Run LLMs locally on your Ubuntu machine with integrated AMD-GPU

Why run LLMs locally?

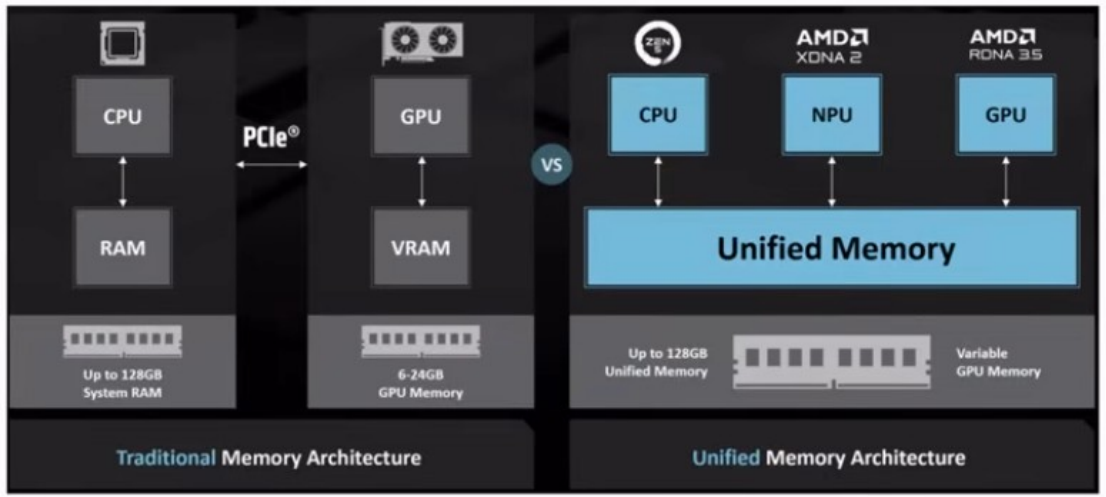
- Pros:
  - Privacy
  - Security
  - Cost control
  - Control over models used
  - Consistent quality of answers
- Cons:
  - Not as scalable as the cloud providers
  - Not suitable for big models

Requirements

- Integrated AMD GPU
  - supported by Kernel builtin **Vulkan** API driver (RADV)
  - Ryzen 7 PRO 7840U (10 TOPs)
  - Ryzen AI 9 HX 370 (50 TOPs)
  - Ryzen Al Max+ 395 (50 TOPs)
- Ram: min. 32 GB
- OS: Ubuntu 22.04+
- Kernel with GTT support (e.g. v6.8)
- Connect your laptop to a charger

Integrated NPUs and GTT

- Unified memory: memory shared between CPU and GPU/NPU
- Graphics translation table (GTT): allows the graphics card direct memory access (DMA) to the host system memory

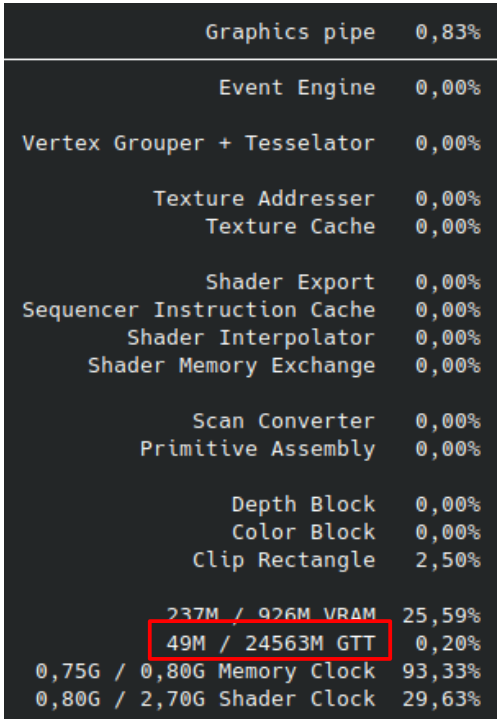


Setup Radeontop, Kernel

- Install radeontop
  - apt-get install radeontop
- Configure GTT in /etc/default/grub
  - GRUB_CMDLINE_LINUX_DEFAULT="amd_iommu=off ttm.pages_limit=6291456 ttm.pages_limit=6291456"
    (24 GB VRAM represented in KiB divided by 4)
- Update grub: sudo update-grub2
- Reboot

Check setup

Run radeontop, check GTT size:



Setup Llama.cpp

- Download llama.cpp with Vulkan support
  - e.g. llama-b6585-bin-ubuntu-vulkan-x64.zip
- Extract the archive
  - e.g. unzip llama-b6585-bin-ubuntu-vulkan-x64.zip
- Start llama-cpp server
  - ./llama-server hf <modelname> <params>
- Open your browser with: **http://127.0.0.1:8080**

Start Llama.cpp with GPT-OSS

- Start llama-cpp server and download GPT-OSS 20b
  - ./llama-server -hf unsloth/**gpt-oss-20b-GGUF:F16** \
    --jinja -ngl 99 --threads -1 --parallel **4** --ctx-size **16384** \
    --temp 1.0 --top-p 1.0 --top-k 0 --no-mmap
    --kv-unified --n_predict **4096** \
    --chat-template-kwargs '{"reasoning_effort": "**low**"}'
- Open your browser with: **http://127.0.0.1:8080**

Start Llama.cpp with Qwen3

- Start llama-cpp server and download Qwen3
  - ./llama-server -hf unsloth/**Qwen3-4B-GGUF:UD-Q4_K_XL** \
    --jinja -ngl 99 --threads -1 --parallel **4** --ctx-size **262144** \
    --temp 0.7 --top-p 0.8 --top-k 20 --presence-penalty 1.0 --no-mmap \
    --kv-unified --cache-type-k q4_0 --cache-type-v q4_0 \
    --n_predict **4096**
- Open your browser with: **http://127.0.0.1:8080**
- Add „**/nothink**“ to your prompt to disable thinking

Testing GPT-OSS

Curl:

time curl -s http://127.0.0.1:8080/v1/chat/completions \
-H "Content-Type: application/json" -H "Authorization: Bearer no-key" \
-d '{
"model": "**unsloth_gpt-oss-20b-GGUF_gpt-oss-20b-F16.gguf**", "stream": **false**,
"messages": [{
    "role": "user", "content": "**When was Beethoven born?**"
}]}' | jq .


```
{
  "choices": [
    {
      "finish_reason": "stop",
      "index": 0,
      "message": {
        "role": "assistant",
        "reasoning_content": "Need answer: 1770-12-16.",
        "content": "Beethoven was born on **16 December 1770**.'
```

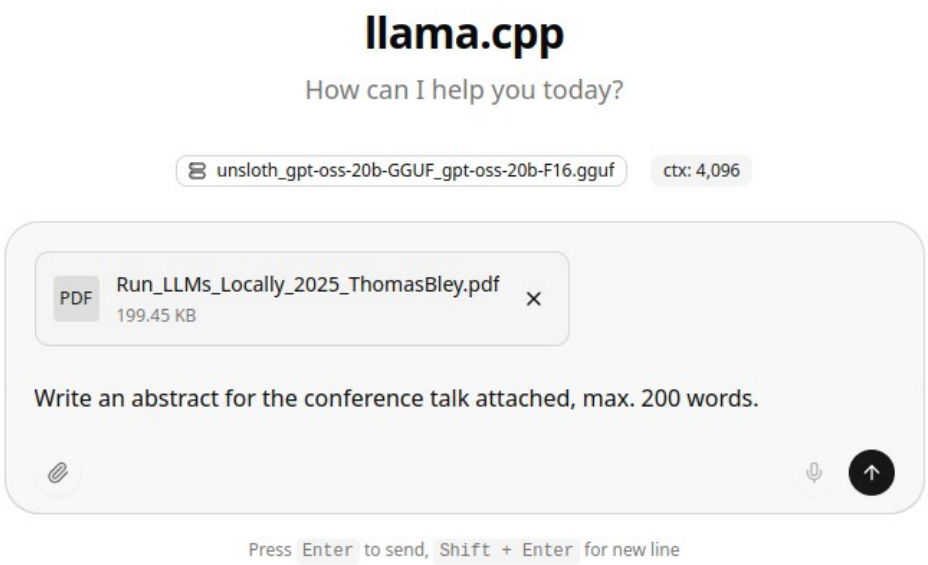
Testing Qwen3

Curl:

time curl -s http://127.0.0.1:8080/v1/chat/completions \
-H "Content-Type: application/json" -H "Authorization: Bearer no-key" \
-d '{
"model": "**unsloth_Qwen3-4B-GGUF_Qwen3-4B-UD-Q4_K_XL.gguf**", "stream": **false**,
"messages": [{
    "role": "user", "content": "**When was Beethoven born? /nothink**"
}]}' | jq .


```
{
  "choices": [
    {
      "finish_reason": "stop",
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "Ludwig van Beethoven was born on **December 16, 1770**."
```

Demo



Abstract

Large language models (LLMs) are increasingly used in enterprise and research, but cloud deployment raises privacy, security, and cost concerns. This talk demonstrates how to run state-of-the-art open-source LLMs locally on a single Ubuntu laptop equipped with an integrated AMD GPU. Using the built-in RADV Vulkan driver and the Graphics Translation Table (GTT) for unified memory, users can leverage the full GPU compute power without a discrete card. We describe the hardware prerequisites —a recent Ryzen processor with ⩾32 GB RAM, Ubuntu 22.04+, and a kernel supporting GTT—and walk through enabling GTT, installing radeontop, and configuring the kernel command line. The session then covers setting up the Vulkan-enabled llama.cpp server, downloading models such as GPT-OSS 20B and Qwen3-4B, and tuning parameters for optimal performance. Live demonstrations with curl-based API calls showcase response times and answer quality, and we discuss how GTT memory sharing eliminates the need for large VRAM. Attendees will leave with a reproducible recipe for deploying secure, cost-effective LLMs on commodity hardware, enabling privacy-preserving AI workflows in both research and industry.

Context: 2695/4096 (66%) Output: 1197/∞ 18.8 tokens/sec

Resources

- AMD Strix Halo Llama.cpp Toolboxes
- GLM 4.5-Air-106B and Qwen3-235B on AMD "Strix Halo" AI Ryzen MAX+ 395
- AMD Ryzen AI Max 395: GTT Memory Step-by-Step Instructions
- Wikipedia: Graphics Translation Table (GTT)
- unsloth: gpt-oss: How to Run & Fine-tune
- unsloth: Qwen3: How to Run & Fine-tune
- ViceVoice Text-to-Speech on Framework Desktop with Strix Halo

Thank You for listening!

Questions?

Slides:

github.com/thomas-0816/talks/