101"
```

Omitting `AMDGPU_TARGETS` builds for all detected GPUs, which is slower to compile but works.

Step 4: The HSA_OVERRIDE Hack

If your GPU isn't officially supported but is in the same architecture family as one that is, you can trick ROCm into treating it as the supported sibling.

```
# For RDNA 2 cards (RX 6600/6700 series) — pretend to be gfx1030
export HSA_OVERRIDE_GFX_VERSION=10.3.0

# For RDNA 3 APUs — pretend to be gfx1100
export HSA_OVERRIDE_GFX_VERSION=11.0.0
```

For Ollama, add it to the systemd service:

```
sudo systemctl edit ollama
# Add under [Service]:
# Environment="HSA_OVERRIDE_GFX_VERSION=10.3.0"
sudo systemctl restart ollama
```

Risks:

- **It usually works** for cards in the same generation. gfx1031 → gfx1030 is safe. gfx1032 → gfx1030 is safe.
- **Cross-generation overrides are risky.** Don't override an RDNA 2 card to gfx1100 (RDNA 3) — the instruction sets differ.
- **Known ROCm 6.4.3+ regression:** The override causes SIGSEGV crashes on gfx1031/gfx1032 with ROCm 6.4.3+. If you hit this, downgrade to ROCm 6.4.1.

Step 5: Performance Expectations

ROCm works. It's not as fast as CUDA. Here's what to expect:

Model	RX 7900 XTX (ROCm)	RTX 4090 (CUDA)	Gap
7B Q4	~50-55 tok/s	~75-85 tok/s	~30% slower
8B Q4	~50 tok/s	~70 tok/s	~30% slower
14B Q4	~30 tok/s	~42 tok/s	~30% slower
32B Q4	~15-18 tok/s	~22-28 tok/s	~25% slower

The gap is mostly software — CUDA has years of kernel optimization that ROCm is still catching up on. Both GPUs have similar memory bandwidth (960 GB/s vs 1,008 GB/s), so the theoretical ceiling is close. The practical gap narrows on larger models that are bandwidth-bound.

The value proposition: An RX 7900 XTX costs $700-950 new with 24GB VRAM. A used RTX 3090 costs $700-900 for the same 24GB. An RTX 4090 costs $1,600+. If you're willing to fight ROCm setup once, the AMD card delivers 70-85% of RTX 4090 speed at half the price.

Quick Reference: The 5-Minute Diagnostic

1. `rocminfo` → No agents? → **Groups/permissions** (`render` , `video`)
2. `rocminfo` shows GPU? → `ollama ps` shows CPU? → **GPU not in Ollama's allowlist** → HSA_OVERRIDE or community build
3. Model loads but runs slow? → **Check if model fits in VRAM** → Partial offload kills speed
4. Build fails? → **HIP_PATH not set** or **wrong AMDGPU_TARGETS**
5. Crashes on generation? → **ROCm version mismatch** → Reinstall matching versions

---

Bottom Line

AMD GPUs are a real option for local AI in 2026. The RX 7900 XTX with 24GB is the best value card for VRAM-per-dollar. ROCm setup is harder than CUDA — budget an extra hour the first time. Once it's working, it stays working.

If you're buying new: RDNA 3 (RX 7900 series) has the most mature ROCm support. RDNA 4 is getting there. RDNA 2 works with the override hack but expect occasional rough edges.

If you value your time over your money, NVIDIA is still the path of least resistance. If you value your money over your time, AMD rewards the patience.
