Building `llama.cpp` on Windows (Ivy Bridge, non-AVX2)

Step 0 — Environment Setup

Open **x64 Native Tools Command Prompt for Visual Studio 2019**. Optional CPU-specific flags (can also be passed directly to CMake):

```
set GGML_NATIVE=OFF
set GGML_AVX=ON
set GGML_AVX2=OFF
set GGML_F16C=ON
set GGML_FMA=OFF
```

Quick verification:

```
where cl
cl
where ninja
ninja --version
```

Navigate to project directory:

```
cd /d C:\Users\Admin\source\llama.cpp
```

Step 0.5 — Cleanup / Recovery

Before configuring or switching generators:

```
rmdir /s /q build
set CMAKE_PREFIX_PATH=
```

Step 1A — Build with MSVC (Visual Studio generator)

Configure:

```
cmake -S . -B build -A x64 -DCMAKE_BUILD_TYPE=Release ^
  -DGGML_NATIVE=OFF -DGGML_AVX=ON -DGGML_AVX2=OFF ^
  -DGGML_F16C=ON -DGGML_FMA=OFF
```

Build:

```
cmake --build build --config Release
```

Expected build time (Ivy Bridge 4c/8t): 30–60 seconds.

Step 1B — Build with Ninja

Ensure clean state:

```
rmdir /s /q build
set CMAKE_PREFIX_PATH=
```

Configure: (do not pass `-A x64`)

```
cmake -S . -B build -G "Ninja" -DCMAKE_BUILD_TYPE=Release ^
  -DGGML_NATIVE=OFF -DGGML_AVX=ON -DGGML_AVX2=OFF ^
  -DGGML_F16C=ON -DGGML_FMA=OFF
```

Build:

```
cmake --build build
```

Expected build time: 15–30 seconds.

Step 2 — Run the server

```
set MODEL=C:\path\to\your\model.gguf
.\build\bin\Release\llama-server.exe -m "%MODEL%" --port 8080 --threads 8
```

Base URL for LM Studio: `http://127.0.0.1:8080/v1`

Troubleshooting

- LNK2019 errors mentioning `json_abi_v3_12_0` vs `json_abi_v3_11_2` indicate mixed nlohmann::json versions. Solution: delete build folder, clear `CMAKE_PREFIX_PATH`, and rebuild.
- Ninja errors `CMAKE_C_COMPILER not set` usually mean you are not in the x64 Native Tools prompt or the build dir is stale — rerun cleanup step.

One-line Ninja build (copy/paste)

```
cd /d C:\Users\Admin\source\llama.cpp && rmdir /s /q build 2>nul &&
set CMAKE_PREFIX_PATH= && cmake -S . -B build -G "Ninja" -DCMAKE_BUILD_TYPE=Release ^
-DGGML_NATIVE=OFF -DGGML_AVX=ON -DGGML_AVX2=OFF -DGGML_F16C=ON -DGGML_FMA=OFF &&
cmake --build build
```

One-line MSVC build (copy/paste)

```
cd /d C:\Users\Admin\source\llama.cpp && rmdir /s /q build 2>nul &&
set CMAKE_PREFIX_PATH= && cmake -S . -B build -A x64 -DCMAKE_BUILD_TYPE=Release ^
-DGGML_NATIVE=OFF -DGGML_AVX=ON -DGGML_AVX2=OFF -DGGML_F16C=ON -DGGML_FMA=OFF &&
cmake --build build --config Release
```