Flux Locally: Complete Guide to Running Flux on Your Own GPU

January 31, 2026 · by Mark Bartlett

Download this guide as PDF

> **Quick Answer:** Flux is the best open image generation model available — better prompt following, readable text in images, and correct human anatomy compared to Stable Diffusion. You need at least 12GB VRAM to run it comfortably (RTX 3060 12GB works with quantized models). The fastest way to start: install ComfyUI, download a GGUF Q5 Flux model and the FP8 text encoder from HuggingFace, and generate your first image in about 60 seconds per image on a 12GB card. At 24GB (RTX 3090/4090), you can run Flux at full FP16 quality with no compromises.

📚 **More on this topic:** Stable Diffusion Locally · ComfyUI vs A1111 vs Fooocus · What Can You Run on 12GB VRAM · What Can You Run on 24GB VRAM · Planning Tool

Flux is a 12-billion parameter image generation model from Black Forest Labs — the company founded by the creators of Stable Diffusion. It does three things dramatically better than any Stable Diffusion model: it follows complex prompts accurately, it renders readable text inside images, and it draws human hands without extra fingers.

The catch is size. The full model needs 24GB VRAM at FP16, compared to 6.5GB for SDXL. But quantized versions bring that down to 12GB, 8GB, or even 6GB with some quality tradeoff. If you have an RTX 3060 12GB or better, you can run Flux locally right now.

This guide covers which Flux variant to use, what hardware you need, how to set up ComfyUI and Forge, and how to optimize for your specific GPU.

Flux Model Variants

Flux comes in several variants. For local use, you care about two: Dev and Schnell.

Variant	License	Quality	Steps	Use Case
Flux.1 Dev	Non-commercial	High (distilled from Pro)	20-30	Best quality for personal use

Variant	License	Quality	Steps	Use Case
Flux.1 Schnell	Apache 2.0	Good	1-4	Fast drafts, commercial use allowed
Flux.1 Pro	API only	Highest	Managed	No local weights available
Flux 2 Dev	Non-commercial	Higher	20-30	Newer, better typography and photorealism

Flux.1 Dev is what most people run locally. It produces images close to Pro quality and you can find quantized versions that fit on modest GPUs. The non-commercial license means personal use, research, and hobby projects are fine — you just can't sell the images commercially.

Flux.1 Schnell generates usable images in 1-4 steps instead of 20-30, making it roughly 5-7x faster. Quality is lower — less detail, less coherent backgrounds — but for quick iterations and commercial use, it's the only fully open option.

Flux 2 Dev (released November 2025) improves typography, photorealism, and image references over the original. If your UI supports it, it's worth trying, but Flux.1 Dev is still the most widely supported and has the largest ecosystem of LoRAs and workflows.

VRAM Requirements — What You Actually Need

The full Flux.1 Dev model at FP16 needs ~24GB VRAM. But quantized versions are significantly smaller with surprisingly small quality loss.

Flux Model Sizes by Format

Format	Model Size (Disk)	VRAM Usage	Quality vs FP16
FP16/BF16 (full)	23.8 GB	~24 GB	100% (reference)
FP8	11.9 GB	~12-16 GB	~98-99%
GGUF Q8_0	12.7 GB	~16 GB	~99%
GGUF Q6_K	9.85 GB	~12 GB	~97%
GGUF Q5_K_S	8.28 GB	~10 GB	~95%
NF4	~11 GB	~6-8 GB	~90-93%
GGUF Q4_K_S	6.80 GB	~8 GB	~90-93%

Format	Model Size (Disk)	VRAM Usage	Quality vs FP16
GGUF Q2_K	4.02 GB	~4-6 GB	~80%

You also need text encoders and a VAE:

File	Size	Notes
T5-XXL (FP16)	9.79 GB	Best quality, needs VRAM headroom
T5-XXL (FP8)	4.89 GB	Minimal quality loss, recommended
T5-XXL (GGUF Q5)	3.29 GB	Smallest, good for tight setups
CLIP-L	246 MB	Always needed, negligible size
Flux VAE	335 MB	Always needed

What to Run by GPU Tier

VRAM	Recommended Format	Total Disk Space	Expected Speed (1024x1024, Dev)
8 GB (RTX 4060, 3060 8GB)	GGUF Q4 or NF4 + CPU offloading	~10-12 GB	60-90 sec/image
12 GB (RTX 3060 12GB, 4070)	GGUF Q5_K_S + FP8 T5	~12 GB	60-80 sec/image
16 GB (RTX 4060 Ti 16GB, 4070 Ti Super)	GGUF Q8 or FP8	~18 GB	40-55 sec/image
24 GB (RTX 3090, 4090)	FP16 full	~34 GB	12-30 sec/image

The sweet spot is 12GB. A GGUF Q5 Flux model with an FP8 T5 encoder takes ~12GB of disk space and produces images that are very close to full quality. An RTX 3060 12GB handles this well.

At 24GB, you have no compromises. A used RTX 3090 at ~$750 runs Flux at full FP16 in 12-15 seconds per image. An RTX 4090 does it in under 10 seconds.

At 8GB, Flux is usable but slow. You're looking at 60+ seconds per image with aggressive quantization and CPU offloading. For 8GB cards, Stable Diffusion XL is still the better experience.

Method 1: ComfyUI (Recommended)

ComfyUI is the go-to UI for Flux. It supports every quantization format (FP16, FP8, GGUF, NF4), has the most active development, and gives you full control over the generation pipeline.

Step 1: Install ComfyUI

The easiest way is ComfyUI Desktop — a standalone installer that handles Python and dependencies for you.

Download from: https://www.comfyui.com/download

Alternatively, clone from GitHub and install manually:

```
git clone https://github.com/comfyanonymous/ComfyUI.git
cd ComfyUI
pip install -r requirements.txt
```

Step 2: Download Models

You need four files. Where to put them depends on your ComfyUI installation:

File	Download From	Put In
Flux model (pick one from the format table above)	`huggingface.co/city96/FLUX.1-dev-gguf` (GGUF) or `huggingface.co/Comfy-Org/flux1-dev` (FP8)	`ComfyUI/models/diffusion_models/`
T5-XXL text encoder	`huggingface.co/comfyanonymous/flux_text_encoders`	`ComfyUI/models/text_encoders/`
CLIP-L text encoder	Same repo as T5	`ComfyUI/models/text_encoders/`
Flux VAE (`ae.safetensors`)	`huggingface.co/black-forest-labs/FLUX.1-dev`	`ComfyUI/models/vae/`

For 12GB GPUs, download:

- `flux1-dev-Q5_K_S.gguf` (8.28 GB)
- `t5xxl_fp8_e4m3fn.safetensors` (4.89 GB)
- `clip_l.safetensors` (246 MB)

- `ae.safetensors` (335 MB)

For 24GB GPUs, download:

- `flux1-dev-fp16.safetensors` (23.8 GB) or the FP8 all-in-one from Comfy-Org (17.2 GB)
- `t5xxl_fp16.safetensors` (9.79 GB)
- `clip_l.safetensors` (246 MB)
- `ae.safetensors` (335 MB)

Step 3: Install the GGUF Custom Node (If Using GGUF Models)

If you downloaded a GGUF model, you need the ComfyUI-GGUF node:

1. Open ComfyUI Manager (install it if you haven't)
2. Search for "ComfyUI-GGUF" by city96
3. Install and restart ComfyUI

This adds the `UNETLoader (GGUF)` node, which replaces the standard model loader.

Step 4: Build the Workflow

Flux uses separate loaders — **do not** use the "Load Checkpoint" node. Build this workflow:

1. **Load Diffusion Model** (or `UNETLoader (GGUF)` for GGUF files) → load your Flux model
2. **DualCLIPLoader** → load both `clip_l.safetensors` and your T5 encoder
3. **CLIP Text Encode** → your prompt (positive only — Flux ignores negative prompts)
4. **Empty Latent Image** → set to 1024x1024
5. **KSampler** → connect model, positive conditioning, and latent
6. **VAE Loader** → load `ae.safetensors`
7. **VAE Decode** → connect the sampled latent and VAE
8. **Save Image** → output

Step 5: Set the Right Parameters

This is where people get tripped up. Flux uses different settings than Stable Diffusion:

Setting	Flux.1 Dev	Flux.1 Schnell	Common Mistake
CFG Scale	3.0-3.8 (or 1.0 for FP8 checkpoints)	1.0	Using 7.5 like SD → overexposed images

Setting	Flux.1 Dev	Flux.1 Schnell	Common Mistake
Steps	20-30	1-4	Too many Schnell steps wastes time
Sampler	euler	euler	Most samplers work, euler is reliable
Scheduler	simple	simple	—
Negative prompt	Leave empty	Leave empty	Flux ignores negative prompts
Resolution	1024x1024	1024x1024	Non-square works but 1024x1024 is the training resolution

The biggest gotcha: **CFG scale**. Flux uses embedded guidance, not classical classifier-free guidance. A CFG of 3.5 in Flux is like 7.5 in Stable Diffusion. Setting it to 7+ produces blown-out, oversaturated images.

Method 2: Forge (SD WebUI Forge)

If you're already using AUTOMATIC1111's interface and prefer that layout, Forge is a fork that adds native Flux support with better memory management.

Setup

1. Clone Forge from GitHub: `https://github.com/lllyasviel/stable-diffusion-webui-forge`
2. Run the installer script for your OS
3. Place Flux models in the appropriate directories (same files as ComfyUI)
4. Select the Flux model from the UI dropdown

Forge Advantages

- Familiar A1111-style interface
- Built-in NF4 and FP8 support — no custom nodes needed
- GPU weight slider lets you split the model between GPU and CPU for low-VRAM cards
- "Shared" memory offload is ~15% faster than CPU swap

Forge Limitations

- Fewer Flux-specific workflows than ComfyUI
- Slower to add support for new features
- Node-based workflows aren't available

Verdict: Use Forge if you already know the A1111 interface and want something familiar. Use ComfyUI if you're starting fresh — it has better Flux support, more workflows, and a larger community building Flux-specific tools.

Method 3: Fooocus — Not Supported

Fooocus does **not** support Flux. If you're looking for a simple one-click UI, your options are:

- **ComfyUI Desktop** — the official desktop app has gotten much simpler to install
- **RuinedFooocus** — a community fork that adds Flux support, but it's not officially maintained
- **SwarmUI** — can run Flux on GPUs as low as 4GB with automatic optimization

For most people, ComfyUI Desktop is the answer.

Method 4: Python (diffusers)

If you want API-level control or are building Flux into an application, use HuggingFace's diffusers library.

Install

```
pip install diffusers>=0.36.0 transformers accelerate torch sentencepiece protobuf
```

Flux.1 Schnell (No Login Required)

```
import torch
from diffusers import FluxPipeline

pipe = FluxPipeline.from_pretrained(
```

```
    "black-forest-labs/FLUX.1-schnell",
    torch_dtype=torch.bfloat16,
)
pipe.enable_model_cpu_offload()

image = pipe(
    prompt="A cat holding a sign that says hello world",
    guidance_scale=0.0,
    num_inference_steps=4,
    max_sequence_length=256,
    height=1024,
    width=1024,
).images[0]

image.save("output.png")
```

Flux.1 Dev (Requires HuggingFace Login)

Flux.1 Dev is a gated model. You need to:

1. Accept the license at `huggingface.co/black-forest-labs/FLUX.1-dev`
2. Run `huggingface-cli login` with your token

```
import torch
from diffusers import FluxPipeline

pipe = FluxPipeline.from_pretrained(
    "black-forest-labs/FLUX.1-dev",
    torch_dtype=torch.bfloat16,
)
pipe.enable_model_cpu_offload()

image = pipe(
    prompt="a tiny astronaut hatching from an egg on the moon",
    guidance_scale=3.5,
    num_inference_steps=30,
    max_sequence_length=512,
    height=1024,
    width=1024,
).images[0]

image.save("output.png")
```

Low-VRAM Recipe (12GB GPU)

For GPUs with 12GB or less, quantize both the transformer and text encoder to NF4:

```
pip install bitsandbytes>=0.43.0
```

```
import torch
from diffusers import FluxPipeline, FluxTransformer2DModel
from diffusers import BitsAndBytesConfig as DiffusersBnBConfig
from transformers import T5EncoderModel
from transformers import BitsAndBytesConfig as TransformersBnBConfig

text_encoder_2 = T5EncoderModel.from_pretrained(
    "black-forest-labs/FLUX.1-dev",
    subfolder="text_encoder_2",
    quantization_config=TransformersBnBConfig(
        load_in_4bit=True,
        bnb_4bit_quant_type="nf4",
        bnb_4bit_compute_dtype=torch.float16,
    ),
    torch_dtype=torch.float16,
)

transformer = FluxTransformer2DModel.from_pretrained(
    "black-forest-labs/FLUX.1-dev",
    subfolder="transformer",
    quantization_config=DiffusersBnBConfig(
        load_in_4bit=True,
        bnb_4bit_quant_type="nf4",
        bnb_4bit_compute_dtype=torch.float16,
    ),
    torch_dtype=torch.float16,
)

pipe = FluxPipeline.from_pretrained(
    "black-forest-labs/FLUX.1-dev",
    text_encoder_2=text_encoder_2,
    transformer=transformer,
    torch_dtype=torch.float16,
)
pipe.enable_model_cpu_offload()
pipe.vae.enable_slicing()
pipe.vae.enable_tiling()
```

```
image = pipe(
    prompt="photograph of an astronaut riding a horse on mars, detailed, 8k",
    guidance_scale=3.5,
    num_inference_steps=30,
    height=1024,
    width=1024,
).images[0]

image.save("output.png")
```

Key Differences from Stable Diffusion in Code

- `guidance_scale` is embedded guidance, not classical CFG. Use 3.5 for Dev, 0.0 for Schnell.
- **No negative prompts** by default. If you need them, set `true_cfg_scale > 1.0`, but this doubles inference time.
- `max_sequence_length` is 512 for Dev, 256 for Schnell.
- **Use** `bfloat16` if your GPU is Ampere or newer (RTX 30xx+). Fall back to `float16` on older cards.

Optimization Tips

If generation is slow or you're hitting memory limits, try these in order of impact:

1. Quantize the T5 Text Encoder

The T5-XXL encoder alone is ~10GB at FP16. Switching to the FP8 version (4.89GB) saves ~5GB of VRAM with almost no visible quality loss. This is the single highest-impact optimization.

2. Use GGUF or NF4 Quantized Models

On 12GB GPUs, GGUF Q5_K_S gives you ~95% of FP16 quality while fitting comfortably. NF4 is even smaller but with more quality loss (~90-93%).

Speed comparison on 12GB cards:

- NF4 is 1.3-2.5x faster than FP8
- GGUF Q5 is slightly slower than NF4 but produces better images

3. Enable VAE Tiling and Slicing

Essential for generating images above 1024x1024. Without VAE tiling, high-resolution generation will OOM on most consumer GPUs.

In ComfyUI, this is handled automatically. In Python:

```
pipe.vae.enable_slicing()
pipe.vae.enable_tiling()
```

4. Use torch.compile (Advanced)

On RTX 4090, `torch.compile` can provide up to a 3.3x speedup (32 seconds down to ~10 seconds). The first generation takes 1-3 minutes for compilation, but subsequent runs are much faster.

```
pipe.transformer = torch.compile(pipe.transformer, mode="reduce-overhead")
```

Not worth the hassle on lower-end GPUs where the compilation overhead is proportionally larger.

5. Use Schnell for Iteration

When experimenting with prompts, switch to Schnell (1-4 steps) to get fast previews. Once you have a prompt you like, switch back to Dev for the final high-quality generation.

Flux vs Stable Diffusion: Which Should You Use?

Category	Flux	Stable Diffusion (SDXL / SD 3.5)
Text in images	Readable, correct spelling	Garbled or warped text
Human anatomy	Correct fingers and limbs	Still struggles with hands
Prompt adherence	Follows complex prompts accurately	Often misses elements or swaps attributes

Category	Flux	Stable Diffusion (SDXL / SD 3.5)
Photorealism	Comparable to Midjourney 6	Good but less consistent
Speed	12-80 sec depending on GPU/ quant	3-15 sec for SDXL
VRAM minimum	8 GB (quantized)	4 GB (SD 1.5), 8 GB (SDXL)
Negative prompts	Not supported	Full support
LoRA ecosystem	Growing, thousands on CivitAI	Massive, years of community content
Artistic styles	Good general coverage	Deep knowledge of specific artists, styles, manga
Inpainting	Available (Fill-dev model)	Mature and well-supported
ControlNet	Available (Canny, Depth, HED)	Comprehensive support for many control types
Model size	23.8 GB (FP16)	6.5 GB (SDXL)

Use Flux When

- You need readable text in images (signs, labels, documents)
- You're generating people and want correct anatomy
- You have complex prompts with multiple subjects and specific attributes
- You have 12GB+ VRAM

Stick with Stable Diffusion When

- You have 4-8GB VRAM and want fast generation
- You need a specific artistic style with a community fine-tune
- You rely heavily on negative prompts
- You need the deepest possible ControlNet and inpainting ecosystem
- Speed matters more than quality (SDXL generates in seconds, not minutes)

Many people use both. Flux for complex prompts and photorealism, SD for specific styles and fast iteration.

Common Problems and Fixes

Black Images

- **Wrong VAE.** Make sure you're using the Flux VAE (`ae.safetensors`), not an SDXL or SD 1.5 VAE.
- **Corrupted LoRA.** Disable any LoRAs and test with the base model.
- **NaN seed.** Try a fixed seed (like 42) instead of random.

Out of Memory (OOM)

- Use a smaller quantization (Q5 → Q4, or switch to NF4)
- Quantize the T5 encoder to FP8 or GGUF Q5
- Reduce resolution from 1024x1024 to 768x768
- Enable CPU offloading in your UI or script
- In Forge, lower the "GPU Weight" slider

Oversaturated / Blown-Out Images

- **CFG is too high.** Flux uses embedded guidance — set CFG to 1.0 for FP8 checkpoints, 3.0-3.8 for FP16 Dev. Never go above 6.

Slow Generation

- Check that you're actually using the GPU: in ComfyUI, watch your VRAM usage. If it stays at 0, the model is running on CPU.
- Use Schnell (4 steps) instead of Dev (20-30 steps) for quick tests.
- NF4 is faster than GGUF on GPUs with less than 16GB VRAM.
- Close other VRAM-heavy applications (browsers with hardware acceleration, games).

PyTorch Issues

- PyTorch 2.4.0 has a known bug that produces noisy/pixelated Flux images. Downgrade to 2.3.1 or upgrade to 2.5+.
- `bfloat16` requires an Ampere GPU (RTX 30xx or newer). Use `float16` on older cards.
- On macOS, FP8 is not supported. Use GGUF quantized models via ComfyUI-GGUF instead.

The Flux Ecosystem

Flux isn't just a single model anymore. The community has built:

- **LoRAs**: Thousands available on CivitAI — style LoRAs, character LoRAs, concept LoRAs. QLoRA training works on consumer hardware with ~10GB VRAM.
- **ControlNet**: Canny, Depth, and HED control nets from XLabs-AI. Not as comprehensive as SD's ecosystem yet, but the important ones are covered.
- **IP-Adapter**: Style and subject transfer from reference images. Available through XLabs-AI for ComfyUI.
- **Inpainting**: Flux.1 Fill-dev handles inpainting and outpainting.
- **Image editing**: Flux.1 Kontext enables text+image prompted editing — change elements in existing images.

The ecosystem is younger than Stable Diffusion's but growing fast. For most common workflows, the tools are already there.

Bottom Line

Flux is the best open image generation model right now. If you have 12GB+ VRAM, there's no reason not to try it.

8GB VRAM: Flux works with NF4/Q4 quantization but it's slow (60+ seconds). Stable Diffusion XL is still the better daily driver at this tier.

12GB VRAM: The practical minimum for a good Flux experience. Use GGUF Q5_K_S with an FP8 T5 encoder. Expect ~60-80 seconds per image — slower than SDXL, but the quality improvement is worth the wait. An RTX 3060 12GB is the budget entry point.

16GB VRAM: Comfortable. Use FP8 or GGUF Q8 for near-full quality at ~40-55 seconds per image. The upcoming RTX 5060 Ti 16GB will be a great Flux card.

24GB VRAM: No compromises. Full FP16, any resolution, and fast generation. A used RTX 3090 at ~$750 or an RTX 4090 at $1,599 gives you the full Flux experience.

Start with ComfyUI. Download a GGUF Q5 model if you have 12-16GB VRAM, or the full FP16 model if you have 24GB. Generate your first image. Then explore LoRAs, ControlNets, and Schnell for fast iteration.

Get notified when we publish new guides.

Subscribe — free, no spam

Source: https://insiderllm.com/guides/flux-locally-complete-guide/