



[Translation\(s\)](#): [English](#) - [Español](#) - [Français](#) - [Русский](#) - [简体中文](#)

NVIDIA Proprietary Driver

This page describes how to install the NVIDIA proprietary display driver on Debian systems.

Commands in this article prefixed with a # indicate they must be run as root. Replace this character with sudo or switch user to root in your terminal beforehand as necessary.

NOTE: For Apple systems, follow these steps first to prevent a black screen after installing the drivers: <https://askubuntu.com/a/613573/134848>

Tabla de Contenidos

1. [NVIDIA Proprietary Driver](#)
 1. [Identification](#)
 1. [nvidia-detect](#)
 2. [Desktop Drivers](#)
 1. [Prerequisites](#)
 1. [Kernel headers](#)
 2. [Kernel](#)
 2. [Installation](#)
 1. [Debian Unstable "Sid"](#)
 1. [Version 525.105.17](#)
 2. [Version 390.144](#)
 3. [Version 340.108](#)
 2. [Debian 12 "Bookworm"](#)
 1. [Version 525.105.17](#)
 3. [Debian 11 "Bullseye"](#)
 1. [Version 470.129.06](#)
 2. [Version 390.144](#)
 4. [Debian 10 "Buster"](#)
 1. [Version 460.73.01 \(via buster-backports\)](#)
 2. [Version 418.197.02](#)
 3. [Version 390.138 \(legacy GPUs\)](#)
 4. [Version 340.108 \(legacy GPUs\)](#)
 5. [Debian 9 "Stretch"](#)
 1. [Version 418.152 \(via stretch-backports\)](#)
 2. [Version 390.138](#)
 3. [Version 340.108 \(legacy GPUs\)](#)
 4. [Version 304.137 \(legacy GPUs\)](#)
 3. [Installing 32-bit libraries on a 64-bit system](#)
 3. [Wayland](#)
 4. [Tesla Drivers](#)
 5. [Configuration](#)
 1. [Automatic](#)
 2. [Manual](#)
 6. [CUDA](#)
 1. [Debian Unstable "Sid"](#)
 2. [Debian 12 "Bookworm"](#)
 3. [Debian 11 "Bullseye"](#)
 4. [Debian 10 "Buster"](#)
 5. [Debian 9 "Stretch"](#)
 7. [OptiX](#)
 8. [Troubleshooting](#)
 1. [Build failures](#)
 2. [Driver stops working after upgrading Debian](#)
 3. [GPU isn't functional, even with a compatible driver version installed](#)
 4. [Miscellaneous](#)

9. [Uninstallation](#)


10. [See Also](#)

Identification


The NVIDIA graphics processing unit (GPU) series/codename of an installed video card can usually be identified using the `lspci` command. For example:

```
$ lspci -nn | egrep -i "3d|display|vga"
07:00.0 VGA compatible controller [0300]: NVIDIA Corporation GM206 [GeForce GTX 960] [10de:1401]
```

See [HowToIdentifyADevice/PCI](#) for more information. The PCI ID can be used to verify device support.

Note: if this `lspci` command returns more than one line of output, you have an  [Optimus](#) (hybrid) graphics chipset. After you install the necessary driver package, you'll still need to choose one of the methods on the [NVIDIA Optimus](#) page in order to activate and make use of your NVIDIA card.

nvidia-detect

The `nvidia-detect` script (found in the [DebianPkg: nvidia-detect](#) package in the  [non-free](#) section) can also be used to identify the GPU and the recommended driver package to install:




```
$ nvidia-detect
Detected NVIDIA GPUs:
07:00.0 VGA compatible controller [0300]: NVIDIA Corporation GM206 [GeForce GTX 960] [10de:1401]

Checking card:  NVIDIA Corporation GM206 [GeForce GTX 960] (rev a1)
Your card is supported by all driver versions.
Your card is also supported by the Tesla 440 drivers series.
Your card is also supported by the Tesla 418 drivers series.
It is recommended to install the
    nvidia-driver
package.
```


Desktop Drivers

The proprietary "NVIDIA Accelerated Linux Graphics Driver" provides optimized hardware acceleration of OpenGL and Vulkan applications through either Xorg or Wayland. It is a binary-only driver requiring a Linux kernel module for its use.



Multiple precompiled driver versions are available for [Debian Unstable "Sid"](#):

- [Version 525.105.17](#) ( [supported devices](#))
 - Supports Kepler, Maxwell, Pascal, Turing, and all current Ampere GPUs. Supports Vulkan 1.2 and OpenGL 4.6.
- [Version 390.144](#) ( [supported devices](#))
 - Supports Fermi, Kepler, Maxwell, and most Pascal GPUs. Supports Vulkan 1.0 on Kepler and newer, supports up to OpenGL 4.5 depending on your card.
- [Version 340.108 \(legacy GPUs\)](#) ( [supported devices](#))
 - Older legacy driver, for GeForce 8 series through GeForce 300 series. No Vulkan support, supports up to OpenGL 3.3 depending on your card.
 - **Use of the 340-series driver is strongly discouraged.** It is not included in stable releases of Debian anymore, has serious unfixable security vulnerabilities, and may not be updated for new kernels in a timely manner. You are highly recommended to use the built-in Nouveau driver if security is a priority.





Only one precompiled driver version is available for [Debian 12 "Bookworm"](#):

- [Version 525.105.17](#) ( [supported devices](#))
 - Supports Kepler, Maxwell, Pascal, Turing, and all current Ampere GPUs. Supports Vulkan 1.2 and OpenGL 4.6.





Multiple precompiled driver versions are available for [Debian 11 "Bullseye"](#):

- [Version 470.129.06](#) ( [supported devices](#))
 - Supports Kepler, Maxwell, Pascal, Turing, and all current Ampere GPUs. Supports Vulkan 1.2 and OpenGL 4.6.
- [Version 390.144](#) ( [supported devices](#))
 - Supports Fermi, Kepler, Maxwell, and most Pascal GPUs. Supports Vulkan 1.0 on Kepler and newer, supports up to OpenGL 4.5 depending on your card.

Multiple precompiled driver versions are available for [Debian 10 "Buster"](#):

- [Version 460.73.01](#) ( [supported devices](#))
 - Supports Kepler, Maxwell, Pascal, Turing, and all current Ampere GPUs. Supports Vulkan 1.2 and OpenGL 4.6.
 - Note that 460.73.01 is only available in buster-backports.
- [Version 418.197.02](#) ( [supported devices](#))
 - Supports Kepler, Maxwell, Pascal, and most Turing GPUs. Supports Vulkan 1.1 and OpenGL 4.6.
- [Version 390.143](#) ( [supported devices](#))
 - Supports Fermi, Kepler, Maxwell, and most Pascal GPUs. Supports Vulkan 1.0 on Kepler and newer, supports up to OpenGL 4.5 depending on your card.
- [Version 340.108 \(legacy GPUs\)](#) ( [supported devices](#))
 - Older legacy driver, for GeForce 8 series through GeForce 300 series. No Vulkan support, supports up to OpenGL 3.3 depending on your card.

Multiple precompiled driver versions are available for [Debian 9 "Stretch"](#):

- [Version 418.152](#) ( [supported devices](#))
 - Supports Kepler, Maxwell, Pascal, and most Turing GPUs. Supports Vulkan 1.1 and OpenGL 4.6.
 - Note that 418.152 is only available in stretch-backports.
- [Version 390.138](#) ( [supported devices](#))
 - Supports Fermi, Kepler, Maxwell, and most Pascal GPUs. Supports Vulkan 1.0 on Kepler and newer, supports up to OpenGL 4.5 depending on your card.
- [Version 340.108 \(legacy GPUs\)](#) ( [supported devices](#))
 - Older legacy driver, for GeForce 8 series through GeForce 300 series. No Vulkan support, supports up to OpenGL 3.3 depending on your card.
- [Version 304.137 \(legacy GPUs\)](#) ( [supported devices](#))
 - Even older legacy driver, for GeForce 6 series and GeForce 7 series. Only supports OpenGL 2.1.

All driver versions up to, and including, the 418-series, are only available for the x86, x86-64, and 32-bit ARMv7 architectures (Debian [i386](#), [AMD64](#), and [ARMHF](#) ports respectively).

The 450-series and newer has dropped support for 32-bit architectures, now only supporting x86-64 and ARMv8 (Debian [AMD64](#) and [ARM64](#) ports respectively).

Prerequisites

Kernel headers

Before installing the drivers, you **must** obtain the proper kernel headers for the NVIDIA driver to build with.

For a typical 64-bit system using the default kernel, you can simply run:

```
# apt install linux-headers-amd64
```

For 32-bit systems with the non-PAE kernel, you'd instead install:

```
# apt install linux-headers-686
```

Or, for 32-bit systems with the PAE kernel:

```
# apt install linux-headers-686-pae
```

If you're using the kernel from [Debian Backports](#), you must run the same command but with the `-t` flag followed by the name of your backports source. For instance, if you're using backports on a 64-bit Debian 10 system, you might run:

```
# apt install -t buster-backports linux-headers-amd64
```

Kernel

In some cases, if you're aiming to install the bleeding-edge version of the NVIDIA driver from Debian Backports, you may also need to install the kernel from backports to match it. For Debian 10, you might do this with:

```
# apt install -t buster-backports linux-image-amd64
```

Exchange "buster-backports" with your own version's backports repository as necessary.

Installation

Debian Unstable "Sid"

Version 525.105.17

For support of GeForce 600 series and newer GPUs ([supported devices](#)). For older devices, see [Version 390 \(legacy GPUs\)](#).

1. Add "contrib", "non-free" and "non-free-firmware" components to `/etc/apt/sources.list`, for example:

```
# Debian Sid
deb http://deb.debian.org/debian/ sid main contrib non-free non-free-firmware
```

2. Update the list of available packages, then we can install the [DebianPkg: nvidia-driver](#) package, plus the necessary firmware:

```
# apt update
# apt install nvidia-driver firmware-misc-nonfree
```

DKMS will build the nvidia module for your system, via the [DebianPkg: nvidia-kernel-dkms](#) package.

- *Note about Secureboot:* if you have [SecureBoot](#) enabled, you need to sign the resulting modules. Detailed instructions are available [here](#).

3. Restart your system to load the new driver.

Version 390.144

For support of GeForce 400 series and newer GPUs ([supported devices](#)).

1. Add "contrib", "non-free" and "non-free-firmware" components to `/etc/apt/sources.list`, for example:

```
# Debian Sid
deb http://deb.debian.org/debian/ sid main contrib non-free non-free-firmware
```

2. Update the list of available packages, then we can install the [DebianPkg: nvidia-legacy-390xx-driver](#) package, plus the necessary firmware:

```
# apt update
# apt install nvidia-legacy-390xx-driver firmware-misc-nonfree
```

DKMS will build the nvidia module for your system, via the [DebianPkg: nvidia-legacy-390xx-kernel-dkms](#) package.

- *Note about Secureboot* : if you have [SecureBoot](#) enabled, you need to sign the resulting modules. Detailed instructions are available [here](#).

3. Restart your system to load the new driver.

Version 340.108

For support of GeForce 8 series through GeForce 300 series GPUs([supported devices](#)).

Use of the 340-series driver is strongly discouraged. It is not included in stable releases of Debian anymore, has serious unfixable security vulnerabilities, and may not be updated for new kernels in a timely manner. You are highly recommended to use the built-in Nouveau driver if security is a priority.

1. Add "contrib", "non-free" and "non-free-firmware" components to `/etc/apt/sources.list`, for example:

```
# Debian Sid
deb http://deb.debian.org/debian/ sid main contrib non-free non-free-firmware
```

2. Update the list of available packages, then we can install the [DebianPkg: nvidia-legacy-340xx-driver](#) package, plus the necessary firmware:

```
# apt update
# apt install nvidia-legacy-340xx-driver firmware-misc-nonfree
```

DKMS will build the nvidia module for your system, via the [DebianPkg: nvidia-legacy-340xx-kernel-dkms](#) package.

- *Note about Secureboot* : if you have [SecureBoot](#) enabled, you need to sign the resulting modules. Detailed instructions are available [here](#).

3. Restart your system to load the new driver.

Debian 12 "Bookworm"**Version 525.105.17**

For support of GeForce 600 series and newer GPUs ([supported devices](#)). For older devices, you must use [nouveau](#), which should be already installed and in use.

1. Add "contrib", "non-free" and "non-free-firmware" components to `/etc/apt/sources.list`, for example:

```
# Debian Bookworm
deb http://deb.debian.org/debian/ bookworm main contrib non-free non-free-firmware
```

2. Update the list of available packages, then we can install the [DebianPkg: nvidia-driver](#) package, plus the necessary firmware:

```
# apt update
# apt install nvidia-driver firmware-misc-nonfree
```

DKMS will build the nvidia module for your system, via the [DebianPkg: nvidia-kernel-dkms](#) package.

- *Note about Secureboot* : if you have [SecureBoot](#) enabled, you need to sign the resulting modules. Detailed instructions are available [here](#).

3. Restart your system to load the new driver.

Debian 11 "Bullseye"

Version 470.129.06

For support of GeForce 600 series and newer GPUs ([🌐 supported devices](#)). For older devices, see [Version 390 \(legacy GPUs\)](#).

1. Add "contrib" and "non-free" components to `/etc/apt/sources.list`, for example:

```
# Debian 11 "Bullseye"
deb http://deb.debian.org/debian/ bullseye main contrib non-free
```

2. Update the list of available packages, then we can install the [DebianPkg: nvidia-driver](#) package, plus the necessary firmware:

```
# apt update
# apt install nvidia-driver firmware-misc-nonfree
```

DKMS will build the nvidia module for your system, via the [DebianPkg: nvidia-kernel-dkms](#) package.

- *Note about Secureboot* : if you have [SecureBoot](#) enabled, you need to sign the resulting modules. Detailed instructions are available [here](#).

3. Restart your system to load the new driver.

Version 390.144

For support of GeForce 400 series and newer GPUs ([🌐 supported devices](#)).

1. Add "contrib" and "non-free" components to `/etc/apt/sources.list`, for example:

```
# Debian 11 "Bullseye"
deb http://deb.debian.org/debian/ bullseye main contrib non-free
```

2. Update the list of available packages, then we can install the [DebianPkg: nvidia-legacy-390xx-driver](#) package, plus the necessary firmware:

```
# apt update
# apt install nvidia-legacy-390xx-driver firmware-misc-nonfree
```


DKMS will build the nvidia module for your system, via the [DebianPkg: nvidia-legacy-390xx-kernel-dkms](#) package.

- *Note about Secureboot* : if you have [SecureBoot](#) enabled, you need to sign the resulting modules. Detailed instructions are available [here](#).

3. Restart your system to load the new driver.

Debian 10 "Buster"

Version 460.73.01 (via buster-backports)

For support of GeForce 600 series and newer GPUs ([supported devices](#)). For older devices, see [Version 390 \(legacy GPUs\)](#) and [Version 340 \(legacy GPUs\)](#).

1. Add buster-backports as an additional new line to your `/etc/apt/sources.list`, for example:

```
# buster-backports
deb http://deb.debian.org/debian buster-backports main contrib non-free
```

2. Update the list of available packages, then we can install the [DebianPkg: nvidia-driver](#) package, plus the necessary firmware, from the backports repository:

```
# apt update
# apt install -t buster-backports nvidia-driver firmware-misc-nonfree
```

DKMS will build the nvidia module for your system, via the [DebianPkg: nvidia-kernel-dkms](#) package.

- *Note about Secureboot* : if you have [SecureBoot](#) enabled, you need to sign the resulting modules. Detailed instructions are available [here](#).

3. Reboot your system to load the updated driver.

Version 418.197.02

For support of GeForce 600 series and newer GPUs ([supported devices](#)). For older devices, see [Version 390 \(legacy GPUs\)](#) and [Version 340 \(legacy GPUs\)](#).

1. Add "contrib" and "non-free" components to `/etc/apt/sources.list`, for example:

```
# Debian 10 "Buster"
deb http://deb.debian.org/debian/ buster main contrib non-free
```

2. Update the list of available packages, then we can install the [DebianPkg: nvidia-driver](#) package, plus the necessary firmware:

```
# apt update
# apt install nvidia-driver firmware-misc-nonfree
```

DKMS will build the nvidia module for your system, via the [DebianPkg: nvidia-kernel-dkms](#) package.

- *Note about Secureboot* : if you have [SecureBoot](#) enabled, you need to sign the resulting modules. Detailed instructions are available [here](#).

3. Restart your system to load the new driver.

Version 390.138 (legacy GPUs)

For support of GeForce 400 series and newer GPUs ([supported devices](#)).

1. Add "contrib" and "non-free" components to `/etc/apt/sources.list`, for example:

```
# Debian 10 "Buster"
deb http://deb.debian.org/debian/ buster main contrib non-free
```

2. Update the list of available packages, then we can install the [DebianPkg: nvidia-legacy-390xx-driver](#) package, plus the necessary firmware:

```
# apt update
# apt install nvidia-legacy-390xx-driver firmware-misc-nonfree
```

DKMS will build the nvidia module for your system, via the [DebianPkg: nvidia-legacy-390xx-kernel-dkms](#) package.

- *Note about Secureboot* : if you have [SecureBoot](#) enabled, you need to sign the resulting modules. Detailed instructions are available [here](#).

3. Restart your system to load the new driver.

Version 340.108 (legacy GPUs)

For support of GeForce 8 series through GeForce 300 series GPUs. ([🌐 supported devices](#)).

1. Add "contrib" and "non-free" components to `/etc/apt/sources.list`, for example:

```
# Debian 10 "Buster"
deb http://deb.debian.org/debian/ buster main contrib non-free
```

2. Update the list of available packages, then we can install the [DebianPkg: nvidia-legacy-340xx-driver](#) package:

```
# apt update
# apt install nvidia-legacy-340xx-driver
```

DKMS will build the nvidia module for your system, via the [DebianPkg: nvidia-legacy-340xx-kernel-dkms](#) package.

- *Note about Secureboot* : if you have [SecureBoot](#) enabled, you need to sign the resulting modules. Detailed instructions are available [here](#).

3. After, create an [Xorg server configuration file](#) and then restart your system to enable the nouveau blacklist.

Debian 9 "Stretch"

As of stretch, you don't need nvidia-xconfig anymore, and a xorg.conf file is not needed either in most situations. Also, the 340 series has been forked into its own series of packages to support older cards.

In some situations running nvidia-xconfig is still required for screen-locking and suspend/resume to work properly ([Open in nvidia-graphics-drivers/390.87-8~deb9u1: #922679: Locking screen with light-locker/nvidia-driver crashes system to blinking cursor: 922679](#) Xfce/lightdm/light-locker)

Version 418.152 (via stretch-backports)

For support of GeForce 700 series and newer GPUs ([🌐 supported devices](#)). For older devices, see [Version 340 \(legacy GPUs\)](#) and [Version 304 \(legacy GPUs\)](#).

1. Add stretch-backports as an additional new line to your `/etc/apt/sources.list`, for example:


```
# stretch-backports
deb http://deb.debian.org/debian stretch-backports main contrib non-free
```

2. Update the list of available packages, then we can install the [DebianPkg: nvidia-driver](#) package, plus the necessary firmware, from the backports repository:

```
# apt update
# apt install -t stretch-backports nvidia-driver firmware-misc-nonfree
```

DKMS will build the nvidia module for your system, via the [DebianPkg: nvidia-kernel-dkms](#) package.

3. Restart your system to enable the nouveau blacklist.

Version 390.138

For support of GeForce 400 series and higher GPUs ([supported devices](#)). For older devices, see [Version 340 \(legacy GPUs\)](#) and [Version 304 \(legacy GPUs\)](#).

1. Add "contrib" and "non-free" components to `/etc/apt/sources.list`, for example:

```
# Debian 9 "Stretch"
deb http://deb.debian.org/debian/ stretch main contrib non-free
```

2. Update the list of available packages, then we can install the [DebianPkg: nvidia-driver](#) package, plus the necessary firmware:

```
# apt update
# apt install nvidia-driver firmware-misc-nonfree
```

DKMS will build the nvidia module for your system, via the [DebianPkg: nvidia-kernel-dkms](#) package.

3. Restart your system to enable the nouveau blacklist.

Version 340.108 (legacy GPUs)

For support of GeForce 8 series through GeForce 300 series GPUs ([supported devices](#)).

1. Add "contrib" and "non-free" components to `/etc/apt/sources.list`, for example:

```
# Debian 9 "Stretch"
deb http://deb.debian.org/debian/ stretch main contrib non-free
```

2. Update the list of available packages, then we can install the [DebianPkg: nvidia-legacy-340xx-driver](#) package:

```
# apt update
# apt install nvidia-legacy-340xx-driver
```

DKMS will build the nvidia module for your system, via the [DebianPkg: nvidia-legacy-340xx-kernel-dkms](#) package.

After, create an [Xorg server configuration file](#) and then restart your system to enable the nouveau blacklist.

Version 304.137 (legacy GPUs)

For support of GeForce 6 series and GeForce 7 series GPUs ([supported devices](#)).

1. Add "contrib" and "non-free" components to `/etc/apt/sources.list`, for example:

```
# Debian 9 "Stretch"
deb http://deb.debian.org/debian/ stretch main contrib non-free
```

2. Update the list of available packages, then we can install the [DebianPkg: nvidia-legacy-304xx-driver](#) package:

```
# apt install nvidia-legacy-304xx-driver
```

DKMS will build the nvidia module for your system, via the [DebianPkg: nvidia-legacy-304xx-kernel-dkms](#) package.

After, create an [Xorg server configuration file](#) and then restart your system to enable the nouveau blacklist.

Installing 32-bit libraries on a 64-bit system

In many cases, such as when running proprietary 32-bit games from [Steam](#) or in [Wine](#), you may need 32-bit graphics libraries on your 64-bit system in order for them to function properly. This has been made much easier since [Debian 9/Stretch](#) and now requires minimal extra work.

Note that the following instructions assume that `sudo` is configured on your system. If it isn't, either follow the instructions on the [sudo](#) wiki page or omit the `sudo` and run these commands as root.

After installing the drivers, enable 32-bit multiarch and update your repository listing by running:

```
sudo dpkg --add-architecture i386 && sudo apt update
```

Afterwards, to install the 32-bit version of the NVIDIA libraries package, run:

```
sudo apt install nvidia-driver-libs:i386
```

Restarting the relevant applications may be necessary before they function correctly.

WARNING: If you're forced to use a legacy driver, you will want to instead install one of `nvidia-legacy-390xx-driver-libs:i386`, `nvidia-legacy-340xx-driver-libs:i386`, or `nvidia-legacy-304xx-driver-libs:i386`.

Wayland

The NVIDIA driver supports Wayland, with caveats. The 495-series driver (or newer) is recommended for the best experience, as older versions only support Wayland through an NVIDIA-specific API which is not supported by all desktops, and is generally less reliable.

The NVIDIA driver also lacks support for accelerated XWayland applications in current stable Debian versions. This means that if you run a Xorg-only application on your NVIDIA Wayland desktop (often proprietary video games), they will only be able to render on the CPU without taking advantage of GPU acceleration, leading to incredibly poor performance. 🌐 [Patches have been merged to resolve this](#), however this support will only be available in [Debian 12/Bookworm](#).

In terms of specific desktop support, GNOME supports NVIDIA Wayland sessions in both Debian 10 and Debian 11, though they call their support "preliminary". KDE Plasma supports NVIDIA Wayland sessions starting with Debian 11, though it requires some extra hoops to enable, and generally is not recommended. Refer to the Wayland section of the Debian KDE wiki page for up-to-date information: 🌐 https://wiki.debian.org/KDE#Wayland.2C_touchscreens.2C_autorotation.2C_hi-DPI

In Debian 12/bookworm, almost all issues should be resolved and most Wayland sessions should "just work" with the 525-series driver.

On GNOME desktops, although a proper version of the NVIDIA driver is used, the greeter (GDM3) may still not offer the option to start a Wayland session, either because kernel modesetting is not enabled, or because the hibernate/suspend/resume helper scripts have not been installed on the system.

To enable kernel modesetting with the NVIDIA driver:

```
# echo 'GRUB_CMDLINE_LINUX="$GRUB_CMDLINE_LINUX nvidia-drm.modeset=1"' > /etc/default/grub.d/nvidia-  
# update-grub
```

To install the hibernate/suspend/resume helper scripts:

```
# cp /usr/share/doc/xserver-xorg-video-nvidia/examples/nvidia-sleep.sh /usr/bin  
# cp /usr/share/doc/xserver-xorg-video-nvidia/examples/system-sleep/nvidia /usr/lib/systemd/system-s  
# cp /usr/share/doc/xserver-xorg-video-nvidia/examples/system/nvidia-* /etc/systemd/system/  
# systemctl daemon-reload && systemctl enable nvidia-hibernate nvidia-resume nvidia-suspend  
# echo "options nvidia NVreg_PreserveVideoMemoryAllocations=1" > /etc/modprobe.d/nvidia-power-manage
```

Once these changes are made, reboot the system. The GNOME greeter should then start new sessions under Wayland by default.

Tesla Drivers

The NVIDIA line-up of programmable "Tesla" devices, used primarily for simulations and large-scale calculations, also require separate driver packages to function correctly compared to the consumer-grade GeForce GPUs that are instead targeted for desktop and gaming usage.

In [Debian 10/Buster](#), the default [DebianPkg: nvidia-driver](#) package is based on the Tesla release. This was done in order to resolve several critical security issues, but it means that there is no need to install the separate package for Tesla devices to work. If you need a newer release, the 450-series driver is available in backports via the [DebianPkg: nvidia-tesla-450-driver](#) package.

In [Debian 11/Bullseye](#), the major 418, 440, and 450 releases of the Tesla driver are available and distinct from the default driver. They can be found in the [DebianPkg: nvidia-tesla-418-driver](#), [DebianPkg: nvidia-tesla-440-driver](#), and [DebianPkg: nvidia-tesla-450-driver](#) packages respectively.

In [Debian 12/Bookworm](#), the 470 release of the Tesla driver is available in the [DebianPkg: nvidia-tesla-470-driver](#) package.

The 32-bit libraries can be obtained by installing `nvidia-tesla-418-driver-libs:i386`, `nvidia-tesla-440-driver:i386`, or `nvidia-tesla-450-driver:i386` based on the version of your driver. [Multiarch](#) must be enabled.

Configuration

As the NVIDIA driver is not autodetected by [Xorg](#), a configuration file is required to be supplied. Modern Debian packages for the NVIDIA driver **should not require you to do anything listed here** as they handle this automatically during installation, but if you run into issues, or are using a much older version of Debian, you may try going through these steps.

Automatic

Install the `nvidia-xconfig` package, then run it with `sudo`. It will automatically generate a Xorg configuration file at `/etc/X11/xorg.conf`.

Manual

For example:

/etc/X11/xorg.conf.d/20-nvidia.conf

```
Section "Device"
    Identifier "My GPU"
    Driver "nvidia"
EndSection
```

The configuration file above can be created using these commands:

```
# mkdir -p /etc/X11/xorg.conf.d
# echo -e 'Section "Device"\n\tIdentifier "My GPU"\n\tDriver "nvidia"\nEndSection' > /etc/X11/xorg.conf.d/20-nvidia.conf
```

Please note that this configuration will break Xorg on Optimus systems. For such hardware, see [NVIDIA Optimus](#) instead.

Restart your system at this point to enable the nouveau driver blacklist.

[Additional configuration information](#) is available.

CUDA

Debian Unstable "Sid"

CUDA 11.8.89 is available from the non-free repository:

```
# apt install nvidia-cuda-dev nvidia-cuda-toolkit
```

This installs nvcc and friends. The visual profiler is in a separate package named [DebianPkg: nvidia-visual-profiler](#).

Debian 12 "Bookworm"

CUDA 11.8.89 is available from the non-free repository:

```
# apt install nvidia-cuda-dev nvidia-cuda-toolkit
```

This installs nvcc and friends. The visual profiler is in a separate package named [DebianPkg: nvidia-visual-profiler](#).

Debian 11 "Bullseye"

CUDA 11.2.2 is available from the non-free repository:

```
# apt install nvidia-cuda-dev nvidia-cuda-toolkit
```

This installs nvcc and friends. The visual profiler is in a separate package named [DebianPkg: nvidia-visual-profiler](#).

Debian 10 "Buster"

CUDA 9.2.148 is available from the non-free repository:

```
# apt install nvidia-cuda-dev nvidia-cuda-toolkit
```

And, if [Backports](#) are enabled, CUDA 11.2.1 is available similarly:

```
# apt -t buster-backports install nvidia-cuda-dev nvidia-cuda-toolkit
```

This installs nvcc and friends. The visual profiler is in a separate package named [DebianPkg: nvidia-visual-profiler](#).

Debian 9 "Stretch"

CUDA 8.0.44 is available from the non-free repository:

```
# apt install nvidia-cuda-dev nvidia-cuda-toolkit
```

And, if [Backports](#) are enabled, CUDA 9.1.85 is available similarly:

```
# apt -t stretch-backports install nvidia-cuda-dev nvidia-cuda-toolkit
```

This installs nvcc and friends. The visual profiler is in a separate package named [DebianPkg: nvidia-visual-profiler](#).

CUDA 8 only supports gcc 5.3.1, which is not available for Stretch. To compile you need to add `-ccbin clang-3.8` to the nvcc command line.

The Debian CUDA packages unfortunately do not include the Toolkit samples. To install these yourself you need to download the "Ubuntu 16.04" .run install file for CUDA 8 from <https://developer.nvidia.com/cuda-downloads>. Execute the .run file and (after accepting the license and agreeing to run on a non-supported system) skip the driver and toolkit installation and just select "Samples". Note before this step you must

```
export PERL5LIB=.
```

To compile the samples, you first need to set

```
export HOST_COMPILER=clang++-3.8
```

OptiX

To enable NVIDIA OptiX™ Ray Tracing Engine it is necessary to install an additional library:

```
# apt install libnvoptix1
```

A specific version is available for Tesla cards.

```
# apt install libnvidia-tesla-nvoptix1
```

Troubleshooting

Build failures

The NVIDIA driver can fail to build for several potential reasons.

1. You've installed a kernel from backports without installing the NVIDIA driver from backports. This can, in some cases, mean that the kernel is too new for the driver version you're attempting to use. Check this by viewing the package description for the NVIDIA driver where it will mention something along the lines of, "Building the kernel module has been tested up to Linux X.X" to figure out what's supported.

2. Particularly if you're on Debian Testing or Debian Unstable, the driver might not support your kernel yet. Often, new versions of the Linux kernel will explicitly require an update to the driver in order to be supported, so if the kernel package updates before the driver has a chance to be patched for it, you won't be able to use the NVIDIA driver. Solutions for this, from most to least recommended, are temporarily using an older kernel until the driver is updated, installing a newer version of the driver from Debian Experimental if one is available that supports your kernel version, or finding a patch for the build failure online that can be added to DKMS. The last two options are for advanced users **and may break your system or, in the case of adding a third-party patch, introduce security issues, forcing you to potentially reinstall completely or spend hours recovering your system. Caveat emptor.**

3. Legacy versions of the NVIDIA driver may not always support the latest kernel. For instance, the 304xx series driver, though available in the Debian Unstable

repository, does not support Linux 5.0 or newer. As necessary, you might consider using an older Debian version, or using Nouveau instead. Nouveau has decent performance with GPUs that are old enough to no longer be supported by the proprietary driver.

Driver stops working after upgrading Debian

When going between two major Debian releases (e.g., upgrading from Debian 9/Stretch to Debian 10/Buster), it's possible that the driver will stop functioning despite the build succeeding and no other issues being easily visible. This is most often caused by the [DebianPkg: nvidia-driver](#) package updating to a newer major release that no longer supports your hardware, as NVIDIA regularly drops support for older hardware generations. You will need to uninstall all your existing NVIDIA packages (refer to the section below for instructions on how to do so), and instead install the most recent legacy driver that still supports your GPU.

GPU isn't functional, even with a compatible driver version installed

If you have a hybrid graphics chipset and (after already installing the necessary driver package) 3D acceleration still does not work, you'll still need to choose one of the methods from the [NVIDIA Optimus](#) page in order to activate and make use of your NVIDIA card.

If you have an extremely modern NVIDIA GPU that was manufactured after the release of your Debian version, it may not work even after installing the newest backported driver that claims to support your card. If so, you likely need to upgrade the non-free firmware package on your system as well by installing the [DebianPkg: firmware-misc-nonfree](#) package from backports. For instance, on a Debian 10 system with backports enabled:

```
# apt install -t buster-backports firmware-misc-nonfree
```

After rebooting, the driver should be able to load the appropriate firmware.

Miscellaneous

- The NVIDIA driver conflicts with the nouveau DRM driver ([Closed in nvidia-kernel-common/20100522+1: #580894: nvidia-glx-conflicts-with-nouveau-KMS-driver: 580894](#)). The nouveau kernel module is blacklisted by the [DebianPkg: glx-alternative-nvidia](#) or [DebianPkg: nvidia-kernel-common](#) packages.
 - Restart your system after [configuring Xorg](#) for the NVIDIA driver.
 - From [DebianPkg: xserver-xorg-video-nouveau](#)'s README.Debian:


```
If you decide to switch to the proprietary driver, it is highly
recommended to reboot because it is incompatible with nouveau, and
unloading the latter is not easy and may lead to a blank console.
```

- If you can't change the screen brightness, open your Xorg configuration file (`/etc/X11/xorg.conf` or `/etc/X11/xorg.conf.d/20-nvidia.conf` depending on which method you used) and add

```
Option                "RegistryDwords" "EnableBrightnessControl=1;"
```

to the Device section. In some case (eg. GeForce GT 650M Mac Edition) it may cause screen flickering during boot time (just after grub screen), and system will not boot. In this case you should use instead add the following:

```
setpci -v -H1 -s 00:01.00 BRIDGE_CONTROL=0
```

to the file: `/etc/rc.local`

- You can check whether or not the kernel module for the NVIDIA driver has been loaded, in addition to what version is currently loaded, by running `/sbin/modinfo -F version nvidia-current`
- [Additional troubleshooting information](#) is available.

Uninstallation

If you run into issues with the drivers, switch to a different card, or simply want to use the open-source Nouveau drivers instead, uninstallation is made easy with recent versions of the drivers.

Also note that if issues with the driver prevent you from accessing a desktop, you can access a full-screen TTY with Ctrl-Alt-F3 (or almost any of the "F" keys).

You can remove all packages on your system with `nvidia` in the name by running:

```
# apt purge "*nvidia*"
```

And then reboot the system with:

```
systemctl reboot
```

This should leave you with a functioning system in **almost all cases**. If it seems to still be having issues, you may also try running:

```
# apt install --reinstall xserver-xorg-core xserver-xorg-video-nouveau
```

Or:

```
# X -configure
```

See Also

- [/Configuration](#)
- [/Troubleshooting](#)

- [NVIDIA Optimus](#)
- [Xorg](#)

[CategoryProprietarySoftware](#) [CategoryHardware](#) [CategoryVideo](#)