



Howto/

# NVIDIA

## Contents

1. [About this Howto](#)
2. [Determining your card model](#)
3. [Installing the drivers](#)
  1. [Current GeForce/Quadro/Tesla](#)
  2. [Legacy GeForce 600/700](#)
  3. [Legacy GeForce 400/500](#)
  4. [Legacy GeForce 8/9/200/300](#)
  5. [Legacy GeForce 6/7](#)
  6. [Legacy GeForce 5 \(FX series\)](#)
  7. [Legacy GeForce 2 through GeForce 4](#)
4. [Special notes](#)
  1. [Optimus](#)
  2. [OSTree \(Silverblue/Kinoite/etc\)](#)
  3. [Switching between nouveau/nvidia](#)
  4. [CUDA](#)
  5. [Version Lock](#)
  6. [KMS](#)
  7. [Graphic console](#)
  8. [Suspend](#)
  9. [Vulkan](#)
  10. [Vulkan Beta](#)
  11. [Wayland](#)
  12. [NVENC/NVDEC](#)
  13. [Kernel Open](#)
  14. [Latest/Beta driver](#)
  15. [nvidia-xconfig](#)
  16. [x86\\_64 \(64bit\) users](#)
  17. [PAE \(Physical Address Extension\) kernel users](#)
  18. [VDPAU/VAAPI](#)
  19. [Akmods](#)
  20. [Nouveau compatibility](#)
  21. [Rawhide](#)
5. [Uninstall the NVIDIA driver](#)
6. [Recover from NVIDIA installer](#)
7. [Bug Report](#)

### Login

User Password

## About this Howto

---

This howto will help you install the correct NVIDIA driver on Fedora for your graphics card as well as troubleshoot common driver problems.

The prerequisite is to have followed the [Configuration](#) page to have at least the RPM Fusion nonfree section available. -

## Determining your card model

---

NVIDIA has several driver series, each of which has different hardware support. To determine which driver you need to install, you'll first need to find your graphics card model.

If you don't know it, open a Terminal (Applications > System Tools > Terminal) and type:

```
/sbin/lspci | grep -e VGA
```

You can also check the [supported chips](#) section and see which series is recommended for your card, then install the appropriate driver series. Please remember that you need additional steps for optimus.


You are probably in the [Optimus](#) case if your NVIDIA card is found with the next command:

```
/sbin/lspci | grep -e 3D
```

## Installing the drivers

---

Please remember that once the driver is installed, there is no need to configure xorg.conf by default. Changes will take effect after a **full reboot** on the newest kernel.


 The **Secure Boot** Please have a look on [Howto/Secure Boot](#) in order to sign the nvidia kmod. You will have to enter the BIOS/EFI to import your self generated key.

### Current GeForce/Quadro/Tesla


---

Supported on current stable Xorg server release.

This driver is suitable for any GPU found in 2014 and later.

 The 510+ driver is available by default on Fedora 34+ and later and has dropped support for some older Kepler GPU.

```
sudo dnf update -y # and reboot if you are not on the latest kernel
sudo dnf install akmod-nvidia # rhel/centos users can use kmod-
nvidia instead
sudo dnf install xorg-x11-drv-nvidia-cuda #optional for
cuda/nvdec/nvenc support
```

 Please remember to wait after the RPM transaction ends, until the kmod get built. This can take up to 5 minutes on some systems.

Once the module is built, "modinfo -F version nvidia" should outputs the version of the driver

such as 440.64 and not modinfo: ERROR: Module nvidia not found.

## Legacy GeForce 600/700

---

Supported on current stable Xorg server release.

⚠ This serie is introduced since Fedora 34+.

This driver is suitable for any NVIDIA Kepler GPU found between 2012 and 2014

```
dnf update -y
sudo dnf install xorg-x11-drv-nvidia-470xx akmod-nvidia-470xx
sudo dnf install xorg-x11-drv-nvidia-470xx-cuda #optional for cuda
up to 11.4 support
```

---

⚠ Please remember to wait after the RPM transaction ends, until the kmod get built. This can take up to 5 minutes on some systems.

## Legacy GeForce 400/500

---

Supported on current stable Xorg server release. EOL by NVIDIA at the end of 2022. Still available on "best effort basis" (newer kernel may break, will be discontinued at anytime if not actively maintained)

This driver is suitable for any NVIDIA Fermi GPU found between 2010 and 2012

```
dnf update -y
sudo dnf install xorg-x11-drv-nvidia-390xx akmod-nvidia-390xx
sudo dnf install xorg-x11-drv-nvidia-390xx-cuda #optional for cuda
up to 9.2 support
```

---

⚠ Please remember to wait after the RPM transaction ends, until the kmod get built. This can take up to 5 minutes on some systems.

## Legacy GeForce 8/9/200/300

---

Supported on current stable Xorg server release. EOL by NVIDIA at the end of 2019. Still available on "best effort basis" (newer kernel may break, will be discontinued at anytime if not actively maintained)

```
sudo dnf update -y
sudo dnf install xorg-x11-drv-nvidia-340xx akmod-nvidia-340xx
sudo dnf install xorg-x11-drv-nvidia-340xx-cuda #optional for cuda
up to 6.5 support
```

---

⚠ Please remember to wait until the kmod get built each time a new kernel rise up. This can take up to 5 minutes on some systems.

⚠ Workaround for keeping 340xx driver on newer fedora.

⚠ This driver doesn't support "pre-optimus" devices so if you have a mean to only use the NVIDIA device in bios, please turn it on (or rely on Intel/Nouveau if you want to keep both).

```
sudo dnf copr enable kwizart/kernel-longterm-6.1
sudo dnf install akmods gcc kernel-longterm kernel-longterm-devel
sudo dnf install xorg-x11-drv-nvidia-340xx akmod-nvidia-340xx
```

---

## Legacy GeForce 6/7

---

Supported up to Fedora 27 - EOL, no more nvidia updates

## Legacy GeForce 5 (FX series)

---

Supported up to Fedora 20 - EOL, no more nvidia updates

## Legacy GeForce 2 through GeForce 4

---

Supported up to Fedora 14 - EOL, no more nvidia updates

## Special notes

---

### Optimus

---

NVIDIA Optimus is a technology that allows an Intel integrated GPU and discrete NVIDIA GPU to be built into and accessed by a laptop.

With Fedora 25 and later, Optimus devices are supported automatically by default. Please see the dedicated [Optimus Howto](#).

### OSTree (Silverblue/Kinoite/etc)

---

OSTree based systems are supported with the NVIDIA driver using akmod-nvidia starting with f30. (only the current NVIDIA driver is tested). Please follow the [Configuration](#) page first.

```
sudo rpm-ostree install akmod-nvidia xorg-x11-drv-nvidia
sudo rpm-ostree install akmod-nvidia xorg-x11-drv-nvidia-cuda
#optional if using nvidia-smi or cuda
sudo rpm-ostree kargs --append=rd.driver.blacklist=nouveau --
append=modprobe.blacklist=nouveau --append=nvidia-drm.modeset=1
initcall_blacklist=simpledrm_platform_driver_init
# If any issue arise, please verify this documentation for updates
related to kernel arguments since we cannot update them on ostree
systems from the RPM package. This is an ostree limitation (they
call it: feature).
```

Note: There is a need for a special "Hack" to work with secureboot on OSTree systems. Once you have the keys generated and imported, you need to create a dedicated packages for the key to be available when signing the kernel modules in OSTree. See also <https://github.com/CheariX/silverblue-akmods-keys>

### Switching between nouveau/nvidia

---

With recent drivers as packaged with RPM Fusion, it is possible to switch easily between nouveau and nvidia while keeping the nvidia driver installed. When you are about to select the kernel at the grub menu step. You can edit the kernel entry, find the linux boot command line and manually remove the following options "rd.driver.blacklist=nouveau modprobe.blacklist=nouveau nvidia-drm.modeset=1". This will allow you to boot using the nouveau driver instead of the nvidia binary driver. At this time, there is no way to make the switch at runtime.

### CUDA

---

The driver support CUDA when installing the xorg-x11-drv-nvidia-cuda subpackage. Please

have a look on the dedicated [CUDA Howto](#)

```
sudo dnf install xorg-x11-drv-nvidia-cuda
```

## Version Lock

Sometime, there is a need to lock to a particular driver version for any reason (regression, compatibility with another application, vulka beta branch or else). Using dnf versionlock module is the appropriate way to deal with that. Please remember that version lock will prevent any updates to the nvidia driver including fixes for kernel compatibilities if relevant.

```
dnf install python3-dnf-plugin-versionlock
rpm -qa xorg-x11-drv-nvidia* *kmod-nvidia* nvidia-
{settings,xconfig,modprobe,persistenced} >>
/etc/dnf/plugins/versionlock.list
```

## KMS

KMS stands for "Kernel Mode Setting" which is the opposite of "Userland Mode Setting". This feature allows to set the screen resolution on the kernel side once (at boot), instead of after login from the display manager. This feature has early support in the main NVIDIA driver, but is not enabled by default. Our package enable this feature by default to ease installation with optimus and wayland which requires this.

To disable, use:

```
sudo grubby --update-kernel=ALL --remove-args='nvidia-drm.modeset=1'
```

To re-enable, use

```
sudo grubby --update-kernel=ALL --args='nvidia-drm.modeset=1'
```

## Graphic console

For long time, NVIDIA driver team have advertised "against" using a graphical console (vga parameters or else) over the default text console on non-EFI mode (bios). That's because some drivers bad interaction with the vesafb driver. This problem is now solved as it seems. (See also <https://forums.developer.nvidia.com/t/unusable-linux-text-console-with-nvidia-drm-modeset-1-or-if-nvidia-persistenced-is-loaded>).

If using a legacy bios, you can improve the vesafb driver using this parameter: (along the appropriate vga=id, use vga=ask for the first time).

```
sudo grubby --update-kernel=ALL --args='video=vesafb:mtrr:3'
```

## Suspend

NVIDIA has provided some experimental scripts to enable clean resume on suspend to RAM or suspend to disk (hibernate) that might be specially needed in some environment. Theses scripts will prevent artefacts on resume by moving the full content of the graphic memory into either host RAM (or disk). Packaging is currently (as of 2021-08-23) in in-between state and will be improved as discussed in [https://bugzilla.rpmsfusion.org/show\\_bug.cgi?id=6066](https://bugzilla.rpmsfusion.org/show_bug.cgi?id=6066)

```
sudo dnf install xorg-x11-drv-nvidia-power
```

```
sudo systemctl enable nvidia-{suspend,resume,hibernate}  
# Optional: tweak "nvidia options NVreg_TemporaryFilePath=/var/tmp"  
from /etc/modprobe.d/nvidia.conf as needed if you have issue with  
/tmp as tmpfs with nvidia suspend )
```

---

## Vulkan

---

The main package support Vulkan, but you need to install the vulkan libraries if requested.

```
sudo dnf install vulkan
```

---

## Vulkan Beta

---

A dedicated Vulkan Beta driver version is maintained since October 2022 for fedora 36 and x86\_64 only. <https://developer.nvidia.com/vulkan-driver>

Because it's not a dedicated package, you need to use 'versionlock' to prevent the packages to be updated to the main series.

```
sudo dnf install rpmfusion-nonfree-release-tainted  
sudo dnf distro-sync *nvidia*
```

---

## Wayland

---

NVIDIA works under Wayland (and Xwayland) starting with Fedora 35 and NVIDIA driver 495 and later. With GNOME 41, Wayland can be selected explicitly with GDM.

Please remind that video acceleration with VDPAU isn't available under Wayland.

## NVENC/NVDEC

---

RPM Fusion support ffmpeg compiled with NVENC/NVDEC with Fedora 25 and later. You need to have a recent NVIDIA card (see the [support matrix](#)), and install the cuda sub-package.

```
sudo dnf install xorg-x11-drv-nvidia-cuda-libs
```

---

Please have a look on the ffmpeg [HWAaccel introduction](#) to the feature

## Kernel Open

---

Since the NVIDIA driver 515xx, there is a mean to use a full open source kernel space driver (userspace driver remains proprietary). This is only relevant for Turing, Ampere and later (there is no chance for it to works with older GPU).

With 515xx, the driver is only production ready for datacenter. (no workstation, display GPU as it misses certain features), but you can add an option for using it on such GPU if it's recent enough family. (see the nvidia doc).

For akmod-nvidia to build with the FLOSS (but pre-built) driver by default, you can use:

```
sudo sh -c 'echo "%global _with_kmod_nvidia_open 1" >  
/etc/rpm/macros-nvidia-kmod'  
sudo akmods --force
```

---

You can also swith to the akmod-nvidia-open package that relies on the full source code of

the kernel space driver (it takes longer to build). This package is located in the rpmfusion-nonfree-tainted repository so it doesn't get selected by mistake from unaware users.

```
sudo dnf install rpmfusion-nonfree-release-tainted
sudo dnf swap akmod-nvidia akmod-nvidia-open
```

See also [http://download.nvidia.com/XFree86/Linux-x86\\_64/515.43.04/README/kernel\\_open.html](http://download.nvidia.com/XFree86/Linux-x86_64/515.43.04/README/kernel_open.html)

## Latest/Beta driver

You can install the latest drivers from Rawhide using the following command:

```
sudo dnf install "kernel-devel-uname-r >= $(uname -r)"
sudo dnf update -y
sudo dnf copr enable kwizart/nvidia-driver-rawhide -y
sudo dnf install rpmfusion-nonfree-release-rawhide -y
sudo dnf --enablerepo=rpmfusion-nonfree-rawhide install akmod-nvidia
xorg-x11-drv-nvidia xorg-x11-drv-nvidia-cuda --nogpgcheck
```

Or if you want to grab it from the latest fedora stable release:

```
sudo dnf install "kernel-devel-uname-r == $(uname -r)"
sudo dnf update -y
sudo dnf --releasever=30 install akmod-nvidia xorg-x11-drv-nvidia --nogpgcheck
```

## nvidia-xconfig

This tool is only meant to be used as a sample to create a xorg.conf files. But don't use this directly as the generated xorg.conf is known to broke with many default Fedora/RHEL Xorg server options. Instead, you should probably start with :

```
sudo cp /usr/share/X11/xorg.conf.d/nvidia.conf
/etc/X11/xorg.conf.d/nvidia.conf
```

## x86\_64 (64bit) users

If you wish to have 3D acceleration in 32bit packages such as Wine, be sure to install the appropriate 32bit version of the xorg-x11-drv-nvidia-libs package for your driver variant. For example, if you installed kmod-nvidia then you will require xorg-x11-drv-nvidia-libs.i686. With Current Fedora (not EL), this is handled automatically by RPM (Boolean dependencies).

## PAE (Physical Address Extension) kernel users

If you are on a 32bit (i686) system and have the kernel-PAE installed to access more RAM, please install kernel-PAE-devel. Please note that this step is not required for any 64bit (x64\_64) users.

## VDPAU/VA-API

In order to enable video acceleration support for your player and if your NVIDIA card is recent enough (Geforce 8 and later is needed). You can install theses packages:

```
# sudo dnf install nvidia-vaapi-driver libva-utils vdpauinfo
```

With the native vdpau backend from a NVIDIA card, the output is similar to this:

```
$ vdpauinfo
display: :1    screen: 0
API version: 1
Information string: NVIDIA VDPAU Driver Shared Library 530.41.03
Thu Mar 16 19:21:47 UTC 2023
...
```

Here is an example of an accurate output of vainfo, when the bridge to the VAAPI is correctly installed.

```
$ vainfo
Trying display: wayland
Trying display: x11
libva info: VA-API version 1.16.0
libva info: Trying to open /usr/lib64/dri/nvidia_drv_video.so
libva info: Found init function __vaDriverInit_1_0
libva info: va_openDriver() returns 0
vainfo: VA-API version: 1.16 (libva 2.16.0)
vainfo: Driver version: VA-API NVDEC driver [egl backend]
vainfo: Supported profile and entrypoints
...
```

## Akmods

An [akmod](#) is a type of package similar to dkms. As you start your computer, the akmod system will check if there are any missing kmodes and if so, rebuild a new kmod for you. Akmods have more overhead than regular kmod packages as they require a few development tools such as gcc and automake in order to be able to build new kmodes locally.

## Nouveau compatibility

As nouveau is enabled by default starting with Fedora 11, you may experience problem which is caused by the nouveau kernel module being present in the initrd image. Once the driver is installed and after the reboot, this command should not output anything:

```
lsmod |grep nouveau
```

## Rawhide

Rawhide kernels are built with debug enabled GPL-only symbols which kernel is incompatible with the NVIDIA binary-only driver. You need to use the [Rawhide nodebug repository](#).

```
sudo dnf config-manager --add-
repo=http://dl.fedoraproject.org/pub/alt/rawhide-kernel-
nodebug/fedora-rawhide-kernel-nodebug.repo
sudo dnf update
```

Please remind that Xorg server version is also to take into consideration. In the case of incompatibilities with the xorg-server, you might need to downgrade to the previous fedora release:

```
sudo dnf downgrade xorg-x11-server\* --releasever=29 --allowerase
sudo echo "exclude=xorg-x11*" >> /etc/dnf/dnf.conf
```



## Uninstall the NVIDIA driver

---

```
dnf remove xorg-x11-drv-nvidia\*
```

---

## Recover from NVIDIA installer

---

The NVIDIA binary driver installer overwrite some configuration and libraries. If you want to recover to a clean state, either to use nouveau or the packaged driver, use:

```
rm -f /usr/lib{,64}/libGL.so.* /usr/lib{,64}/libEGL.so.*
rm -f /usr/lib{,64}/xorg/modules/extensions/libglx.so
dnf reinstall xorg-x11-server-Xorg mesa-libGL mesa-libEGL libglvnd\*
mv /etc/X11/xorg.conf /etc/X11/xorg.conf.saved
```

---

## Bug Report

---

If you still cannot make the driver to work, you can either report a problem to NVIDIA or to RPM Fusion packager team. Please read: [If you have a problem, PLEASE read this first](#)

Basically, you need to generate an archive and attach it to bugzilla or any paste service.

```
sudo nvidia-bug-report.sh
zcat nvidia-bug-report.log.gz | fpaste
```

---

## CategoryHowto

---

Howto/NVIDIA (last edited 2023-04-24 13:44:26 by [NicolasChauvet](#))